IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 2 February 2024, accepted 18 February 2024, date of publication 26 February 2024, date of current version 1 March 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3369679

RESEARCH ARTICLE

Spatio-Temporal Consistency for Multivariate Time-Series Representation Learning

SANGHO LEE[1,2], (Graduate Student Member, IEEE), WONJOON KIM[3], (Member, IEEE), AND YOUNGDOO SON[1,2], (Member, IEEE)

[1]Department of Industrial and Systems Engineering, Dongguk University, Seoul 04620, Republic of Korea
[2]Data Science Laboratory, Dongguk University, Seoul 04620, Republic of Korea
[3]Division of Future Convergence (HCI Science Major), Dongduk Women's University, Seoul 02748, Republic of Korea

Corresponding authors: Wonjoon Kim (wjkim@dongduk.ac.kr) and Youngdoo Son (youngdoo@dongguk.edu)

This work was supported in part by the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (MSIT) of Korea under Grant RS-2023-00208412 and Grant RS-2023-00239877; in part by the National Research Foundation of Korea (NRF) funded by the Ministry of Education under Grant RS-2023-00271054; and in part by Korea Institute of Planning and Evaluation for Technology in Food, Agriculture, Forestry (IPET) through Smart Agri Products Flow Storage Technology Development Program funded by Ministry of Agriculture, Food and Rural Affairs (MAFRA) of Korea under Grant 322050-3.

ABSTRACT Label sparsity in multivariate time series (MTS) makes using label information for practical applications challenging. Thus, unsupervised representation learning methods have gained attention to learn effective representations suitable for various MTS tasks without relying on labels. Recently, contrastive learning has emerged as a promising approach to generate robust representations by capturing underlying MTS information. However, the existing methods have some limitations, such as insufficient consideration of cross-variable relationships of MTS and high sensitivity to positive pairs. Therefore, we proposed a novel spatio-temporal contrastive representation learning method (STCR) designed to address these limitations. STCR focuses on learning robust representations by encouraging spatio-temporal consistency, which comprehensively considers spatial information as well as temporal dependencies in MTS. The results of extensive experiments on MTS classification and forecasting tasks demonstrate the efficacy of STCR in generating high-quality representations, achieving state-of-the-art performance on both tasks.

INDEX TERMS Contrastive learning, cross-variable relations, multivariate time series, representation learning, temporal dependency.

I. INTRODUCTION

Multivariate time series (MTS), which consists of synchronous variables related to one another over time, is a crucial data type that is used in various fields, such as engineering, finance, and medicine [1], [2], [3]. However, using MTS label information in practice is challenging because it is sometimes sparsely labeled [4], [5], [6]. Thus, learning universal representation suitable for various MTS tasks without label information has attracted considerable attention [7], [8].

Contrastive learning has achieved superior performance in generating representations without label information in a self-supervised manner [9], [10]. This approach encourages the representations of context views, which are the variants of the original data with transformations, especially data augmentations, to be similar; thereby, effective representations can be learned, achieving promising performance in various downstream tasks. However, the unique characteristics of time series, such as temporal dependency and irregular patterns, have hindered the application of conventional augmentation-based contrastive learning approaches [6]. For example, applying rotation, one of the augmentation methods typically used in the computer vision domain, to time series may corrupt their trends or patterns because of the change in the distribution [11]. Thus, contrastive learning methods specialized in time series have emerged by introducing augmentation methods suitable for time series or effectively reflecting temporal structural information [11], [12]. Although these methods can generate useful representations

The associate editor coordinating the review of this manuscript and approving it for publication was Wai-Keung Fung.

for various time-series tasks, two notable limitations exist: insufficient consideration of spatial information in MTS and high sensitivity to positive pairs.

Spatial information, which indicates cross-variable relations within MTS, is typically crucial in various tasks because it enables the representations to reflect the information of associated variables [13], [14]. Thus, several studies have attempted to consider spatial information of MTS to enhance model performance by enabling representations with similar structures to be close [8], [15], [16], [17], [18], [19]. In particular, most of them use graphs to capture spatial information owing to their superiority in capturing cross-variable relations in MTS [13]. However, they are often designed for a specific task, such as classification or forecasting [16], [17], [18], [19]. Although a few studies have only recently been proposed for MTS representation learning to focus on capturing spatial information [8], [20], their model performance is highly sensitive to how an MTS is converted into a graph because graphs created in different manners can contain different spatial information [21], [22]. Therefore, the existing methods are challenging to achieve high performance in tasks for MTS compared with those for univariate time series (UTS).

Moreover, the representations obtained by time-series contrastive learning are sensitive to configured positive pairs [11]. The following three typical selection strategies exist for constructing positive pairs:

1) *Subseries consistency* [23] constructs a time-series instance and its sampled subseries as a positive pair.
2) *Local consistency* [24] enforces the local smoothness by selecting neighboring segments as positive samples.
3) *Transformation consistency* [25] regards the augmented time series as a positive sample.

However, the representations obtained from these selection strategies are vulnerable to distribution changes in time series [11]. For example, the subseries and local consistencies are vulnerable to level shifts and anomalies in time series, respectively. Therefore, the existing methods may fail to configure appropriate positive and negative pairs, which causes performance degradation.

To address these problems, we proposed a **S**patio-**T**emporal **C**ontrastive **R**epresentation learning (STCR) to learn robust representations suitable for various MTS tasks by effectively capturing intrinsic spatial information along with temporal dependencies. In particular, we introduced *spatio-temporal consistency* that comprehensively considers underlying spatial relations in MTS captured by graphs with diverse edge structures and its temporal dependency while mitigating the drawbacks of conventional selection strategies. Then, we learned effective representations of MTS by encouraging spatio-temporal consistency.

To demonstrate the effectiveness of STCR, we conducted extensive comparative experiments on classification and forecasting, which are major tasks in MTS, with state-of-the-art methods (SOTAs). Consequently, STCR generates remarkably informative representations that perform better on classification and forecasting tasks than SOTAs.

This study has the following contributions:

- We proposed a novel representation learning that encourages *spatio-temporal consistency* to incorporate spatial information of MTS with temporal dependency.
- To effectively capture the inherent cross-variable relations in MTS, STCR converts an MTS into graphs with diverse edge structures and enforces their representations to be consistent.
- Our method generates robust representations that achieve superior performance for both classification and forecasting tasks compared to SOTAs by using spatio-temporal consistency for configuring positive pairs.

The remainder of this paper is organized as follows. In Section II, we briefly review the existing time-series representation learning methods. Next, we introduce the detailed algorithm of our approach in Section III. In Section IV, we present the experimental settings and results for demonstrating the effectiveness of the proposed method in learning high-quality representations suitable for various MTS tasks. Finally, concluding remarks are provided in Section V.

II. RELATED WORKS

Because MTS datasets often have insufficient label information, unsupervised representation learning for MTS has been studied extensively [6]. Generally, in the existing methods, temporal dependency is considered for generating MTS representations. For example, T-Loss [23] explicitly considered the temporal structure of MTS using a triplet loss with time-based negative sampling to handle subseries consistency. TNC [24] captured local temporal relationships between timestamps by constructing a graph, in which each timestamp is considered a node, and learning a graph representation. Moreover, TST [26], a transformer-based representation learning method for MTS, was introduced.

Recent studies on MTS representation learning have used contrastive learning that effectively captures underlying information in time series. In TS-TCC [25], several data augmentations, such as jittering, scaling, and permutation, were exploited to learn transformation-invariant representations of UTS and MTS. TS2Vec [11] introduced a hierarchical contrasting method to learn contextual representations for arbitrary subseries at various semantic levels. In addition, InfoTS [27] and AutoTCL [28] analyzed data augmentations in contrastive learning based on information theory and suggested the criteria for adaptively selecting optimal data augmentations for time series representation learning.

However, the existing methods have two notable limitations. First, although spatial information is crucial to analyze MTS because its variables affect each other [14], most studies have focused on reflecting temporal information rather than spatial information. Thus, their performance on MTS is low compared to that on UTS. Although there are recent studies that consider spatial information of MTS [8], [15], [16],

[17], [18], [19], most of them are designed for a specific task [16], [17], [18], [19]; also, the model performance is often sensitive to how an MTS is converted into a graph, the data type often used to capture cross-variable relations of MTS [29], because the graphs created in different ways can cause incoherent spatial information [21], [22]. Second, they used only one or two selection strategies, vulnerable to certain distribution changes of MTS, to construct positive pairs; thus, each method may inherit the drawback from the corresponding selection strategy [11].

By contrast, our method effectively captures inherent spatial information of MTS by considering incoherent cross-variable relations caused by graphs with different edge structures along with temporal dependency, even enhancing model performance by encouraging spatio-temporal consistency instead of the existing selection strategies. Using this approach, we generated a universal representation that improves performance for both MTS classification and forecasting tasks.

III. PROPOSED METHOD

To learn universal representations of MTS with spatio-temporal information, we proposed *STCR*, which is a novel contrastive representation learning method suitable for MTS.

A. PROBLEM STATEMENT

Let $\mathcal{X} = \{\boldsymbol{x}_i \in \mathbb{R}^{L_x \times V}\}_{i=1}^{N}$ be a set of MTS, where L_x and V are the sequence length and the number of variables, respectively. We define nonlinear mapping functions $f : \boldsymbol{x}_i \to \boldsymbol{z}_i^{\mathcal{T}}$, $g : \boldsymbol{x}_i \to \mathcal{S}_i$, and $h : [\boldsymbol{z}_i^{\mathcal{T}}; \boldsymbol{z}_i^{\mathcal{S}}] \to \boldsymbol{z}_i$, where $\boldsymbol{z}_i^{\mathcal{T}}$, $\mathcal{S}_i$, and $\boldsymbol{z}_i$ have dimensions $L_x \times d_{z^{\mathcal{T}}}$, $V \times d_{z^{\mathcal{S}}}$, and $L_x \times d_z$, respectively, and $\boldsymbol{z}_i^{\mathcal{S}} \in \mathbb{R}^{L_x \times d_{z^{\mathcal{S}}}}$ is obtained by multiplying $\boldsymbol{x}_i$ with $\mathcal{S}_i$. The objective is simultaneously learning f, g, and h to map $\boldsymbol{x}_i$ to its spatio-temporal representation z_i.

