SVGL: Scale-Variable Graph Learning in Model Space for Multivariate Time Series Classification

Shikang Liu, Ziyu Tang, Xiren Zhou*, Huanhuan Chen*

School of Computer Science and Technology, University of Science and Technology of China
skliu00@mail.ustc.edu.cn, ziyutang@mail.ustc.edu.cn, zhou0612@ustc.edu.cn, hchen@ustc.edu.cn

Abstract

Multivariate time series classification (MTSC) has broad applications in numerous domains. Existing MTSC methods typically focus on either temporal dynamics or variable interactions of the data, often overlooking cross-scale couplings among different variables. To bridge this gap, we propose Scale-Variable Graph Learning (SVGL), a novel framework that effectively captures data-inherent scale-variable interactions for MTSC. SVGL begins with spectral analysis to adaptively identify key periodic scales for each variable. A period-aware reservoir computing network is then incorporated to fit the variable at these scales, encoding the sequential and periodic dynamics into multi-scale dynamic representations. Subsequently, we construct a scale-variable graph to model interactions of the encoded temporal dynamics, where nodes represent scale-variable pairs and edges denote their correlations. After sparsely initializing the graph via nearest neighbors, a parallel graph learning architecture is integrated in SVGL, combining global graph convolutional and sample-specific graph attention to aggregate effective features for classification. Extensive experiments on 30 UEA datasets demonstrate that SVGL outperforms state-of-the-art baselines in accuracy and maintains low training overhead.

Introduction

Multivariate time series classification (MTSC) plays a critical role in widespread applications, including human activity recognition (Yang et al. 2015), medical monitoring (Song et al. 2018), financial analysis (Majumdar and Laha 2020), and industrial fault diagnosis (Cheng et al. 2023). These diverse scenarios have attracted growing interest from the machine learning community and inspired a wealth of methods to improve classification accuracy and interpretability.

Existing MTSC approaches generally pursue two complementary directions: capturing temporal dynamics and modeling variable interactions. On the temporal side, Convolutional Neural Networks (CNNs) (Tang et al. 2022; Yue et al. 2022) leverage local receptive fields to extract shape patterns, but they struggle to capture global dependencies. Recurrent Neural Networks (RNNs) (Karim et al. 2019; Zhang et al. 2020) propagate information sequentially through hidden state recursion, yet suffer from memory decay due to vanishing gradients. Transformers (Zhou et al. 2022; Zhang and Yan 2023) employ self-attention to encode long-range dependencies. However, they often miss fine-grained local patterns at adjacent time steps. Further advances turn to identifying key periodicities for multi-scale analysis. Some works reshape time series into a 2D format aligned with its dominant period, thereby extracting intra- and inter-period features (Wu et al. 2023). Despite the progress, these methods largely ignore cross-variable influences, limiting their effectiveness and interpretability (Tian et al. 2025).

On the variable side, Graph Neural Networks (GNNs) have risen to explicitly model variable interactions (Jin et al. 2024). In particular, Graph Convolutional Networks (Kipf and Welling 2017; Jia et al. 2020; Liu et al. 2024) rely on static or learnable graph structures to aggregate multi-hop information; while Graph Attention Networks (Deng and Hooi 2021; Zhang et al. 2022) generate adaptive edge weights based on dynamic similarities between variables. Nevertheless, most GNN-based approaches emphasize variable dependencies in isolation and lack an essential multi-scale perspective for temporal dynamics. As a result, such models could miss crucial patterns whose expression requires synergy across scales and variables.

Recent studies (Cai et al. 2024; Mu, Shahzad, and Zhu 2025) have explored bridging these two directions for enhanced performance. They primarily merge all variables' energy spectra to extract unified key scales, followed by modeling variable interactions separately at each scale. However, these methods still overlook a critical insight: *in real-world scenarios, variables may display explicit or implicit cross-scale couplings when observed at their key scales.* Simply put, the dynamics of one variable at a given scale could be influenced by another variable's behavior at a different scale. Hereafter, we refer to this phenomenon as "scale-variable interactions". A classic example is the butterfly effect, where a minute local disturbance propagates, amplifies, and causes large-scale systemic change. Therefore, unified scale identification and scale-independent analysis remain inadequate for capturing and exploiting such complex scale-variable interactions, leading to suboptimal feature extraction.

To address these limitations, we propose a Scale-Variable Graph Learning (SVGL) framework, as shown in Figure 1. SVGL adaptively extracts multi-scale dynamic representations for each variable and explicitly models scale-variable

*Corresponding Authors: Xiren Zhou, Huanhuan Chen.

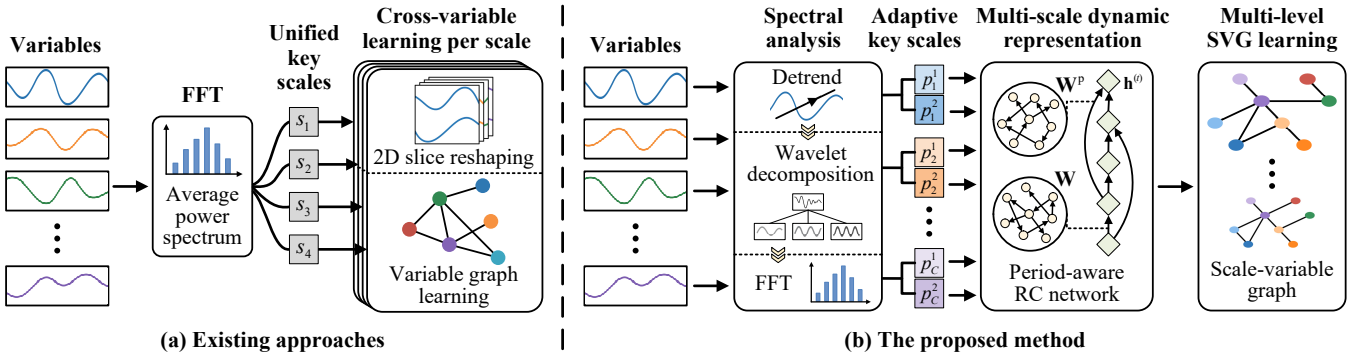

Figure 1: Comparison of existing and proposed methods for bridging multi-scale temporal dynamics and variable interactions. Existing methods extract unified key scales for all variables and perform cross-variable learning separately at each scale. In contrast, we identify adaptive key scales per variable, derive dynamic representations at those scales, followed by learning global and sample-specific SVGs to capture scale-variable interactions.

interactions via graph structures. The representation module follows the Model Space Learning (MSL) paradigm (Chen et al. 2014), that is, fitting time series with a generative model to capture data-inherent temporal dynamics and viewing the fitted model as a compact dynamic representation. Previous studies on MSL (Bianchi et al. 2021; Zhou et al. 2023, 2024; Liu, Zhou, and Chen 2025) have demonstrated that Reservoir Computing (RC), a training-free variant of RNNs, captures temporal dynamics efficiently and accurately. Inspired by this, SVGL first identifies key periodic scales per variable using spectral analysis, then for each scale incorporates a dedicated period-aware RC network (Liu et al. 2025) to fit the variable. This network features two reservoirs in its hidden layer, one controlling state transitions between adjacent time steps, and the other between periods. Consequently, the network's fitted readout model effectively captures sequential and periodic dynamics of the variable, serving as its dynamic representation at that scale.

Afterward, we construct a Scale-Variable Graph (SVG) to describe interactions of the captured temporal dynamics across scales and variables. In SVG, each node corresponds to a scale-variable pair, its feature encodes the learned dynamic representation, and edges denote their correlations. To avoid quadratic complexity, the SVG structure is sparsely initialized through nearest neighbor analysis. Building on this, we design a parallel graph learning architecture to capture and exploit multi-level scale-variable interactions. Specifically, graph convolutional layers extract global node relationships at the dataset level, while graph attention layers adaptively learn sample-specific edge weights. Combining the obtained features as input, a final fully connected layer enables effective classification.

Our main contributions are summarized as follows:

- We make a critical observation that cross-scale influences and couplings exist among variables. To address this, we propose a novel SVGL framework that effectively captures such scale-variable interactions for MTSC.
- The period-aware RC network is incorporated to fit each variable at key scales, encoding the sequential and periodic dynamics into multi-scale dynamic representations.
- A parallel graph learning architecture is further incorporated, combining global graph convolution and sample-specific graph attention to aggregate features along scale–variable interactions.
- Extensive experiments on 30 UEA datasets ensure a comprehensive evaluation, demonstrating that SVGL outperforms state-of-the-art MTSC methods.

Related Work

Multivariate Time Series Classification

Multivariate Time Series Classification aims to predict categorical labels from multivariate sequences observed over time. Recent DL alternatives in this field can be roughly divided into several types: CNN-based methods, such as OS-CNN (Tang et al. 2022), TimesNet (Wu et al. 2023), and ModernTCN (Luo and Wang 2024), leverage convolutional kernels of unified or varying sizes to capture intra-sequence and inter-variable dependencies. RNN/CNN Hybrid architectures, notably MLSTM-FCN (Karim et al. 2019) and Tapnet (Zhang et al. 2020), exploit recurrent structures to capture temporal dependencies combined with CNNs for local feature extraction. Transformer-based methods, including FEDformer (Zhou et al. 2022) and Crossformer (Zhang and Yan 2023), have emerged for *General Time Series Analysis*. They effectively utilize multi-scale behaviors of time series and capture long-term dependencies through the self-attention mechanism. Furthermore, MLP-based models (Zeng et al. 2023; Wang et al. 2024) have shown scalability benefits. However, these methods either overlook inherent interactions across variables or fail to explicitly model the complex inter-variable relationships, potentially limiting their representational capacity and interpretability.

Addressing these limitations, graph neural networks enable the modeling of variable interactions via an explicit graph structure. Recent GNN-based approaches span learning static variable graphs, such as SimTSC (Zha et al. 2022), to employing dynamic or spatiotemporal graphs, like TodyNet (Liu et al. 2024) and GraphWaveNet (Wu et al. 2019).

Some methods extend graph learning by introducing hypergraph structures (Younis and Ahmadi 2024; Tian et al. 2025), which capture higher-order interactions between multiple variables simultaneously. Others consider relationships at multiple time scales, exemplified by Time2Graph (Cheng et al. 2020), which explicitly models temporal dynamics between shapelets. Aside from graph learning, MPTSNet (Mu, Shahzad, and Zhu 2025) captures variable dependencies across diverse periodic scales. Nonetheless, existing methods often limit variable interaction modeling to single or unified time scales, lacking adaptive identification of key scales for individual variables. The neglect of cross-scale interactions further impedes their ability to fully exploit hierarchical temporal dynamics within multivariate time series.

Model Space Learning for Time Series Analysis

MSL fits time series with a well-designed generative model that captures its intrinsic temporal dynamics. The fitted model thus serves as a compact dynamic representation of the original series for downstream analysis in the model space. Early works adopted simple models like ARIMA (Xiong and Yeung 2002) or HMMs (Srivastava et al. 2007), which primarily capture linear temporal dependencies. Subsequently, reservoir computing, notably Echo State Networks (ESNs) (Jaeger 2001), emerged as an effective nonlinear alternative, providing more accurate dynamic representations without extensive training overhead (Chen et al. 2014). Further developments extended MSL by incorporating multi-dimensional reservoir models, such as CubeRes (Zhou et al. 2024), to simultaneously capture spatiotemporal dependencies across multiple directions. Moreover, Chen et al. (2025) employed an ESN-based ordinary differential equation to model irregular sequences in continuous-time model space. However, most MSL methods overlook multi-scale characteristics inherent in time series. Despite recent efforts to incorporate a spectral perspective for multi-scale analysis (Liu et al. 2025), they remain unable to capture inter-variable relationships, let alone the interactions of temporal dynamics across different scales and variables.

Methodology

This section proposes the SVGL framework, which consists of two main stages:

- **Multi-scale Dynamic Representation.** Key periodic scales for each variable are identified, followed by fitting with the period-aware RC network at each scale. The fitted readout model captures sequential and periodic dynamics of the variable, serving as its scale-specific dynamic representation in the model space.
- **Multi-level SVG Structure Learning.** Nearest neighbor analysis constructs an initial sparse SVG between dynamic representations in the model space. Graph convolution and attention layers then learn both global and sample-specific graph structures, producing effective combined features for classification.

These are depicted in Figure 1 and further described below.

Multi-scale Dynamic Representation

Identifying Key Scales for Each Variable To capture multi-scale temporal dynamics of each variable, it is crucial to identify key scales at which the variable exhibits significant periodic patterns. Most related work applies a straightforward Fast Fourier Transform (FFT) to select frequencies and periods with the highest amplitudes (Wu et al. 2023; Mu, Shahzad, and Zhu 2025). However, this approach suffers from low-frequency drift caused by aperiodic trends and neglects smaller scales. Thus, we adopt a more crafted pipeline (Liu et al. 2025), integrating detrending, wavelet decomposition, and FFT to identify key scales for each variable.