B. SPATIO-TEMPORAL CONTRASTIVE REPRESENTATION LEARNING

To learn an effective representation z_i of an MTS instance $\boldsymbol{x}_i$, we proposed *spatio-temporal consistency* that can simultaneously consider the spatial relations of MTS along with its temporal structure. STCR simultaneously trains f, g, and h by encouraging this consistency between two context views of $\boldsymbol{x}_i$ obtained from four modules: *random cropping*, *temporal embedding*, *spatial embedding*, and *projection*. Figure 1 displays an overview of the proposed method for generating the representation of an MTS instance.

1) RANDOM CROPPING

We randomly cropped an MTS instance to create two subseries, where each subseries is used to obtain a context view.

As shown in Figure 2, given an MTS instance $\boldsymbol{x}_i \in \mathbb{R}^{L_x \times V}$, we randomly extracted two subseries $\tilde{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{x}}_i'$, which have overlapping time segments $[s_1, e_1]$ and $[s_2, e_2]$ such that $0 < s_1 \leq s_2 \leq e_1 \leq e_2 \leq L_x$. Following [23], the learned

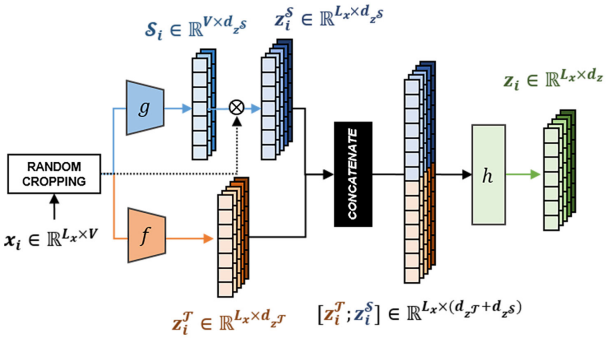


FIGURE 1. Overview of STCR.

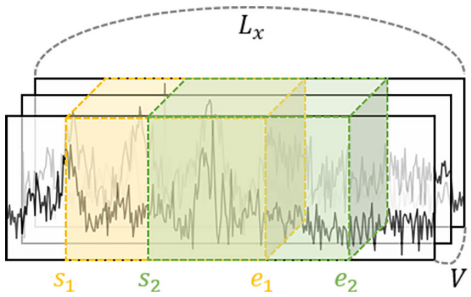


FIGURE 2. Random cropping.

representations on the overlapped segment $[s_2, e_1]$ should be consistent for two context views.

This approach enables us to learn position-agnostic representations while avoiding dimension collapse [11]. Note that random cropping is only used in the training phase.

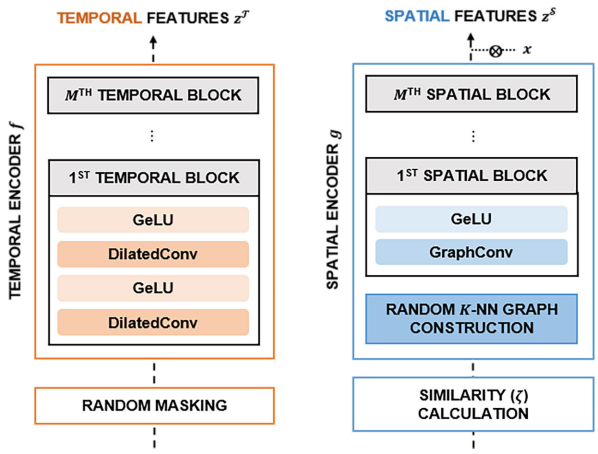
2) TEMPORAL EMBEDDING

As in Figure 3(a), to capture temporal structures of MTS, we first applied random masking to the subseries and subsequently obtained temporal features by passing the masked subseries through encoder f. Random masking helps to generate a transformation-invariant representation capturing the underlying temporal structure for MTS without a strong inductive bias [11].

Let $\tilde{\boldsymbol{x}}_i \in \mathbb{R}^{L_{\tilde{x}_i} \times V}$ be a subseries derived by random cropping for an MTS instance $\boldsymbol{x}_i$. Next, $\tilde{\boldsymbol{x}}_i$ is masked along the time axis with a binary mask $\boldsymbol{m}_i \in \{0, 1\}^{L_{\tilde{x}_i}}$ that is independently sampled from a Bernoulli distribution with p in every forward pass of a learning process. Note that the same mask vector is applied to every variable of MTS to focus on temporal consistency. Subsequently, the masked subseries is passed to encoder f to derive temporal features $\tilde{\boldsymbol{z}}_i^{\mathcal{T}} \in \mathbb{R}^{L_{\tilde{x}_i} \times d_{z^{\mathcal{T}}}}$ by

$$\tilde{\boldsymbol{z}}_i^{\mathcal{T}} = f(\boldsymbol{m}_i \times \tilde{\boldsymbol{x}}_i). \tag{1}$$

The encoder f has several temporal blocks that consist of one-dimensional dilated convolutional layers (*DilatedConv*) and *GeLU* activation functions for capturing the long-term dependency of MTS as a large receptive field [30].



(a) Temporal Embedding (b) Spatial Embedding

FIGURE 3. Two embedding modules.

3) SPATIAL EMBEDDING

In this module illustrated in Figure 3(b), we used a K-nearest neighbor (K-NN) graph, which effectively identifies the cross-variable relations within MTS [29].

First, we calculated similarities between variables in $\tilde{\boldsymbol{x}}_i$ using the heat kernel, a popular method used to construct the edges of K-NN graph [31] as follows:

$$\zeta_i^{(v,u)} = e^{-\|\tilde{\boldsymbol{x}}_{i,v} - \tilde{\boldsymbol{x}}_{i,u}\|^2/2}, \tag{2}$$

where v and u denote variables of $\tilde{\boldsymbol{x}}_i$. Then, we converted $\tilde{\boldsymbol{x}}_i$ into a K-NN graph, $\mathcal{G}_i^K$, where each node corresponds to a variable and the K largest similarities for each variable form the connected edges. Next, we constructed an encoder g consisting of spatial blocks with graph convolutional layers (*GraphConv*) followed by *GeLU* to handle the graph $\mathcal{G}_i^K$.

However, because various graphs can be constructed depending on K, the graphs with different Ks may contain different spatial information [21], [22]. Therefore, when we convert MTS into K-NN graphs, setting the appropriate K is a challenging problem. To recognize the underlying spatial structure of MTS by addressing this problem, we encouraged the representations of K-NN graphs to be consistent across various K. In particular, we randomly selected K from $[\lceil \rho \times V \rceil, V]$ to construct $\mathcal{G}_i^K$, where ρ is a *connection parameter*. The number of connections K is independently sampled in every forward pass of a learning process. Subsequently, we used an adjacency matrix A_i^K of $\mathcal{G}_i^K$ to capture spatial information $\mathcal{S}_i \in \mathbb{R}^{V \times d_{z^{\mathcal{S}}}}$ using the following expression:

$$\mathcal{S}_i = g(\tilde{\boldsymbol{x}}_i, A_i^K). \tag{3}$$

To reflect the information of the node itself and stable learning of *GraphConv*, we used self-loop and feature normalization techniques to A_i^K [32]. Then, (3) is formulated as

$$\mathcal{S}_i = g(\tilde{\boldsymbol{x}}_i, \bar{D}^{-\frac{1}{2}} \bar{A}_i^K \bar{D}^{-\frac{1}{2}}), \tag{4}$$

where $\bar{A}_i^K$ is an adjacency matrix with self-loop ($\bar{A}_i^K = A_i^K + I$), I is an identity diagonal matrix of A_i^K, and $\bar{D}$ is a degree matrix of $\bar{A}_i^K$ ($\bar{D}_{ii} = \sum_j \bar{A}_{ij}^K$).

$\mathcal{S}_i$ has no information about temporal structure; hence, we used $\mathcal{S}_i$ as a weight to allow $\tilde{\boldsymbol{x}}_i$ to reflect spatial relations between variables over time. We multiplied $\tilde{\boldsymbol{x}}_i$ with $\mathcal{S}_i$ to generate a feature vector $\tilde{\boldsymbol{z}}_i^{\mathcal{S}} \in \mathbb{R}^{L_{\tilde{x}_i} \times d_{z^{\mathcal{S}}}}$ as follows:

$$\tilde{\boldsymbol{z}}_i^{\mathcal{S}} = \tilde{\boldsymbol{x}}_i \times \mathcal{S}_i = \tilde{\boldsymbol{x}}_i \times g(\tilde{\boldsymbol{x}}_i, \bar{D}^{-\frac{1}{2}} \bar{A}_i^K \bar{D}^{-\frac{1}{2}}). \tag{5}$$

4) PROJECTION

To obtain a spatio-temporal feature vector $\tilde{\boldsymbol{z}}_i$, one of the context views of $\boldsymbol{x}_i$, we concatenated the temporal and spatial feature vectors and passed them through a projection head h:

$$\tilde{\boldsymbol{z}}_i = h\left([\tilde{\boldsymbol{z}}_i^{\mathcal{T}}; \tilde{\boldsymbol{z}}_i^{\mathcal{S}}]\right). \tag{6}$$

5) SPATIO-TEMPORAL CONSISTENCY

Since configuring positive pairs is essential in contrastive learning, previous works have used several selection strategies [23], [24], [25]. However, these strategies are based on strong assumptions of data distribution [11] and cannot consider cross-variable relations, which is crucial for various MTS tasks. To address this issue, we proposed a novel selection strategy, spatio-temporal consistency, which constructs the representations at the same timestamp in two augmented contexts as positive pairs. Specifically, an augmented context view is obtained by employing *random cropping* and *timestamp masking*, which helps improve the robustness of learned representations in terms of temporal dimension, in addition to introducing *randomness on* K, which is helpful in capturing inherent spatial information.

This approach has three advantages. First, cropping and masking do not require any strong assumptions of data distribution while helping learn robust representations and avoid representation collapse [11], [33], [34]. Second, since different K often leads to diverse spatial information [21], [22], randomness on K ensures the capture of inherent spatial information by considering diverse edge structures during model training. Finally, the sequential use of random cropping, timestamp masking, and randomness on K enhances the robustness of the learned representations by enforcing each timestamp to reconstruct itself in individual context views while reflecting spatio-temporal information.