Given a time series $\mathbf{X} \in \mathbb{R}^{T \times C}$ of length T with C variables, let $\mathbf{x}_i = \{\mathbf{x}_i^{(t)}\}_{t=1}^{T}$ denote the time series of i-th variable. We begin by applying the Hodrick-Prescott filter (Hodrick and Prescott 1997) to each variable, which efficiently estimates the long-term trend component $\hat{\boldsymbol{\tau}}_i$:

$$\hat{\boldsymbol{\tau}}_i = \text{HPFilter}(\mathbf{x}_i), \quad \mathbf{r}_i = \mathbf{x}_i - \hat{\boldsymbol{\tau}}_i. \tag{1}$$

Here, $\mathbf{r}_i$ is the detrended residual of i-th variable, representing a mixture of multiple periodic patterns.

To decouple the mixed periodic patterns, wavelet decomposition is then employed to separate the residual $\mathbf{r}_i$ into different resolutions. This yields a series of periodic components, each corresponding to a distinct scale:

$$\mathbf{r}_i^s \;=\; \mathcal{W}^{-1}\big[(\mathcal{W}(\mathbf{r}_i))_s\big], \quad s = 0, \ldots, S, \tag{2}$$

where $\mathcal{W}(\cdot)$ denotes the wavelet transform operator that produces an approximation $\mathbf{a}_i^0$ and detail coefficients $\mathbf{d}_i^s$ at resolutions $s \in [1, S]$; $(\cdot)_s$ extracts coefficients at s-th resolution; and $\mathcal{W}^{-1}(\cdot)$ reconstructs the periodic component $\mathbf{r}_i^s$.

After that, each periodic component is analyzed using FFT to identify the prominent frequency and its corresponding periodic scale:

$$f_i^s = \arg\max_f(|\text{FFT}(\mathbf{r}_i^s)|), \quad p_i^s = \frac{T}{f_i^s} \tag{3}$$

where $|\text{FFT}(\cdot)|$ calculates the magnitude of FFT results; the frequency f_i^s is selected based on the maximum magnitude; and p_i^s indicates that the i-th variable $\mathbf{x}_i$ exhibits a prominent periodic pattern at that scale.

This pipeline results in a series of key periodic scales for each variable $\mathbf{x}_i$. Detrending prevents spectral energy from concentrating at zero frequency, while wavelet decomposition ensures that the identified scales cover both low- and high-frequency domains. Collectively, we obtain a set of scale-variable pairs for $\mathbf{X}$:

$$\mathcal{P}(\mathbf{X}) = \{(p_i^s, \mathbf{x}_i) \mid 1 \le i \le C, 0 \le s \le S\}, \tag{4}$$

where each pair indicates the periodic dynamics for a specific variable at a particular scale that needs to be captured. Subsequently, we apply the period-aware RC network to each pair, capturing such dynamics and deriving a respective dynamic representation in the model space.

Fitting with Period-Aware RC Network For each pair $(p, \mathbf{x}) \in \mathcal{P}(\mathbf{X})$, we incorporate a period-aware RC network (Liu et al. 2025) to fit $\mathbf{x}$ bidirectionally and capture temporal

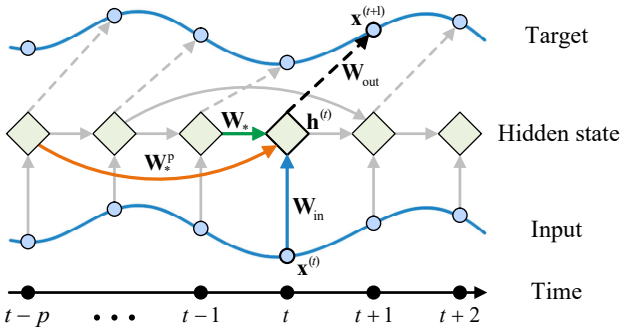

Figure 2: Iteration of the period-aware RC network in one direction. Hidden state update is influenced by adjacent time steps and periods through independent reservoirs $\mathbf{W}_*$ and $\mathbf{W}_*^{\text{p}}$, capturing both sequential and periodic dynamics. The hidden state is then used to approximate next step input.

dynamics at scale p[1]. As illustrated in Figure 2, the network features dual time-lag dependencies in each direction. Starting from an initial zero state, it sequentially reads the input, passes through a linear input layer, and then into a hidden layer consisting of two reservoirs. These reservoirs contain randomly interconnected neurons, describing the influence between adjacent states and states with a span of p. Specifically, the forward and backward iterations are as follows:

$$\begin{aligned}\mathbf{h}_{\text{f}}^{(t)} &= \tanh\left(\mathbf{W}_{\text{f}}\mathbf{h}_{\text{f}}^{(t-1)} + \mathbf{W}_{\text{f}}^{\text{p}}\mathbf{h}_{\text{f}}^{(t-p)} + \mathbf{W}_{\text{in}}\mathbf{x}^{(t)}\right),\\ \mathbf{h}_{\text{b}}^{(t)} &= \tanh\left(\mathbf{W}_{\text{b}}\mathbf{h}_{\text{b}}^{(t+1)} + \mathbf{W}_{\text{b}}^{\text{p}}\mathbf{h}_{\text{b}}^{(t+p)} + \mathbf{W}_{\text{in}}\mathbf{x}^{(t)}\right),\end{aligned} \tag{5}$$

where subscripts "f" and "b" indicate the iteration direction; $\mathbf{W}_*$ and $\mathbf{W}_*^{\text{p}}$ are reservoir weights adjusting state transitions between adjacent time steps and periods, respectively; and $\mathbf{W}_{\text{in}}$ denote input weights. Following the convention in RC, all these weights are randomly initialized and untrainable.

Upon iteration, the final hidden state $\mathbf{h}_{\text{bi}}$ is obtained by concatenating $\mathbf{h}_{\text{f}}$ and $\mathbf{h}_{\text{b}}$. The output layer then uses a linear readout model to map it to the output value $\mathbf{y}$:

$$\mathbf{y}^{(t)} = \mathbf{G}(\mathbf{h}_{\text{bi}}^{(t)}) = \mathbf{W}_{\text{out}}\mathbf{h}_{\text{bi}}^{(t)}. \tag{6}$$

Here, $\mathbf{G}$ denotes the readout model, and $\mathbf{W}_{\text{out}}$ is its parameter, typically determined based on the target output.

Using hidden states that encapsulate temporal dependencies of span p, we apply a "next-step prediction task" to derive the readout model. In this task, the next input value serves as the target, requiring the output layer to predict future observations based on historical information. This allows readout model to fit the recursive relationships within time series $\mathbf{x}$. As a result, the model $\mathbf{G} = \mathbf{W}_{\text{out}}\mathbf{h}$ effectively captures periodic dynamics at the specific scale, thus serving as a dynamic representation of the variable in model space. Concretely, this task can be solved via ridge regression:

$$\mathbf{W}_{\text{out}} = \hat{\mathbf{x}}\mathbf{H}^\top(\mathbf{H}\mathbf{H}^\top + \beta\mathbf{I})^{-1}, \tag{7}$$

where $\hat{\mathbf{x}} = [\mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \cdots, \mathbf{x}^{(T)}]$ is the target vector; $\mathbf{H} = [\mathbf{h}_{\text{bi}}^{(1)}, \cdots, \mathbf{h}_{\text{bi}}^{(T-1)}]$ is the state matrix; β is a regularization coefficient; and $\mathbf{I}$ is the identity matrix.

[1]For generality, we omit subscripts and superscripts of p and $\mathbf{x}$.

By deriving the readout model $\mathbf{G}_i^s$ for each pair in $\mathcal{P}(\mathbf{X})$, we obtain multiple dynamic representations of each variable, which capture temporal dynamics at their key periodic scales. These readout models collectively form a model space $\{\mathbf{G}_i^s\}_{i=1,s=0}^{C,S}$. In the following, we treat each scale-variable pair as a graph node, learning interactions between scales and variables in the model space for classification.

Multi-level SVG Structure Learning

This section analyzes the interactions of temporal dynamics across scales and variables through a Scale-Variable Graph. After formally defining SVG, we construct an initial sparse graph via nearest neighbor analysis, which markedly reduces computational complexity. Graph convolutional and attention layers are then integrated to learn global and sample-specific SVG structures respectively (Figure 3), generating combined features for effective MTSC.

Definition 1 *A **Scale-Variable Graph** $\mathcal{G}$ is defined as a triple $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$, where $\mathcal{V} = \mathcal{P}(\mathbf{X})$ is the set of nodes, i.e., scale-variable pairs; $\mathcal{E}$ is the set of edges between nodes; and $\mathcal{F}$ is the feature set.*

Each node $\mathbf{v}_i^s := (p_i^s, \mathbf{x}_i)$ is associated with a feature $\mathbf{F}_i^s$ containing the corresponding dynamic representation's weights $\mathbf{W}_{out,i}^s$. The maxpooling of hidden states is also incorporated to enhance features' representational capability:

$$\mathbf{F}_i^s = \text{MaxPool}\left(\mathbf{h}_{\text{bi},i}^s\right) \oplus \mathbf{W}_{\text{out},i}^s. \tag{8}$$

Initial Graph Construction Given that most scenarios lack prior knowledge, assuming a complete SVG is the natural way to avoid information loss. However, this strategy leads to quadratic complexity in the number of variables and can impose a heavy computational burden. To reduce this cost while retaining informative edges, we construct an initial sparse SVG via nearest neighbors.

Specifically, we compute pairwise similarities based on feature vectors of different nodes. The similarity between nodes $\mathbf{v}_i^s$ and $\mathbf{v}_j^r$ is measured using Euclidean distance:

$$\text{d}(\mathbf{v}_i^s, \mathbf{v}_j^r) = \|\mathbf{F}_i^s - \mathbf{F}_j^r\|_2. \tag{9}$$

In general, a higher similarity between two nodes implies closer feature proximity, signaling a greater likelihood that they exhibit similar temporal dynamics; thus, these nodes can be considered closely related. Based on a preset hyperparameter k, we select the k nearest neighbors for each node and add the corresponding directed edges, yielding an edge set of size $k \times |\mathcal{V}|$ that constitutes the initial graph.

Global Graph Structure Learning To capture the dataset-level interactions across scales and variables, we employ graph convolutional layers to learn a global adjacency based on a shared embedding. Let $\mathbf{E} \in \mathbb{R}^{|\mathcal{V}|\times d}$ be a trainable embedding for all nodes in the initial SVG. The normalized global adjacency $\mathbf{A}_{\text{g}}$ is derived as:

$$\begin{aligned}\hat{\mathbf{A}}_{\text{g}} &= \sigma\left(\mathbf{E}\mathbf{E}^\top\right) \odot \mathbf{A}_0 + \mathbf{I},\\ \mathbf{A}_{\text{g}} &= \hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{A}}_{\text{g}}\hat{\mathbf{D}}^{-\frac{1}{2}}.\end{aligned} \tag{10}$$

Here, σ is the LeakyReLU activation; $\mathbf{A}_0 \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{V}|}$ is the binary mask of initial edges; $\odot$ denotes elementwise

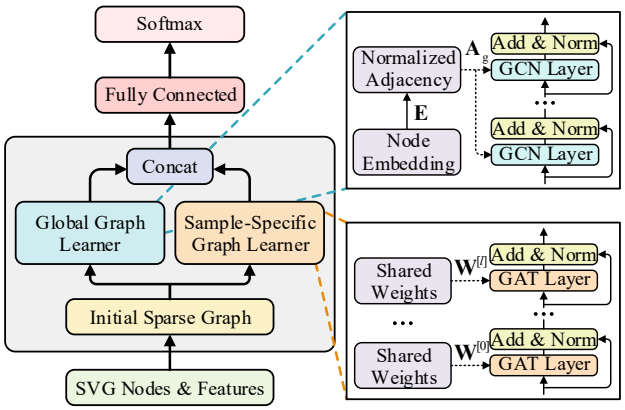

Figure 3: Illustration of multi-level SVG structure learning.

multiplication; and $\hat{\mathbf{D}}$ is the degree matrix of unnormalized adjacency $\hat{\mathbf{A}}_{\mathrm{g}}$. This formulation ensures that reciprocal edges in the initial SVG receive identical weights, while its low-rank factorization maintains computational efficiency.