6) LOSS FUNCTION

To simultaneously train f, g, and h, we encouraged the *spatio-temporal consistency* of two context views obtained from the same MTS instance using two contrastive loss functions: *instance-wise* and *timestamp-wise*.

Given two subseries, $\tilde{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{x}}_i'$, randomly cropped from $\boldsymbol{x}_i$, we obtained $\tilde{\boldsymbol{z}}_i$ and $\tilde{\boldsymbol{z}}_i'$ by passing $\tilde{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{x}}_i'$ through the temporal embedding, spatial embedding, and projection modules. Next, we used representations from other MTS instances at timestamp t in the same batch as negatives to calculate instance-wise loss $\mathcal{L}_i^{IW}$ using the following

Algorithm 1 Learning Procedure of STCR

Input: MTS dataset $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N\}$, temporal encoder f, spatial encoder g, projection head h, and number of optimization iterations Q
Output: Trained f, g, and h
Initialize f, g, and h.
while $q \leq Q$ **do**
  for $\boldsymbol{x}_i \in \mathcal{X}$ **do**
    Randomly crop $\boldsymbol{x}_i$ to overlapped $\tilde{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{x}}'_i$.
    Create two context views $\tilde{\boldsymbol{z}}_i$ and $\tilde{\boldsymbol{z}}'_i$ by (1)-(6).
    Calculate $\mathcal{L}_i^{IW}$ by (7).
    Calculate $\mathcal{L}_i^{TW}$ by (8).
  end for
  Update f, g, and h by (9).
end while

equation:

$$\mathcal{L}_i^{IW} = -\frac{1}{L_x}\sum_{t=1}^{L_x} log \frac{e^{\tilde{z}_{i,t}\cdot\tilde{z}'_{i,t}}}{\sum_{j=1}^{B}\left(e^{\tilde{z}_{i,t}\cdot\tilde{z}'_{j,t}} + \mathbb{1}_{[i\neq j]}e^{\tilde{z}_{i,t}\cdot\tilde{z}_{j,t}}\right)}, \quad (7)$$

where B is the batch size, and $\mathbb{1}$ is an indicator function.

However, although the instance-wise contrastive loss is effective for the classification task, it is insufficient for the forecasting task, which requires fine-grained representations for every timestamp [11]. Thus, we used a timestamp-wise contrastive loss function to obtain a discriminate representation over time for achieving a decent performance in forecasting as well as classification. For the timestamp-wise contrastive loss function, STCR defines the representations at the same timestamp from two context views of $\boldsymbol{x}_i$ as positive pairs, whereas those at different timestamps from $\boldsymbol{x}_i$ are defined as negative pairs; thereby, the timestamp-wise contrastive loss $\mathcal{L}_i^{TW}$ is formulated as follows:

$$\mathcal{L}_i^{TW} = -\frac{1}{L_x}\sum_{t=1}^{L_x} log \frac{e^{\tilde{z}_{i,t}\cdot\tilde{z}'_{i,t}}}{\sum_{t'\in\Phi}\left(e^{\tilde{z}_{i,t}\cdot\tilde{z}'_{i,t'}} + \mathbb{1}_{[t\neq t']}e^{\tilde{z}_{i,t}\cdot\tilde{z}_{i,t'}}\right)}, \quad (8)$$

where Φ is the set of the overlapping timestamps of two subseries. Finally, the overall loss function $\mathcal{L}$ of STCR is defined as follows:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}\left(\mathcal{L}_i^{IW} + \mathcal{L}_i^{TW}\right), \quad (9)$$

where N is the number of instances in the training dataset.

Through this learning process, we can obtain f, g, and h for generating universal representations suitable for various tasks of MTS. The proposed method is summarized in Algorithm 1. Furthermore, we analyzed the complexity of the proposed method from two perspectives of the number of learnable parameters and computational time in Section A.

IV. EXPERIMENTS

We evaluated the representations generated by STCR on both the classification and forecasting tasks of MTS. In this section, we first explain the implementation of the proposed method in detail and then elaborate on the experimental settings and results for each task.

A. IMPLEMENTATION DETAILS

1) PREPROCESSING

Following previous works [11], [23], [35], we used three preprocessing techniques to handle MTS:

- *Normalization of variables with different scales.* Because each variable in MTS has a different scale, we normalized each variable independently using a z-score. In forecasting tasks, we measured all evaluation metrics with normalized values.
- *Handling variable-length and missing values.* For a variable-length dataset, we padded all instances to have the same length by *NaNs*, meaning the missing values. When missing values occurred, we masked the corresponding positions as zero. Moreover, when we handled the graph of subseries in the spatial embedding module, we padded the node features to the same dimension as the sequence length of the input instance by zero.
- *Use of timestamp information.* When available, we also utilized timestamp information, including minute, hour, day-of-week, day-of-month, day-of-year, month-of-year, and week-of-year.

2) HYPERPARAMETERS

Because most previous studies for representation learning assume that label information and downstream tasks are unknown, selecting optimal hyperparameters based on the model performance is difficult. Hence, following [11], we used fixed hyperparameters regardless of the downstream tasks and did not perform hyperparameter optimization.

Referring to [11], the batch size B was set to 8, and the learning rate was 10^{-3}. The number of optimization iterations was set to 200 for datasets when the number of instances was less than 100,000; otherwise, it was set to 600. In the training phase, when the instance has a sequence length larger than 3,000, we clipped the sequence into segments with 3,000 timestamps. Encoder f for the temporal embedding module contained ten hidden temporal blocks consisting of two *DilatedConv* with an activation function, *GeLU* [36], and skip connections existed between neighboring blocks. For the ℓ-th block, the dilation parameter was set to 2^ℓ. The kernel size was set to 3, each *DilatedConv* had a dimension of 64, and a residual block mapped the hidden features to $d_{\mathcal{Z}\mathcal{T}}$-dimensional temporal features. We set p in the Bernoulli distribution for random masking to 0.5. Subsequently, the encoder g used in the spatial embedding module was configured with three hidden spatial blocks consisting of *GraphConv* and *GeLU*. We set the output dimensions of *GraphConv*s corresponding to each spatial

block to 128, 64, and d_{z^S}, respectively. In addition, for the spatial embedding module, we set the connection parameter ρ to 0.5 in the training phase because low-quality graphs with insufficient information for relationships between variables of MTS hinder obtaining effective spatial features; thereby, the range of K was $[\lceil 0.5 \times V \rceil, V]$. In the inference phase, we fixed K to the median value of the range of K used in the training phase. In the projection module, we configured h with two fully connected layers with 64 and d_z dimensions, respectively. Here, d_{z^T}, d_{z^S}, and d_z were equally set to 320.

All experiments were executed on the Pytorch platform using an Intel Core i9-10900X at 3.70 GHz CPU, 256 GB RAM, and GeForce RTX 3090 24GB GPU. STCR was implemented based on the official code of TS2Vec.[1]

B. MULTIVARIATE TIME-SERIES CLASSFICATION

1) EXPERIMENTAL SETTINGS

For classification tasks, a class should be assigned to each MTS instance; hence, instance-level representations are required. For a fair comparison, we used max pooling over all timestamps to obtain the instance-level representations following [11]. Then, following the same protocol with [11] and [23], we trained a support vector machine (SVM) classifier with the RBF kernel using the instance-level representations to predict the class of each instance. We set the penalty C with a grid search ranging in $[10^{-4}, 10^4]$ by cross-validation for the training dataset.

a: DATASETS

We used MTS classification datasets from the University of East Anglia and the University of California Riverside (UEA & UCR) time-series classification repository for evaluation. Among 30 MTS datasets in the repository, we selected 11 datasets with at least ten variables and 100 training instances. The repository separately provides training and test datasets, so we used the training dataset to train the model and the test dataset to evaluate the trained model.

b: BASELINES

To demonstrate that the representations learned by STCR are suitable for the classification task, we compared our method to SOTAs in unsupervised time-series representation learning, including T-Loss [23], TS-TCC [25], TST [26], TNC [24], TS2Vec [11], InfoTS [27], and AutoTCL [28] in addition to DTW [37]. Each method is summarized in Section II. We used the results reported in [11] for all baseline methods except InfoTS and AutoTCL. InfoTS[2] and AutoTCL[3] were implemented by their official codes.

c: EVALUATION METRIC

We evaluated the classification performance by measuring the accuracy score.

[1]https://github.com/yuezhihan/ts2vec
[2]https://github.com/chengw07/InfoTS
[3]https://github.com/AslanDing/AutoTCL

2) EXPERIMENTAL RESULTS

Table 1 depicts the classification performance of the proposed method compared with the baseline methods for 11 MTS datasets. The last row presents the average rank. Moreover, we performed statistical tests on the classification performance to ensure the significance of performance improvement by STCR. In particular, we used a two-sample Wilcoxon signed rank test [38] between STCR and each baseline method. The superscripts * and ** imply the rank test's p-value was smaller than 0.1 and 0.05, respectively.

STCR achieved the best performance in five of 11 datasets, and the average rank was 2.545, outperforming the baselines. Moreover, the statistical tests showed that STCR performed significantly better than SOTAs. In specific, the proposed method achieved superior performance on MTS datasets regardless of the number of variables V and sequence length L_x by reflecting both temporal and spatial information of MTS. Notably, STCR generally outperformed the baselines on datasets with a large number of variables by successfully capturing the spatial information of MTS. For example, for the *PEMS-SF* dataset, which is related to traffic involving sufficient spatial information with $V = 963$, STCR improved classification performance by approximately 10% than the second-best accuracy score.

By contrast, the baselines exhibited performance differences according to the number of variables or the sequence length. For example, TS-TCC performed poorly in the datasets with short sequences, such as *JapanesVowels* and *InsectWingbeat*. In addition, T-Loss and TS2Vec showed the worst performance in the datasets with many variables, including *PEMS-SF* and *FaceDetection*. These results confirmed that STCR is effective for MTS classification tasks.