Based on $\mathbf{A}_{\mathrm{g}}$, we stack multiple graph convolutional layers to propagate global structural information through SVG nodes and extract high-level features. The propagation rule at layer l is as follows:

$$\mathbf{F}_{\mathrm{g}}^{[l+1]} = \mathrm{BN}\Big(\sigma\big(\mathbf{A}_{\mathrm{g}}\mathbf{F}_{\mathrm{g}}^{[l]}\mathbf{W}_{\mathrm{g}}^{[l]}\big) + \mathbf{F}_{\mathrm{g}}^{[l]}\Big), \tag{11}$$

where $\mathbf{F}_{\mathrm{g}}^{[l]}$ denotes the node feature matrix at layer l; $\mathbf{F}_{\mathrm{g}}^{[0]}$ is formed by stacking initial features of all nodes; and $\mathbf{W}_{\mathrm{g}}^{[l]}$ is the trainable weight matrix. Residual connections and batch normalization are integrated to improve gradient flow and training stability. This design effectively captures holistic scale-variable interactions, with its output $\mathbf{F}_{\mathrm{GC}}$ aggregating multi-hop information among the multi-scale dynamic representations in the model space.

Sample-specific Graph Structure Learning Although the above component captures global node relationships across the entire dataset, individual samples may exhibit unique scale-variable interactions crucial for accurate classification. To address this, graph attention layers are incorporated to compute edge weights adaptively per sample.

Given the input feature matrix $\mathbf{F}_{\mathrm{a}}$, where each row represents a node feature, the attention score between the connected m-th and n-th nodes is derived as:

$$\alpha_{mn} = \sigma\Big(\mathbf{a}^{\top}\big[\mathbf{F}_{\mathrm{a}}^{(m)}\mathbf{W} \,\|\, \mathbf{F}_{\mathrm{a}}^{(n)}\mathbf{W}\big]\Big), \tag{12}$$

where $\mathbf{W}$ is a shared linear projection; $\mathbf{a}$ is a learnable vector; and $\|$ denotes concatenation, same as $\oplus$. Edge weights are then normalized using softmax on the neighbourhood, constituting a sample-specific adjacency $\mathbf{A}_{\mathrm{att}}$ that assigns higher weights to node pairs with stronger interactions.

Subsequently, in the l-th layer, node feature matrix $\mathbf{F}_{\mathrm{a}}^{[l]}$ is updated based on these edge weights:

$$\mathbf{F}_{\mathrm{a}}^{[l+1]} = \mathrm{BN}\Big(\sigma\big(\mathbf{A}_{\mathrm{att}}^{[l]}\mathbf{F}_{\mathrm{a}}^{[l]}\mathbf{W}^{[l]}\big) + \mathbf{F}_{\mathrm{a}}^{[l]}\Big), \tag{13}$$

with residual connections and batch normalization similarly applied. Here, $\mathbf{W}^{[l]}$ is trainable; and $\mathbf{F}_{\mathrm{a}}^{[0]}$ is initialized the same as $\mathbf{F}_{\mathrm{g}}^{[0]}$. This component adaptively emphasizes important node relationships for each sample, enabling the output $\mathbf{F}_{\mathrm{GA}}$ to reflect sample-specific interactions of dynamic representations across scales and variables.

Classification and Training After learning both global and sample-specific SVG structures, two sets of node features, $\mathbf{F}_{\mathrm{GC}}$ and $\mathbf{F}_{\mathrm{GA}}$, are obtained. To fuse global and sample-specific information, we apply maxpooling and concatenate them into the final features $\mathbf{Z}$. A fully connected layer is then utilized for classification:

$$\hat{\mathbf{Y}} = \mathrm{Softmax}\big(\mathbf{W}_{\mathrm{c}}\,\mathbf{Z} + \mathbf{b}_{\mathrm{c}}\big), \tag{14}$$

where $\mathbf{W}_{\mathrm{c}}$ and $\mathbf{b}_{\mathrm{c}}$ are the trainable weights and bias. Training is performed on the learned multi-scale dynamic representations by minimizing the cross-entropy loss between output $\hat{\mathbf{Y}}$ and the ground-truth.

Experiments

This section evaluates SVGL on public benchmarks. All experiments run on a desktop with AMD Ryzen 9 9950X CPU, 32 GB RAM, and NVIDIA GeForce RTX 4090 GPU.

Experimental Settings

Datasets We employ the widely adopted UEA Archive (Bagnall et al. 2018), which contains 33 datasets spanning EEG, motion, audio, and other domains. These datasets exhibit substantial diversity in length, number of variables, class count, and training size, enabling a comprehensive evaluation of the proposed method.

In the main paper, we report on 30 of these datasets and analyze the results. The remaining three, namely *AsphaltRegularityCoordinates*, *EigenWorms*, and *InsectWingbeat*, are excluded because some baselines fail to complete execution due to memory or computational limitations.

Competing Methods Following established practices (Luo and Wang 2024; Wu et al. 2023), we first compare the proposed SVGL with 12 state-of-the-art General Time Series Analysis methods on 10 representative UEA datasets. These methods include **LSTNet** (Lai et al. 2018), **TS2Vec** (Yue et al. 2022), **LSSL** (Gu, Goel, and Ré 2022), **FEDformer** (Zhou et al. 2022), **Flowformer** (Wu et al. 2022), **SCINet** (Liu et al. 2022), **DLinear** (Zeng et al. 2023), **PatchTST** (Nie et al. 2023), **MICN** (Wang et al. 2023), **TimesNet** (Wu et al. 2023), **Crossformer** (Zhang and Yan 2023), and **ModernTCN** (Luo and Wang 2024).

Following (Chen et al. 2024; Mu, Shahzad, and Zhu 2025), we further compare SVGL with 3 baselines and 8 advanced MTSC-specific approaches on the 30 UEA datasets. These comprise **EDI** (ED-1NN), **DTWI** (DTW-1NN-I), **DTWD** (DTW-1NN-D), **WEASEL+MUSE** (Schäfer and Leser 2017), **MLSTM-FCN** (Karim et al. 2019), **TapNet** (Zhang et al. 2020), **OS-CNN**, **MOS-CNN** (Tang et al. 2022), **TodyNet** (Liu et al. 2024), **MPTSNet** (Mu, Shahzad, and Zhu 2025), and **TimeMIL** (Chen et al. 2024).

Dataset	LSTNet *SIG.'18*	TS2Vec *AAAI'22*	LSSL *ICLR'22*	FEDf. *ICML'22*	Flowf. *ICML'22*	SCINet *NIPS'22*	Dlinear *AAAI'23*	PatchTST *ICLR'23*	MICN *ICLR'23*	TimesNet *ICLR'23*	Crossf. *ICLR'23*	M.TCN *ICLR'24*	SVGL *Ours*
EthanolConcentration	39.9	30.8	31.1	31.2	33.8	34.4	36.2	32.8	35.3	35.7	38.0	36.3	**44.1**
FaceDetection	65.7	50.1	66.7	66.0	67.6	68.9	68.0	68.3	65.2	68.6	68.7	**70.8**	68.7
Handwriting	25.8	**51.5**	24.6	28.0	33.8	23.6	27.0	29.6	25.5	32.1	28.8	30.6	35.5
Heartbeat	77.1	68.3	72.7	73.7	77.6	77.5	75.1	74.9	74.7	78.0	77.6	77.2	**78.2**
JapaneseVowels	98.1	98.4	98.4	98.4	98.9	96.0	96.2	97.5	94.6	98.4	**99.1**	98.8	95.4
PEMS-SF	86.7	68.2	86.1	80.9	86.0	83.8	75.1	89.3	85.5	89.6	85.9	89.1	**90.3**
SelfRegulationSCP1	84.0	81.2	90.8	88.7	92.5	92.5	87.3	90.7	86.0	91.8	92.1	**93.4**	92.4
SelfRegulationSCP2	52.8	57.8	52.2	54.4	56.1	57.2	50.5	57.8	53.6	57.2	58.3	60.3	**63.9**
SpokenArabicDigits	**100**	98.8	**100**	**100**	98.8	98.1	81.4	98.3	97.1	99.0	97.9	98.7	98.7
UWaveGestureLibrary	87.8	90.6	85.9	85.3	86.6	85.1	82.1	85.8	82.8	85.3	85.3	86.7	**92.8**
Ours 1-to-1 Wins	8	7	8	8	7	7	9	9	10	8	8	6	-
Ours 1-to-1 Ties	0	0	0	0	0	0	0	0	0	0	1	1	-
Ours 1-to-1 Losses	2	3	2	2	3	3	1	1	0	2	1	3	-
Average Accuracy (↑)	71.8	69.6	70.9	70.7	73.2	71.7	67.9	72.5	70.0	73.6	73.2	74.2	**76.0**
Average Rank (↓)	7.10	8.35	8.35	8.55	5.35	7.90	10.00	7.15	10.70	5.00	5.40	3.75	**3.40**
P-Value ($\alpha = 0.05$)	2.0E-02	1.3E-01	2.0E-02	1.4E-02	6.4E-02	3.7E-02	5.9E-03	2.0E-02	2.0E-03	6.4E-02	2.8E-02	1.5E-01	-

Table 1: Accuracy comparison with state-of-the-art *General Time Series Analysis* methods on 10 UEA datasets. **Bold** indicates the best result and underline denotes the second-best. P-values are from Wilcoxon signed-rank tests. % in accuracies are omitted.

Dataset	EDI	DTWI	DTWD	W.+MUSE *arxiv'17*	M.-FCN *Neur.'19*	TapNet *AAAI'20*	OS-CNN *ICLR'22*	MOS-CNN *ICLR'22*	TodyNet *Info.'24*	MPTSNet *AAAI'25*	TimeMIL *ICML'24*	SVGL *Ours*
Cricket	94.4	98.6	**100**	**100**	98.6	95.3	97.2	98.9	90.3	94.2	99.7	**100**
ERing	13.3	13.3	13.3	**95.2**	87.4	87.8	86.7	90.7	84.8	94.4	94.2	**95.2**
FingerMovements	55.0	52.0	53.0	56.0	57.0	53.0	56.0	56.6	58.0	**63.4**	58.0	60.0
Heartbeat	62.0	65.9	71.7	72.2	68.4	75.6	72.8	71.8	75.6	76.1	75.1	**78.2**
LSST	45.6	57.5	55.1	60.5	60.4	55.0	60.6	55.6	59.6	60.2	57.8	**60.9**
NATOPS	86.0	85.0	88.3	93.3	90.0	92.7	93.9	94.3	92.2	93.7	92.0	**94.4**
PenDigits	97.3	93.9	97.7	94.7	98.3	97.9	98.7	98.2	98.4	**98.9**	97.6	97.2
RacketSports	86.8	84.2	80.3	86.8	87.5	86.5	83.5	**92.2**	86.2	87.0	86.2	88.8
UWaveGestureLibrary	88.1	86.9	90.3	90.6	89.1	88.9	**93.4**	91.9	82.8	88.0	89.3	92.8
Ours 1-to-1 Wins	28	25	24	20	23	26	19	20	21	20	19	-
Ours 1-to-1 Ties	0	1	1	3	2	2	2	0	2	1	2	-
Ours 1-to-1 Losses	2	4	5	7	5	2	9	10	7	9	9	-
Average Accuracy (↑)	59.0	66.1	65.3	72.9	70.1	70.2	72.1	73.8	70.3	73.5	75.0	**76.2**
Average Rank (↓)	10.30	8.75	8.12	5.25	6.93	7.43	5.53	5.00	6.52	5.53	5.07	**3.57**
P-Value ($\alpha = 0.05$)	1.3E-08	8.7E-05	2.7E-04	1.3E-02	6.9E-04	2.1E-05	4.7E-03	2.8E-02	2.2E-03	1.4E-01	9.6E-02	-

Table 2: Accuracy comparison with advanced *MTSC-specific* methods. Statistical metrics are drawn from 30 UEA datasets, while detailed accuracies appear only for a subset due to space constraints.

Implementation Details We use default train–test splits from the UEA archive. All datasets are standardized to maintain uniformity, and 10% of the training set is reserved for validation. We repeat each experiment five times and report the mean results. For competing methods, results are sourced from their original publications or official repositories. Refer to the Appendix for more details.

Performance Evaluation

According to Table 1, the proposed SVGL surpasses current general time series analysis methods across multiple classification metrics. Specifically, SVGL achieves the highest average accuracy of 76.0% and the lowest average rank of 3.40. Moreover, it consistently wins more in one-to-one comparisons. Compared with strong baselines such as ModernTCN, SVGL demonstrates superior performance on EthanolConcentration and UWaveGestureLibrary, with gains of 7.8% and 6.1%, respectively. Statistical tests also validate our significant improvement over 8 baselines.

From Table 2, SVGL outperforms state-of-the-art MTSC-specific methods, attaining the highest average accuracy of 76.2% and the best average rank of 3.57 across 30 UEA datasets. In pairwise comparisons, SVGL exhibits statistically significant advantages over most baselines and wins on the majority of datasets. Notably, when compared with MPTSNet, which learns cross-variable relationships separately at each key scale, our approach yields a 2.7% increase in average accuracy and performs better on 20 datasets. This stems from our capturing of cross-scale couplings and scale-variable interactions that effectively improve MTSC.