C. MULTIVARIATE TIME-SERIES FORECASTING

1) EXPERIMENTAL SETTINGS

Given the last T observations $x_{t-T+1}, \cdots, x_t$, we predict the H upcoming observations $x_{t+1}, \cdots, x_{t+H}$ by using z_t, the spatio-temporal representation of the last timestamp t. Specifically, we adopted a protocol of [11], which used a linear regression model with L_2 regularization trained by using z_t as the input to directly predict future observations $\hat{x}_{t+1}, \cdots, \hat{x}_{t+H}$. We set the regularization coefficient α by a grid search on the validation dataset from search space $\{0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50, 100, 200, 500, 1000\}$.

a: DATASETS

To evaluate forecasting performance, we used four public datasets, including three ETT datasets [35] and Electricity dataset [39]. ETT datasets, including ETTh1, ETTh2, and ETTm1, collect two years of power transformer data, containing long-term trends, periodicity, and irregular patterns from two stations. ETTh1 and ETTh2 were collected every hour, and ETTm1 was collected in a 15-min unit. The Electricity dataset contained the electricity consumption data for 321 clients over three years. Following [35]

TABLE 1. Accuracy scores of STCR compared to baselines. For each dataset, the best score is highlighted in boldface.

Dataset	V	L_x	DTW	T-Loss	TNC	TS-TCC	TST	TS2Vec	InfoTS	AutoTCL	**STCR (ours)**
HandMovementDirection	10	400	0.231	**0.351**	0.324	0.243	0.243	0.338	**0.419**	0.378	0.351
PhonemeSpectra	11	217	0.151	0.222	0.207	**0.252**	0.085	0.233	0.199	0.103	0.186
JapaneseVowels	12	29	0.949	**0.989**	0.978	0.930	0.978	0.984	0.984	0.984	**0.989**
SpokenArabicDigits	13	93	0.963	0.905	0.934	0.970	0.923	**0.988**	0.948	0.925	0.955
NATOPS	24	51	0.883	0.917	0.911	0.822	0.850	0.928	0.917	0.589	**0.944**
FingerMovements	28	50	0.530	0.580	0.470	0.460	0.560	0.480	0.520	0.500	**0.590**
Heartbeat	61	405	0.717	0.741	0.746	**0.751**	0.746	0.683	0.741	0.722	0.746
MotorImagery	64	3000	0.500	0.580	0.500	**0.610**	0.500	0.510	0.500	0.540	0.550
FaceDetection	144	62	0.529	0.513	0.536	0.544	0.534	0.501	0.524	0.509	**0.557**
InsectWingbeat	200	30	-	0.156	0.469	0.264	0.105	0.466	0.469	**0.472**	0.453
PEMS-SF	963	144	0.711	0.676	0.699	0.734	0.740	0.682	0.717	0.821	**0.931**
Average Rank			6.455**	4.636**	5.000**	4.545*	5.727**	5.000*	4.182*	5.182**	**2.545**

and [11], we resampled the Electricity data into hourly data. In addition, we split ETT datasets into training, validation, and test datasets with 12, 4, and 4 months, respectively [35]. The Electricity was split into 60%, 20%, and 20% [11]. To demonstrate the performance of both short- and long-term forecasting, we lengthened the prediction length H progressively, from 1 day to 30 days for hourly data and from 6 hours to 7 days for minute data.

b: BASELINES

We compared the forecasting performance of STCR with TS2Vec, InfoTS, and AutoTCL, which are the SOTAs of MTS representation learning. We also compared our method with the SOTAs of MTS forecasting tasks, including Informer [35], LogTrans [40], LSTnet [41], TCN [30], and StemGNN [42].

- *Informer* is a transformer-based method for efficiently forecasting MTS.
- *LogTrans* improves forecasting performance by mitigating the memory bottleneck of the transformer.
- *LSTnet* uses both convolutional and recurrent neural networks to recognize short-term and long-term trends.
- *TCN* is an MTS forecasting method that introduces *DilatedConv* for the first time.
- *StemGNN* is an MTS forecasting method that uses spectral information with Fourier transform to improve forecasting performance.

We used the results reported in [11] for all baseline methods except InfoTS and AutoTCL. InfoTS and AutoTCL were implemented using their official codes.

c: EVALUATION METRICS

We used two evaluation metrics for forecasting tasks: mean squared error (MSE) and mean absolute error (MAE). MSE is measured as follows:

$$MSE = \frac{1}{HV}\sum_{h=1}^{H}\sum_{v=1}^{V}(x_{t+h}^{v} - \hat{x}_{t+h}^{v})^2, \tag{10}$$

where x_{t+h}^{v} and $\hat{x}_{t+h}^{v}$ are the observed and predicted values on variable v at timestamp $t+h$, respectively. Another metric, MAE, is measured as follows:

$$MAE = \frac{1}{HV}\sum_{h=1}^{H}\sum_{v=1}^{V}|x_{t+h}^{v} - \hat{x}_{t+h}^{v}|. \tag{11}$$

2) EXPERIMENTAL RESULTS

Table 2 shows the forecasting results of STCR compared with the baselines of forecasting tasks.

Although the proposed method showed comparable performance with the baselines, STCR achieved the lowest average MSE and MAE of 0.770 and 0.626, respectively. In addition, especially on long-term forecasting with long prediction length H, STCR outperformed the baselines because spatial information between variables of MTS enhances the ability to recognize long-term patterns [14], and *DilatedConv* can capture long-term dependency [43]. However, the forecasting performance of STCR for *Electricity* was slightly low. This dataset has only two variables, and even one of the two variables is timestamp information. Although we used additional timestamp features through preprocessing, these variables may not have sufficient structural relations among them. Thus, capturing the inherent relationship between variables may not be necessary. However, STCR showed performance comparable to several baseline methods, even for the *Electricity* dataset.

D. ABLATION STUDIES

We performed extensive ablation studies to demonstrate the effectiveness of each component of our method. STCR creates two context views using three *context components*: 1) random cropping, 2) random masking in the temporal embedding module, and 3) randomness on K in the spatial embedding module. The representations are obtained by encouraging spatio-temporal consistency with two *contrastive loss functions*: 1) instance-wise and 2) timestamp-wise. We compared our approach to three ablation models for context components: STCR without random cropping (*STCR w/o C*), STCR without random masking (*STCR w/o M*), and STCR without randomness on K (*STCR w/o K*). We also compared STCR to two ablation models for loss function: STCR without instance-wise loss (*STCR w/o* $\mathcal{L}^{IW}$) and STCR

TABLE 2. Forecasting results on MSE and MAE for STCR compared to the baselines. Here, H denotes the prediction length. For each dataset, the lowest MSE and MAE are highlighted in boldface.

Dataset	H	Informer		StemGNN		TCN		LogTrans		LSTnet		TS2Vec		InfoTS		AutoTCL		**STCR (ours)**	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	24	0.577	0.549	0.614	0.571	0.767	0.612	0.686	0.604	1.293	0.901	0.599	0.534	0.660	0.573	**0.402**	**0.444**	0.591	0.542
	48	0.685	0.625	0.748	0.618	0.713	0.617	0.766	0.757	1.456	0.960	0.629	0.555	0.711	0.606	**0.455**	**0.476**	0.626	0.565
	168	0.931	0.752	**0.663**	**0.608**	0.995	0.738	1.002	0.846	1.997	1.214	0.755	0.636	0.895	0.705	0.721	0.626	0.762	0.644
	336	1.128	0.873	0.927	0.730	1.175	0.800	1.362	0.952	2.655	1.369	**0.907**	**0.717**	1.045	0.778	0.973	0.744	0.910	0.721
	720	1.251	0.896	-	-	1.453	1.311	1.397	1.291	2.143	1.380	1.048	0.790	1.188	0.842	1.184	0.836	**0.997**	**0.770**
ETTh2	24	0.720	0.665	1.292	0.883	1.365	0.888	0.828	0.750	2.742	1.457	0.398	**0.461**	0.423	0.488	0.554	0.592	**0.375**	0.467
	48	1.457	1.001	1.099	0.847	1.395	0.960	1.806	1.034	3.567	1.687	**0.580**	**0.573**	0.663	0.630	0.973	0.783	0.605	0.597
	168	3.489	1.515	2.282	1.228	3.166	1.407	4.070	1.681	3.242	2.513	1.901	1.065	2.020	1.103	4.135	1.598	**1.587**	**1.002**
	336	2.723	1.340	3.086	1.351	3.256	1.481	3.875	1.763	2.544	2.591	2.304	1.215	2.391	1.224	3.202	1.365	**1.956**	**1.130**
	720	3.467	1.473	-	-	3.690	1.588	3.913	1.552	4.625	3.709	2.650	1.373	2.587	1.323	3.940	1.670	**2.092**	**1.159**
ETTm1	24	**0.323**	**0.369**	0.620	0.570	0.324	0.374	0.419	0.412	1.968	1.170	0.443	0.436	0.446	0.420	0.382	0.427	0.430	0.432
	48	0.494	0.503	0.744	0.628	**0.477**	**0.450**	0.507	0.583	1.999	1.215	0.582	0.515	0.583	0.500	0.508	0.500	0.560	0.514
	96	0.678	0.614	0.709	0.624	0.636	0.602	0.768	0.792	2.762	1.542	0.622	0.549	0.650	0.554	**0.505**	**0.500**	0.601	0.545
	288	1.056	0.786	0.843	0.683	1.270	1.351	1.462	1.320	1.257	2.076	0.709	0.609	0.729	0.605	**0.620**	**0.583**	0.665	0.589
	672	1.192	0.926	-	-	1.381	1.467	1.669	1.461	1.917	2.941	0.786	0.655	0.861	0.681	0.784	0.683	**0.779**	**0.654**
Electricity	24	0.312	0.387	0.439	0.388	0.305	0.384	0.297	0.374	0.356	0.419	**0.287**	**0.374**	-	-	-	-	0.331	0.406
	48	0.392	0.431	0.413	0.455	0.317	0.392	0.316	0.389	0.429	0.456	**0.307**	**0.388**	-	-	-	-	0.352	0.421
	168	0.515	0.509	0.506	0.518	0.358	0.423	0.426	0.466	0.372	0.425	**0.332**	**0.407**	-	-	-	-	0.375	0.438
	336	0.759	0.625	0.647	0.596	**0.349**	0.416	0.365	0.417	0.352	**0.409**	**0.349**	0.420	-	-	-	-	0.391	0.449
	720	0.969	0.788	-	-	0.447	0.486	**0.344**	**0.403**	0.380	0.443	0.375	0.438	-	-	-	-	0.414	0.467
Average		1.156	0.781	0.977	0.706	1.192	0.837	1.314	0.892	1.903	1.444	0.828	0.636	1.057	0.735	1.288	0.789	**0.770**	**0.626**

* All $H \geq 672$ cases of StemGNN fail for the out-of-memory even when $B = 1$; InfoTS and AutoTCL failed for the out-of-memory even when $B = 1$ in the Electricity dataset.