Efficiency Evaluation

The computational cost of SVGL mainly lies in fitting each variable to derive dynamic representations and the subsequent graph learning process. Given the sample size N, series length T, and variable count C, the former involves only a single forward pass with a time complexity of $O(NTC)$. The latter trains a neural network using node features as in-

Method	Train Time		Test Time		FLOPs		Params	
	μ_g	$Q_{\frac{1}{2}}$	μ_g	$Q_{\frac{1}{2}}$	μ_g	$Q_{\frac{1}{2}}$	μ_g	$Q_{\frac{1}{2}}$
M.-FCN	0.80	0.73	0.14	0.11	0.11	0.11	0.32	0.28
TapNet	**0.23**	**0.21**	**0.09**	**0.07**	0.18	0.18	0.97	0.86
OS-CNN	0.42	0.33	0.12	0.10	0.25	0.24	0.62	0.58
MOS-CNN	0.94	0.82	0.31	0.31	0.79	0.65	1.94	1.16
TodyNet	3.20	3.28	0.53	0.49	0.80	0.47	0.31	**0.23**
MPTSNet	15.56	11.19	6.22	4.18	26.88	23.78	64.03	41.10
TimeMIL	13.61	14.29	12.00	14.18	0.71	0.54	0.76	0.74
SVGL	0.42	0.25	1.13	0.96	**0.05**	**0.03**	**0.30**	0.28

Table 3: Overall efficiency comparison of 8 methods: training time (ms) per epoch and sample, test time (ms) per sample, FLOPs (G), and parameter count (M). Geometric mean and Median across 30 UEA datasets are reported.

put, thus independent of sequence length. By initializing the sparse graph via nearest neighbors, it avoids quadratic dependence on variable count, yielding an $O(NC)$ complexity for each forward propagation.

In Table 3, we compare the overall efficiency of 8 MTSC-specific methods on 30 UEA datasets. Clearly, since the incorporated RC networks obviate backpropagation, SVGL trains faster than most baselines, with nearly the fewest parameters. During inference, it also records the lowest FLOPs. Although the test time is slightly longer, likely due to the recursive computation paradigm of RC networks, it remains entirely practical. In comparison, lightweight methods such as TapNet and OS-CNN lag behind in accuracy, while stronger baselines like MPTSNet and TimeMIL demand far greater computational resources. These underscore the superiority of our proposed method.

Ablation Study

Impact of Components We ablate on 30 UEA datasets to assess the impact of key SVGL components:

- **MS**: Multi-Scale Analysis. Removing MS limits the model to extract a single dynamic representation for each variable at scale 1.
- **GL**: Global Graph Learner. Removing GL limits the model to learn only sample-specific SVG structures.
- **SSL**: Sample-Specific Graph Learner. Removing SSL limits the model to learn only the global SVG structure.

The full SVGL is compared with 5 variants. From results in Table 4, each component proves beneficial. Specifically, introducing a multi-scale perspective enhances accuracy by 7.4% (Variant 2 vs. 1) and 7.9 % (SVGL vs. Variant 5). Learning either global or sample-specific SVGs increases by 2.6% and 3.3%, respectively, while learning multi-level SVGs provides an additional 1.3% to 2.0%. These confirm that all components are indispensable, highlighting the effectiveness of capturing scale-variable interactions.

Impact of Nearest Neighbors To examine whether and how sparse SVG initialization affects performance, we vary the number of nearest neighbors k used to form the initial graph. Notably, when $k \geq |\mathcal{V}|$, the graph becomes fully connected. Table 4 shows that average accuracy rises as k

	Components			Average Accuracy	Value of k	Average Accuracy
	MS	GL	SSL			
Variant 1				0.642	$k = 8$	0.739
Variant 2	✓			0.716	$k = 16$	0.754
Variant 3	✓	✓		0.742	$k = 32$	**0.762**
Variant 4	✓		✓	0.749	$k = 64$	0.760
Variant 5		✓	✓	0.683	$k = 128$	0.759
SVGL	✓	✓	✓	**0.762**	$k = 256$	0.758

Table 4: Ablation results on 30 UEA datasets.

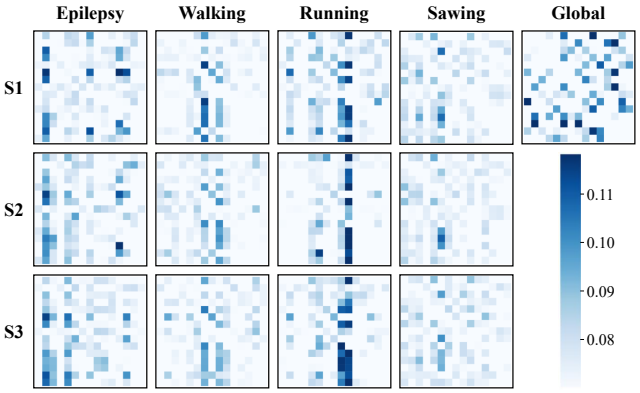

Figure 4: Visualization of global and sample-specific SVGs learned on the Epilepsy dataset, with 3 samples per class displayed. i-p denotes the i-th variable at its key scale p.

grows, then plateaus and even dips slightly after $k = 32$. This implies that a sparse start not only reduces computation but also emphasizes more informative edges. However, excessive sparsity could remove crucial links and harm performance. Therefore, we set $k = 32$ by default as a trade-off.

Case Study

To provide deeper insight into SVGL, we randomly select 3 samples per class in the Epilepsy dataset and visualize the learned multi-level SVGs in Figure 4. From the global view, our method effectively distinguishes the interaction strengths among scale-variable pairs. Furthermore, we find that samples within the same class could share similar association patterns, while different classes display clear differences. These confirm that SVGL can identify key scale-variable interactions and accurately categorize time series.

Conclusion

This paper reveals understudied cross-scale couplings of time series variables. Based on this insight, we propose SVGL, a novel framework that learns key scale-variable interactions for MTSC. SVGL adaptively derives multi-scale dynamic representations for each variable, followed by learning multi-level scale-variable graphs to extract discriminative features. Experiments on 30 UEA datasets demonstrate SVGL's superior accuracy and low training overhead. Ablation further confirms the indispensability of each component. Future work may explore deeper integration across scales and variables to enhance performance.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (No. 62576327, 62206261, 62137002, 62176245), in part by the Fundamental Research Funds for the Central Universities (No. WK2150110039).