TABLE 3. Accuracy scores of ablation models and STCR. For each dataset, the best score is highlighted in boldface.

Dataset	Context Component			Loss Function		**STCR (ours)**
	w/o C	*w/o M*	*w/o K*	*w/o* $\mathcal{L}^{IW}$	*w/o* $\mathcal{L}^{TW}$	
HandMovementDirection	**0.392**	0.311	0.284	0.270	0.351	0.351
PhonemeSpectra	0.189	**0.209**	0.182	0.163	0.153	0.186
JapaneseVowels	0.976	0.984	0.984	0.965	0.981	**0.989**
SpokenArabicDigits	**0.960**	0.930	0.952	0.952	0.935	0.955
NATOPS	0.917	0.922	0.911	0.906	0.900	**0.944**
FingerMovements	0.520	0.480	0.470	0.480	0.520	**0.590**
Heartbeat	0.717	0.722	0.722	0.737	0.727	**0.746**
MotorImagery	0.500	0.500	0.490	0.500	0.520	**0.550**
FaceDetection	0.551	0.526	0.528	0.549	0.522	**0.557**
InsectWingbeat	0.449	0.449	0.441	0.438	0.437	**0.453**
PEMS-SF	0.908	0.850	0.919	0.902	0.919	**0.931**
Average Performance Drop Rate (%)	1.785	4.599	6.288	7.183	5.116	-

without timestamp-wise loss (*STCR w/o* $\mathcal{L}^{TW}$). The accuracy scores of ablation models for STCR are listed in Table 3.

1) RANDOM CROPPING

We compared the classification performance of STCR and *STCR w/o C*. Random cropping provides two subseries of an MTS instance with different lengths and positions. As presented in Table 3, *STCR w/o C* decreased approximately 1.785% compared to STCR on average for all datasets. Although the average performance drop rate was low compared to other ablation models, in some datasets, such as *JapaneseVowels* and *Heartbeat*, this model achieved a larger drop rate than others for context components; hence, it is one of the essential components of STCR.

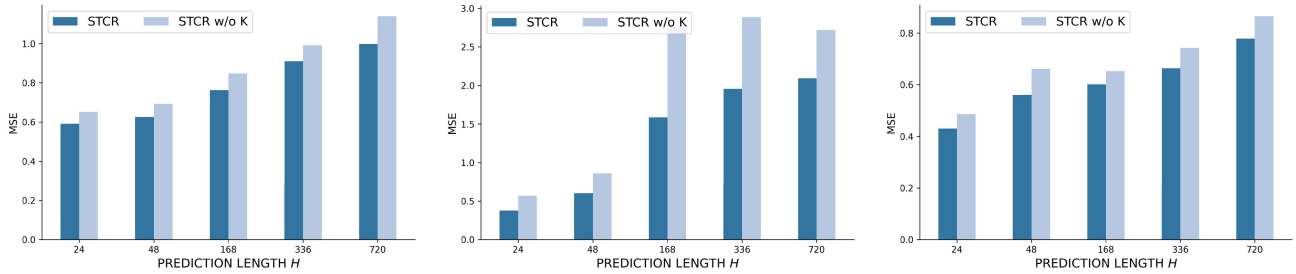
2) RANDOM MASKING

In general, most existing augmentation-based contrastive learning methods require a strong inductive bias, such as transformation invariance, that is not always suitable for handling MTS [11]. Therefore, we only used random masking, a transformation that does not require strong assumptions. To verify the efficacy of random masking, we compared the classification performance of STCR and *STCR w/o M*. As presented in Table 3, *STCR w/o M* showed the classification performance of 4.599% decrease than STCR on average. Thus, we demonstrated that random masking can improve representation quality without unrealistic assumptions.

3) RANDOMNESS ON K

We conducted three experiments to investigate the effects of *randomness on K* on classification performance, the ability to recognize long-term patterns, and the robustness to ρ.

First, we compared the classification performance of STCR and *STCR w/o K* to show the effect of the randomness on K. For the ablation model, we fixed K to $0.5 \times V$ in the training phase. In Table 3, the classification performance of



(a) ETTh1 (b) ETTh2 (c) ETTm1

FIGURE 4. Difference of MSE between STCR and *STCR w/o K* on ETT datasets with various *H*s.

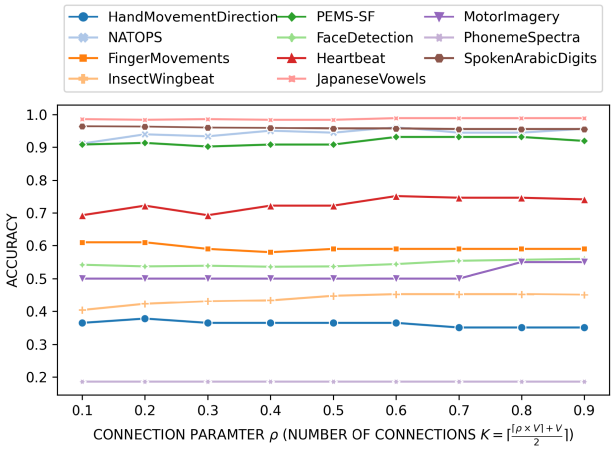


FIGURE 5. Accuracy scores of STCR across various connection parameters, ρ, (and corresponding K).

TABLE 4. MSE and MAE of ablation models. For each dataset, the lowest results are highlighted in boldface. (AEGR: Average error growth rate).

Dataset	H	*w/o* $\mathcal{L}^{IW}$		*w/o* $\mathcal{L}^{TW}$		STCR	
		MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	24	**0.588**	**0.542**	0.688	0.584	0.591	**0.542**
	48	0.628	0.569	0.730	0.611	**0.626**	**0.565**
	168	0.766	0.650	0.877	0.692	**0.762**	**0.644**
	336	0.955	0.745	1.024	0.766	**0.910**	**0.721**
	720	1.034	0.792	1.157	0.841	**0.997**	**0.770**
ETTh2	24	**0.368**	**0.462**	0.434	0.481	0.375	0.467
	48	0.616	0.595	0.657	0.610	**0.605**	**0.597**
	168	1.693	1.016	1.726	1.017	**1.587**	**1.002**
	336	2.614	1.264	2.652	1.288	**1.956**	**1.130**
	720	2.139	1.177	2.330	1.189	**2.092**	**1.159**
ETTm1	24	0.463	0.465	0.484	0.460	**0.430**	**0.432**
	48	0.612	0.552	0.638	0.547	**0.560**	**0.514**
	96	0.642	0.573	0.652	0.572	**0.601**	**0.545**
	288	0.720	0.619	0.745	0.630	**0.665**	**0.589**
	672	0.818	0.676	0.888	0.699	**0.779**	**0.654**
AEGR (%)		8.274	3.543	15.854	6.350	-	-

STCR w/o K was substantially decreased compared to that of STCR. Specifically, because random K explicitly affects capturing spatial structures of MTS, *STCR w/o K* showed the largest average drop rate of 6.288% compared to other ablation models in terms of context components.

Next, we verified that randomness on K enhances the ability to recognize long-term patterns as well as the short-term patterns of MTS. As shown in Figure 4, STCR outperformed *STCR w/o K* on both short- and long-term forecasting tasks. Moreover, as the prediction length H increased, the performance difference between STCR and *STCR w/o K* gradually increased in the ETTh2 dataset. Thus, we demonstrated that random K can improve forecasting performance, even if the prediction length is long, by capturing inherent spatial information from the graphs with diverse K.

We demonstrated the robustness against connection parameter ρ used to determine the number of connections K in the inference phase. Figure 5 shows the accuracy scores of STCR by varying $\rho \in [0.1, 0.9]$ (and the corresponding K). We observed that the classification performances for various ρ values are similar for most datasets, which implies that STCR is not highly sensitive to ρ owing to the randomness on K in the training phase. Therefore, random K enhances robustness to the number of connections, which may be challenging to set appropriately, by encouraging consistent representations for graphs consisting of various Ks.

Therefore, we confirmed that random K improves classification and forecasting performances by well-reflecting inherent structural information for relationships between variables of MTS and enhancing robustness to K.

4) INSTANCE-WISE CONTRASTIVE LOSS FUNCTION

As presented in Table 3, the classification performance of *STCR w/o* $\mathcal{L}^{IW}$ was highly decreased for most datasets. Also, the ablation model showed a larger average performance drop rate (7.183%) than *STCR w/o* $\mathcal{L}^{TW}$. Thus, the instance-wise contrastive loss, $\mathcal{L}^{IW}$, fulfills a more important role for the classification task than $\mathcal{L}^{TW}$ by encouraging instances belonging to the same class to be close to each other with spatio-temporal consistency at the instance level.

5) TIMESTAMP-WISE CONTRASTIVE LOSS FUNCTION

This loss function is useful for forecasting tasks by considering spatio-temporal consistency at the timestamp level.

TABLE 5. Accuracy scores of ablation models against selection strategies for configuring positive pairs compared to spatio-temporal consistency. For each dataset, the best score is highlighted in boldface.

Dataset	**STCR (ours)**	→ Subseries	→ Local	→ Jittering	→ Fliping	→ Permutation
HandMovementDirection	0.351	**0.405**	0.257	0.257	0.230	0.311
PhonemeSpectra	0.186	0.129	0.109	0.180	0.186	**0.199**
JapaneseVowels	**0.989**	0.968	0.981	0.968	0.981	0.976
SpokenArabicDigits	**0.955**	0.940	0.910	0.953	0.921	0.937
NATOPS	**0.944**	0.933	0.828	0.894	0.928	0.894
FingerMovements	**0.590**	0.560	0.530	0.450	0.570	**0.590**
Heartbeat	**0.746**	**0.746**	0.732	0.741	0.722	0.722
MotorImagery	**0.550**	0.490	0.500	0.540	0.500	0.500
FaceDetection	0.557	**0.558**	0.547	0.548	0.537	0.543
InsectWingbeat	**0.453**	0.417	0.365	0.445	0.446	0.434
PEMS-SF	**0.931**	0.908	0.896	0.861	0.902	0.844
Average	**0.659**	0.641	0.605	0.622	0.629	0.632

Table 4 provides the MSE and MAE results of two ablation models, *STCR w/o* $\mathcal{L}^{IW}$ and *STCR w/o* $\mathcal{L}^{TW}$, and STCR on ETT datasets. On average, two ablation models showed worse forecasting performance than STCR. Specifically, although STCR showed slightly superior performance than ablation models on short-term forecasting, STCR achieved overwhelming performance for long-term forecasting. Moreover, *STCR w/o* $\mathcal{L}^{TW}$ exhibited a higher average error growth rate than *STCR w/o* $\mathcal{L}^{IW}$ did. This result implies that the instance-wise and timestamp-wise contrastive loss functions can improve forecasting performance as well as classification performance; in addition, the timestamp-wise contrastive loss enables the model to effectively generate representations more suitable for forecasting tasks than the instance-level contrastive loss.

6) SPATIO-TEMPORAL CONSISTENCY

The proposed method enhances the model performance by obtaining robust representations using spatio-temporal consistency instead of conventional selection strategies for configuring positive pairs. To demonstrate its effectiveness, we performed additional ablation studies against the proposed spatio-temporal consistency. We compared the classification performance of our method to those of five ablation models: two STCRs replacing the proposed spatio-temporal consistency with subseries and local consistencies, respectively, and three STCRs replacing random masking with jittering, flipping, and permutation, respectively. Consequently, as shown in Table 5, the spatio-temporal consistency outperformed the ablation models in most datasets and achieved the best average accuracy score.

V. CONCLUSION

We proposed a novel representation learning method, STCR, to learn universal representations for various MTS tasks by encouraging spatio-temporal consistency instead of conventional selection strategies to capture inherent spatial information and temporal dependency. We obtained two context views using random cropping, temporal embedding, spatial embedding, and projection modules; a spatio-temporal representation is learned by instance-wise and timestamp-wise contrastive loss functions encouraging spatio-temporal consistency between two context views. Through extensive experiments on classification and forecasting tasks, we demonstrated that STCR is useful for generating robust representations while performing better than SOTAs.

Limitations. The proposed method has two limitations. First, STCR shows relatively low performance in some datasets, especially for those with few variables. Because STCR reflects structural relations among the variables in MTS by a K-NN graph, the performance for the datasets with few variables or no meaningful spatial information can be suboptimal. Second, although STCR shows comparable processing times with TS2Vec, the SOTA with high efficiency in MTS representation learning, in a reasonable number of variables, the processing time rapidly increases with the number of variables (see Table 6). For example, in *PEMS-SF* dataset, which has 963 variables, STCR remarkably improved classification performance by about 36% compared to TS2Vec by reflecting spatial information (see Table 1). However, in terms of computational time, since the K-NN graph constructed by these variables requires a large amount of additional computation in learning the graph neural network and processing with the learned network in the inference phase ($\mathcal{O}(V^2)$ where V is the number of variables or nodes), STCR performed slower than TS2Vec.

Future research directions. One possible solution to handle MTS datasets with a small number of variables using our approach is to construct K-NN graphs in the latent space formed by simple embedding. If these representations in the latent space are learned to preserve spatial information on cross-variable relations of the input MTS data, their K-NN graphs can provide an effect similar to our approach. Besides, by constructing graphs with latent representations, STCR can also be adapted to UTS. Meanwhile, to enhance the efficiency of our method, we can devise a technique that simplifies a large graph to a small graph without losing spatial information. Furthermore, we expect to enhance the

capability to generate more high-quality representations by developing an advanced temporal embedding to capture more sophisticated temporal information of MTS and successfully incorporating it with inherent spatial information of MTS.

APPENDIX A
COMPLEXITY ANALYSIS

Since our encoder has an additional embedding module to handle the K-NN graph compared to TS2Vec [11], which is the SOTA with high efficiency in MTS representation learning, it requires more computation. Here, we discuss the efficiency of the proposed method from two perspectives: the number of parameters and computational time.

To examine the number of learnable parameters of our method, we denoted the number of variables passing through l-th layer of the temporal encoder as $V^{(l)}$, and the sequence length after l'-th layer of the spatial encoder as $L_x^{(l')}$. L and L' are the number of layers in the temporal and spatial encoders, respectively. In addition, the size of the one-dimensional convolution filter in the temporal encoder is denoted as d_k. Thus, the STCR has the following number of learnable parameters:

$$\sum_{l=1}^{L} V^{(l-1)}V^{(l)}d_k + \sum_{l'=1}^{L'} L_x^{(l'-1)}L_x^{(l')} + (V^{(L)} + L_x^{(L')})d_h + d_h d_z, \tag{12}$$

where $V^{(0)}$ and $L_x^{(0)}$ are the number of variables and sequence lengths of the original time series, respectively. In addition, d_h is the number of hidden features in the projection head. The first and second terms are the number of learnable parameters of the temporal and spatial encoders, respectively, and the others refer to the number of parameters in the projection head consisting of two fully connected layers.

By contrast, TS2Vec does not consider spatial information of multivariate time series, so it has the following number of learnable parameters:

$$\tilde{V}V^{(0)} + \sum_{l=1}^{L} V^{(l-1)}V^{(l)}d_k \tag{13}$$

where $\tilde{V}$ is the number of variables, and $V^{(0)}$ is the number after the input projection layer. The first term indicates the number of parameters in the input projection layer, and the second term regards the rest parts of the encoder. Therefore, the total number of learnable parameters for TS2Vec and STCR differs approximately as much as the number of learnable parameters required by the spatial encoder of STCR.

Note that as the number of instances increases, the time complexity of STCR increases linearly, similar to other general methods that do not perform additional operations between instances. In addition, the memory complexity is generally irrelevant to the number of instances, assuming that the batch size is equal. Thus, unlike the existing methods without consideration of spatial information, the number of learnable parameters of STCR is significantly affected by the number of variables; that is, the computational resources and time complexity of the proposed method become especially high when the number of variables increases rather than the number of instances increases.

TABLE 6. Training time per iteration and inference time per instance for STCR and TS2Vec.

Dataset	TS2Vec		STCR	
	Training	Inference	Training	Inference
HandMovementDirection	0.223	6.89e-4	0.209	2.86e-3
PhonemeSpectra	0.193	2.02e-4	0.184	2.45e-3
JapaneseVowels	0.059	2.46e-4	0.062	2.53e-3
SpokenArabicDigits	0.158	1.73e-4	0.167	2.48e-3
NATOPS	0.148	1.94e-4	0.174	2.78e-3
FingerMovements	0.148	2.00e-4	0.180	2.91e-3
Heartbeat	0.227	3.85e-4	0.349	5.65e-3
MotorImagery	0.508	1.13e-3	0.562	8.36e-3
FaceDetection	0.151	2.01e-4	1.167	2.65e-2
InsectWingbeat	0.138	2.07e-4	2.133	6.33e-2
PEMS-SF	0.178	5.61e-4	45.91	1.09e-0

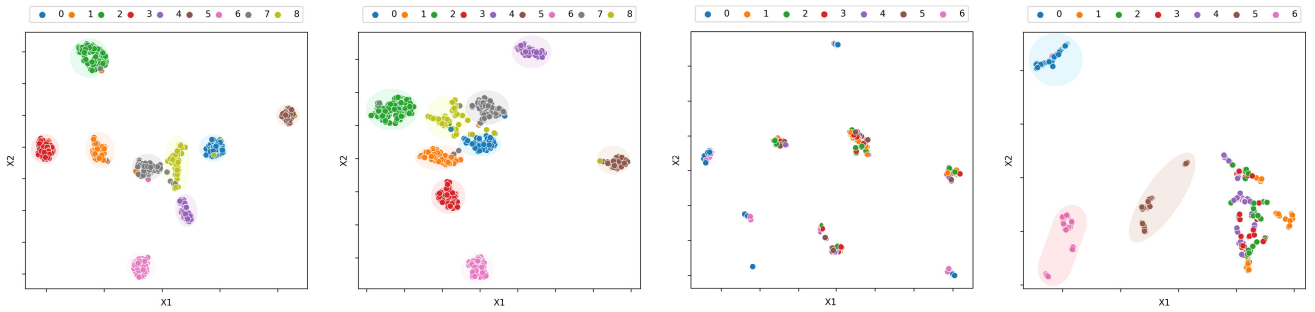
TABLE 7. Accuracy scores of STCR compared to PatchTST, TimesNet, and DLinear. For each dataset, the best score is highlighted in boldface.

Dataset	PatchTST	TimesNet	DLinear	STCR
HandMovementDirection	0.514	0.527	**0.568**	0.351
PhonemeSpectra	0.050	0.061	0.057	**0.186**
JapaneseVowels	0.927	0.965	0.962	**0.989**
SpokenArabicDigits	0.920	**0.960**	0.955	0.955
NATOPS	0.800	0.894	0.939	**0.944**
FingerMovements	0.490	0.520	0.510	**0.590**
Heartbeat	0.712	0.737	0.727	**0.746**
MotorImagery	0.500	0.500	0.500	**0.550**
FaceFaceDetection	0.500	**0.651**	0.646	0.557
InsectInsectWingbeat	0.521	**0.548**	0.195	0.453
PEMS-SF	0.855	0.861	0.821	**0.931**

Subsequently, we also compared the processing time of our method to that of TS2Vec. For two methods, we reported the training time per iteration and the inference time per instance in Table 6. The proposed method shows comparable processing times with TS2Vec with a reasonable number of variables, but the processing time rapidly increases with the number of variables. For example, in the PEMS-SF dataset, which has 963 variables, STCR remarkably improved classification performance by about 25 percentage points (or 36.5%) compared to TS2Vec by reflecting spatial information (see Table 1 in the manuscript). However, in terms of computational time, since the graph constructed by these variables requires a large amount of additional computation in learning the graph neural network and processing with the learned network in the inference phase ($\mathcal{O}(V^2)$ where V is the number of variables or nodes), STCR can be slower than TS2Vec.

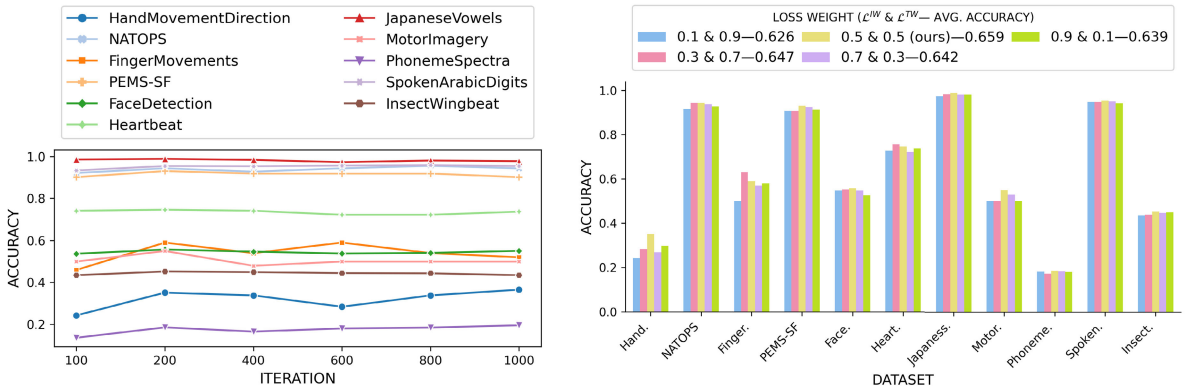
APPENDIX B
DISCUSSION ON TEMPORAL EMBEDDING

PatchTST [44], TimesNet [45], and DLinear [46], which are the recent methods for time-series analysis by introducing individual end-to-end frameworks explicitly using objective functions specialized for each classification or



(a) TS2Vec (Japanese.) (b) STCR (Japanese.) (c) TS2Vec (PEMS-SF) (d) STCR (PEMS-SF)

FIGURE 6. Visualization on JapaneseVowels ((a) and (b)) and PEMS-SF ((c) and (d)).



(a) Iteration numbers for optimization (Q) (b) Weights on two loss functions

FIGURE 7. Accuracy scores of STCR (a) across various iteration numbers for optimization and (b) when assigning different weights for two loss functions.

forecasting task, focused on capturing fine-grained temporal dependencies by various advanced techniques, such as masked time-series modeling. Here, we further compared the classification and forecasting performances of our method with those of the three recent methods on 11 UEA & UCR datasets for the classification task and ETT datasets for the forecasting task. PatchTST, TimesNet, and DLinear were implemented based on the official code of TimesNet.[4]

As shown in Table 7, for the classification task, STCR achieved outstanding performance in most datasets compared to these methods by effectively considering cross-variable relations that ensure representations with similar structures are close. Especially in the PEMS-SF dataset, which contains abundant spatial information with 963 variables related to traffic, STCR remarkably outperformed them.

However, in Table 8, our method exhibited similar or relatively high MSE and MAE compared to the three methods in the forecasting tasks that are remarkably affected by sophisticated temporal dependencies rather than spatial information. Therefore, in future work, we can enhance the model performance, especially for forecasting tasks, by designing an innovative temporal embedding module to capture more sophisticated temporal information of MTS and successfully incorporating it with inherent spatial information of MTS.

[4]https://github.com/thuml/Time-Series-Library

TABLE 8. Forecasting results on MSE and MAE for STCR compared to PatchTST, TimesNet, and DLinear.

Dataset	H	PatchTST		TimesNet		DLinear		STCR	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	24	0.423	0.455	1.780	0.972	0.632	0.578	0.591	0.542
	48	0.481	0.500	1.091	0.761	0.672	0.594	0.626	0.565
	168	0.628	0.571	1.308	0.858	0.834	0.681	0.762	0.644
	336	0.979	0.714	1.900	1.002	0.916	0.705	0.910	0.721
	720	1.055	0.776	1.656	0.985	0.996	0.762	0.997	0.770
ETTh2	24	0.344	0.410	0.458	0.477	0.450	0.476	0.375	0.467
	48	0.716	0.571	1.334	0.808	0.780	0.629	0.605	0.597
	168	0.875	0.504	0.822	0.644	1.628	0.896	1.587	1.002
	336	0.995	0.612	1.316	0.807	1.718	0.915	1.956	1.130
	720	3.310	0.807	1.821	0.934	3.197	1.257	2.092	1.159
ETTm1	24	0.266	0.343	0.378	0.404	0.376	0.449	0.430	0.432
	48	0.343	0.382	0.547	0.506	0.484	0.514	0.560	0.514
	96	0.420	0.453	0.531	0.501	0.505	0.510	0.601	0.545
	288	0.495	0.501	0.704	0.596	0.524	0.509	0.665	0.589
	672	0.719	0.598	0.804	0.615	0.655	0.592	0.779	0.654

APPENDIX C
GRAPHICAL ANALYSIS

We performed a graphical analysis of the representations learned by the proposed method using UMAP [47].

TABLE 9. Notations.

Notation	Description	Dimension
$\mathcal{X}$	Set of multivariate time-series instances	-
N	Number of instances	-
$\boldsymbol{x}_i$	Multivariate time-series instance	$L_x \times V$
L_x	Sequence length of $\boldsymbol{x}_i$	-
V	Number of variables	-
$\tilde{\boldsymbol{x}}_i$	Randomly extracted subseries	$L_{\tilde{x}_i} \times V$
$L_{\tilde{x}_i}$	Sequence length of $\tilde{\boldsymbol{x}}_i$	-
$\tilde{\boldsymbol{x}}'_i$	Another randomly extracted subseries	$L_{\tilde{x}'_i} \times V$
$L_{\tilde{x}'_i}$	Sequence length of $\tilde{\boldsymbol{x}}'_i$	-
f	Temporal encoder	-
$\boldsymbol{z}_i^{\mathcal{T}}$	Temporal features of $\boldsymbol{x}_i$	$L_x \times d_{z^{\mathcal{T}}}$
$\tilde{\boldsymbol{z}}_i^{\mathcal{T}}$	Temporal features of $\tilde{\boldsymbol{x}}_i$	$L_{\tilde{x}_i} \times d_{z^{\mathcal{T}}}$
$\boldsymbol{m}_i$	Binary mask $\in \{0, 1\}$ of $\tilde{\boldsymbol{x}}_i$	$L_{\tilde{x}_i}$
g	Spatial encoder	-
$\boldsymbol{z}_i^{\mathcal{S}}$	Spatial features of $\boldsymbol{x}_i$	$L_x \times d_{z^{\mathcal{S}}}$
$\tilde{\boldsymbol{z}}_i^{\mathcal{S}}$	Spatial features of $\tilde{\boldsymbol{x}}_i$	$L_{\tilde{x}_i} \times d_{z^{\mathcal{S}}}$
$\zeta_i^{(v,u)}$	Similarity between variables v and u of $\tilde{\boldsymbol{x}}_i$	-
ρ	Connection parameter	-
K	Number of connections	-
$\mathcal{G}_i^K$	Graph with K connections of $\tilde{\boldsymbol{x}}_i$	-
A_i^K	Adjacency matrix of $\mathcal{G}_i^K$	$V \times V$
$\bar{A}_i^K$	Adjacency matrix with self-loop ($\bar{A}_i^K = A_i^K + I$)	$V \times V$
$\bar{D}$	Degree matrix of $\bar{A}_i^K$	$V \times V$
I	Identity diagonal matrix of A_i^K	$V \times V$
$\mathcal{S}_i$	Spatial information of $\tilde{\boldsymbol{x}}_i$	$V \times d_{z^{\mathcal{S}}}$
h	Projection head	-
$\boldsymbol{z}_i$	Spatio-temporal representation of $\boldsymbol{x}_i$	$L_x \times d_z$
$\tilde{\boldsymbol{z}}_i$	Spatio-temporal representation of $\tilde{\boldsymbol{x}}_i$	$L_{\tilde{x}_i} \times d_z$
$\tilde{\boldsymbol{z}}'_i$	Spatio-temporal representation of $\tilde{\boldsymbol{x}}'_i$	$L_{\tilde{x}'_i} \times d_z$
$\mathcal{L}^{IW}$	Instance-wise contrastive loss	-
$\mathcal{L}^{TW}$	Timestamp-wise contrastive loss	-
$\mathcal{L}$	Overall loss	-
B	Batch size	-
Φ	Set of the overlapping timestamps of two subseries	-

We selected two datasets, PEMS-SF and JapaneseVowels, which exhibit the largest and smallest performance gaps, respectively, between STCR and TS2Vec.

As shown in Figure 6, for the JapaneseVowels dataset, we observed that both methods form well-distinguished clusters for all classes. However, for the PEMS-SF dataset, the representations learned by STCR form significantly better distinct groups for each class than TS2Vec, especially the classes 0, 5, and 6. Through this analysis, we can reaffirm the effectiveness of the proposed method.

APPENDIX D
SENSITIVITY ANALYSIS

To investigate the impact of hyperparameters used in our method, we performed sensitivity analyses for the number of optimization iterations Q and the assigned weights for two loss functions, $\mathcal{L}^{IW}$ and $\mathcal{L}^{TW}$.

Iteration numbers for optimization. We observed the classification performance of the proposed method on 11 MTS datasets by varying optimization iterations, Q. As shown in Figure 7(a), in most datasets, the classification performance gradually increased until about 200 iterations, whereas slightly decreasing or being stable after that.

Weights on two loss functions. We compared the classification performances of STCR on 11 MTS datasets when assigning different weights for instance-wise and timestamp-wise contrastive losses, $\mathcal{L}^{IW}$ and $\mathcal{L}^{TW}$. As shown in Figure 7(b), although assigning the same weights achieved the best average accuracy, we show that our approach is not sensitive to the weights in most datasets.

APPENDIX E
NOTATIONS

For a comprehensive understanding, we provided a notation table, including the dimensions of the variables, in Table 9.

REFERENCES

[1] J. Choi, Y. Son, and M. K. Jeong, "Restricted relevance vector machine for missing data and application to virtual metrology," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 4, pp. 3172–3183, Oct. 2022.
[2] W. Ding, W. Li, Z. Zhang, C. Wan, J. Duan, and S. Lu, "Time-varying Gaussian Markov random fields learning for multivariate time series clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 11, no. 35, pp. 1–17, Dec. 2022.
[3] S. Lee, J. Choi, and Y. Son, "Efficient visibility algorithm for high-frequency time-series: Application to fault diagnosis with graph convolutional network," *Ann. Oper. Res.*, pp. 1–21, Mar. 2023. [Online]. Available: https://doi.org/10.1007/s10479-022-05071-x
[4] T. Ching et al., "Opportunities and obstacles for deep learning in biology and medicine," *J. The Roy. Soc. Interface*, vol. 15, no. 141, 2018, Art. no. 20170387.
[5] J. Yu and Y. Son, "Weighted co-association rate-based Laplacian regularized label description for semi-supervised regression," *Inf. Sci.*, vol. 545, pp. 688–712, Feb. 2021.
[6] K. Wickstrøm, M. Kampffmeyer, K. Ø. Mikalsen, and R. Jenssen, "Mixing up contrastive learning: Self-supervised representation learning for time series," *Pattern Recognit. Lett.*, vol. 155, pp. 54–61, Mar. 2022.
[7] L. Jing and Y. Tian, "Self-supervised visual feature learning with deep neural networks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 11, pp. 4037–4058, Nov. 2021.
[8] Z. Cheng, Y. Yang, S. Jiang, W. Hu, Z. Ying, Z. Chai, and C. Wang, "Time2Graph+: Bridging time series and graph representation learning via multiple attentions," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 2, pp. 2078–2090, Feb. 2023.
[9] J. Zhang and K. Ma, "Rethinking the augmentation module in contrastive learning: Learning hierarchical augmentation invariance with expanded views," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 16629–16638.
[10] X. Xu, F. Zhou, K. Zhang, and S. Liu, "CCGL: Contrastive cascade graph learning," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 5, pp. 4539–4554, May 2023.
[11] Z. Yue, Y. Wang, J. Duan, T. Yang, C. Huang, Y. Tong, and B. Xu, "Ts2vec: Towards universal representation of time series," in *Proc. AAAI Conf. Artif. Intell.*, vol. 36, no. 8, 2022, pp. 8980–8987.
[12] X. Yang, Z. Zhang, and R. Cui, "TimeCLR: A self-supervised contrastive learning framework for univariate time series representation," *Knowl.-Based Syst.*, vol. 245, Jun. 2022, Art. no. 108606.
[13] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," 2017, *arXiv:1709.04875*.
[14] S. Guo, Y. Lin, H. Wan, X. Li, and G. Cong, "Learning dynamics and heterogeneity of spatial–temporal graph data for traffic forecasting," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 11, pp. 5415–5428, Nov. 2022.
[15] M. W. Kadous and C. Sammut, "Classification of multivariate time series and structured data using constructive induction," *Mach. Learn.*, vol. 58, nos. 2–3, pp. 179–216, Feb. 2005.
[16] A. Cini, I. Marisca, and C. Alippi, "Filling the G_ap_s: Multivariate time series imputation by graph neural networks," 2021, *arXiv:2108.00298*.
[17] Z. Gao, Z. Li, H. Zhang, J. Yu, and L. Xu, "Dynamic spatiotemporal interactive graph neural network for multivariate time series forecasting," *Knowl.-Based Syst.*, vol. 280, Nov. 2023, Art. no. 110995.
[18] Y. Zheng, H. Y. Koh, M. Jin, L. Chi, K. T. Phan, S. Pan, Y.-P. P. Chen, and W. Xiang, "Correlation-aware spatial–temporal graph learning for multivariate time-series anomaly detection," *IEEE Trans. Neural Netw. Learn. Syst.*, pp. 1–15, Nov. 2023. [Online]. Available: https://doi.org/10.1109/TNNLS.2023.3325667

[19] Y. Wang, Y. Xu, J. Yang, M. Wu, X. Li, L. Xie, and Z. Chen, "Graph-aware contrasting for multivariate time-series classification," 2023, *arXiv:2309.05202*.
[20] Y. Wang, M. Wu, X. Li, L. Xie, and Z. Chen, "Multivariate time series representation learning via hierarchical correlation pooling boosted graph neural network," *IEEE Trans. Artif. Intell.*, vol. 1, no. 5, pp. 1–13, Feb. 2023.
[21] A. Deng and B. Hooi, "Graph neural network-based anomaly detection in multivariate time series," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, Feb. 2021, pp. 4027–4035.
[22] S. Lee, H. Park, C. Choi, W. Kim, K. K. Kim, Y.-K. Han, J. Kang, C.-J. Kang, and Y. Son, "Multi-order graph attention network for water solubility prediction and interpretation," *Sci. Rep.*, vol. 13, no. 1, p. 957, Mar. 2023.
[23] J.-Y. Franceschi, A. Dieuleveut, and M. Jaggi, "Unsupervised scalable representation learning for multivariate time series," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–12.
[24] S. Tonekaboni, D. Eytan, and A. Goldenberg, "Unsupervised representation learning for time series with temporal neighborhood coding," 2021, *arXiv:2106.00750*.
[25] E. Eldele, M. Ragab, Z. Chen, M. Wu, C. K. Kwoh, X. Li, and C. Guan, "Time-series representation learning via temporal and contextual contrasting," 2021, *arXiv:2106.14112*.
[26] G. Zerveas, S. Jayaraman, D. Patel, A. Bhamidipaty, and C. Eickhoff, "A transformer-based framework for multivariate time series representation learning," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2021, pp. 2114–2124.
[27] D. Luo, W. Cheng, Y. Wang, D. Xu, J. Ni, W. Yu, X. Zhang, Y. Liu, Y. Chen, H. Chen, and X. Zhang, "Time series contrastive learning with information-aware augmentations," in *Proc. AAAI Conf. Artif. Intell.*, 2023, vol. 37, no. 4, pp. 4534–4542.
[28] X. Zheng, T. Wang, W. Cheng, A. Ma, H. Chen, M. Sha, and D. Luo, "Auto TCL: Automated time series contrastive learning with adaptive augmentations," in *Proc. 32nd Int. Joint Conf. Artif. Intell. (IJCAI)*, Aug. 2023, pp. 1–19.
[29] L. N. Ferreira and L. Zhao, "Time series clustering via community detection in networks," *Inf. Sci.*, vol. 326, pp. 227–242, Jan. 2016.
[30] S. Bai, J. Zico Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018, *arXiv:1803.01271*.
[31] D. Bo, X. Wang, C. Shi, M. Zhu, E. Lu, and P. Cui, "Structural deep clustering network," in *Proc. Web Conf.*, Apr. 2020, pp. 1400–1410.
[32] H. T. Pham and B.-S. Yang, "Estimation and forecasting of machine health condition using ARMA/GARCH model," *Mech. Syst. Signal Process.*, vol. 24, no. 2, pp. 546–558, Feb. 2010.
[33] M. A. Islam, S. Jia, and N. D. Bruce, "How much position information do convolutional neural networks encode?" in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–11.
[34] O. Semih Kayhan and J. C. van Gemert, "On translation invariance in CNNs: Convolutional layers can exploit absolute spatial location," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 14262–14273.
[35] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, no. 12, 2021, pp. 11106–11115.
[36] D. Hendrycks and K. Gimpel, "Gaussian error linear units (GELUs)," 2016, *arXiv:1606.08415*.
[37] Y. Chen, B. Hu, E. Keogh, and G. E. A. P. A. Batista, "DTW-D: Time series semi-supervised learning from a single example," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2013, pp. 383–391.
[38] W. J. Conover, *Practical Nonparametric Statistics*, vol. 350. Hoboken, NJ, USA: Wiley, 1999.
[39] D. Dua and C. Graff, *UCI Machine Learning Repository*. Irvine, CA, USA: Univ. of California, 2019. [Online]. Available: http://archive.ics.uci.edu/ml
[40] S. Li, X. Jin, Y. Xuan, X. Zhou, W. Chen, Y.-X. Wang, and X. Yan, "Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–11.
[41] G. Lai, W.-C. Chang, Y. Yang, and H. Liu, "Modeling long- and short-term temporal patterns with deep neural networks," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jun. 2018, pp. 95–104.
[42] D. Cao, Y. Wang, J. Duan, C. Zhang, X. Zhu, C. Huang, Y. Tong, B. Xu, J. Bai, J. Tong, and Q. Zhang, "Spectral temporal graph neural network for multivariate time-series forecasting," in *Proc. NIPS*, 2020, pp. 17766–17778.
[43] S. Li, L. Ge, Y. Lin, and B. Zeng, "Adaptive spatial–temporal fusion graph convolutional networks for traffic flow forecasting," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2022, pp. 4189–4196.
[44] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A time series is worth 64 words: Long-term forecasting with transformers," 2022, *arXiv:2211.14730*.
[45] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "TimesNet: Temporal 2D-variation modeling for general time series analysis," 2022, *arXiv:2210.02186*.
[46] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?" in *Proc. AAAI Conf. Artif. Intell.*, 2023, vol. 37, no. 9, pp. 11121–11128.
[47] L. McInnes, J. Healy, and J. Melville, "UMAP: Uniform manifold approximation and projection for dimension reduction," 2018, *arXiv:1802.03426*.

SANGHO LEE (Graduate Student Member, IEEE) was born in Seoul, Republic of Korea, in 1995. He received the B.S. and M.S. degrees in industrial and systems engineering from Dongguk University, Seoul, in 2018 and 2020, respectively, where he is currently pursuing the Ph.D. degree in industrial and systems engineering. His research interests include time-series analysis, machine learning, deep learning, and their applications to practical industrial problems.

WONJOON KIM (Member, IEEE) received the Ph.D. degree in industrial engineering from Seoul National University, in 2017. He is currently an Assistant Professor with the Division of Future Convergence (HCI Science Major), Dongduk Women's University. His research interests include human factor, artificial intelligence, deep learning, and the process of new product development with particular interests in product design and improvement.

YOUNGDOO SON (Member, IEEE) received the B.S. degree in physics and the M.S. degree in industrial and management engineering from Pohang University of Science and Technology (POSTECH), Gyeongbuk, Republic of Korea, in 2010 and 2012, respectively, and the Ph.D. degree in industrial engineering from Seoul National University, in 2015. He is currently an Associate Professor with the Department of Industrial and Systems Engineering, Dongguk University, Seoul, Republic of Korea. His research interests include machine learning, data analytics, and their applications to industrial problems.