References

Bagnall, A.; Dau, H. A.; Lines, J.; Flynn, M.; Large, J.; Bostrom, A.; Southam, P.; and Keogh, E. 2018. The UEA multivariate time series classification archive, 2018. *arXiv preprint arXiv:1811.00075*.

Bianchi, F. M.; Scardapane, S.; Løkse, S.; and Jenssen, R. 2021. Reservoir Computing Approaches for Representation and Classification of Multivariate Time Series. *IEEE Transactions on Neural Networks and Learning Systems*, 32(5): 2169–2179.

Cai, W.; Liang, Y.; Liu, X.; Feng, J.; and Wu, Y. 2024. MSGNet: Learning Multi-scale Inter-series Correlations for Multivariate Time Series Forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 11141–11149.

Chen, A.; Zhou, X.; and Chen, H. 2025. Efficient Anomaly Detection of Irregular Sequences in Ct-Echo Model Space. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 15731–15739.

Chen, H.; Tiňo, P.; Rodan, A.; and Yao, X. 2014. Learning in the Model Space for Cognitive Fault Diagnosis. *IEEE Transactions on Neural Networks and Learning Systems*, 25(1): 124–136.

Chen, X.; Qiu, P.; Zhu, W.; Li, H.; Wang, H.; Sotiras, A.; Wang, Y.; and Razi, A. 2024. TimeMIL: Advancing Multivariate Time Series Classification via a Time-aware Multiple Instance Learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, 7190–7206. PMLR.

Cheng, C.; Liu, X.; Zhou, B.; and Yuan, Y. 2023. Intelligent Fault Diagnosis With Noisy Labels via Semisupervised Learning on Industrial Time Series. *IEEE Transactions on Industrial Informatics*, 19(6): 7724–7732.

Cheng, Z.; Yang, Y.; Wang, W.; Hu, W.; Zhuang, Y.; and Song, G. 2020. Time2graph: Revisiting Time Series Modeling with Dynamic Shapelets. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 3617–3624.

Deng, A.; and Hooi, B. 2021. Graph Neural Network-based Anomaly Detection in Multivariate Time Series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 4027–4035.

Gu, A.; Goel, K.; and Ré, C. 2022. Efficiently Modeling Long Sequences with Structured State Spaces. In *International Conference on Learning Representations*.

Hodrick, R. J.; and Prescott, E. C. 1997. Postwar US Business Cycles: An Empirical Investigation. *Journal of Money, Credit, and Banking*, 1–16.

Jaeger, H. 2001. The "echo state" approach to analysing and training recurrent neural networks-with an erratum note. *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, 148(34): 13.

Jia, Z.; Lin, Y.; Wang, J.; Zhou, R.; Ning, X.; He, Y.; and Zhao, Y. 2020. GraphSleepNet: Adaptive Spatial-Temporal Graph Convolutional Networks for Sleep Stage Classification. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, 1324–1330. International Joint Conferences on Artificial Intelligence Organization.

Jin, M.; Koh, H. Y.; Wen, Q.; Zambon, D.; Alippi, C.; Webb, G. I.; King, I.; and Pan, S. 2024. A Survey on Graph Neural Networks for Time Series: Forecasting, Classification, Imputation, and Anomaly Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Karim, F.; Majumdar, S.; Darabi, H.; and Harford, S. 2019. Multivariate LSTM-FCNs for Time Series Classification. *Neural Networks*, 116: 237–245.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*.

Lai, G.; Chang, W.-C.; Yang, Y.; and Liu, H. 2018. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 95–104.

Liu, H.; Yang, D.; Liu, X.; Chen, X.; Liang, Z.; Wang, H.; Cui, Y.; and Gu, J. 2024. Todynet: Temporal dynamic graph neural network for multivariate time series classification. *Information Sciences*, 677: 120914.

Liu, M.; Zeng, A.; Chen, M.; Xu, Z.; Lai, Q.; Ma, L.; and Xu, Q. 2022. Scinet: Time series modeling and forecasting with sample convolution and interaction. *Advances in Neural Information Processing Systems*, 35: 5816–5828.

Liu, S.; Wei, C.; Zhou, X.; and Chen, H. 2025. Spectral-Aware Reservoir Computing for Fast and Accurate Time Series Classification. In *Forty-second International Conference on Machine Learning*.

Liu, S.; Zhou, X.; and Chen, H. 2025. Multiscale Temporal Dynamic Learning for Time Series Classification. *IEEE Transactions on Knowledge and Data Engineering*, 37(6): 3543–3555.

Luo, D.; and Wang, X. 2024. ModernTCN: A Modern Pure Convolution Structure for General Time Series Analysis. In *The Twelfth International Conference on Learning Representations*, 1–43.

Majumdar, S.; and Laha, A. K. 2020. Clustering and Classification of Time Series Using Topological Data Analysis with Applications to Finance. *Expert Systems with Applications*, 162: 113868.

Mu, Y.; Shahzad, M.; and Zhu, X. X. 2025. MPTSNet: Integrating Multiscale Periodic Local Patterns and Global Dependencies for Multivariate Time Series Classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 19572–19580.

Nie, Y.; Nguyen, N. H.; Sinthong, P.; and Kalagnanam, J. 2023. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. In *The Eleventh International Conference on Learning Representations*.

Schäfer, P.; and Leser, U. 2017. Multivariate time series classification with WEASEL+ MUSE. *arXiv preprint arXiv:1711.11343*.

Song, H.; Rajan, D.; Thiagarajan, J.; and Spanias, A. 2018. Attend and Diagnose: Clinical Time Series Analysis Using Attention Models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Srivastava, P. K.; Desai, D. K.; Nandi, S.; and Lynn, A. M. 2007. HMM-ModE–Improved classification using profile hidden Markov models by optimising the discrimination threshold and modifying emission probabilities with negative training sequences. *BMC Bioinformatics*, 8(1): 1–17.

Tang, W.; Long, G.; Liu, L.; Zhou, T.; Blumenstein, M.; and Jiang, J. 2022. Omni-Scale CNNs: A simple and effective kernel size configuration for time series classification. In *The Tenth International Conference on Learning Representations*.

Tian, C.; Lu, Z.; Zhang, Z.; Yang, H.; Cao, W.; Guo, Z.; Sun, X.; and Jin, L. 2025. HyperMixer: Specializable Hypergraph Channel Mixing for Long-term Multivariate Time Series Forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 20885–20893.

Wang, H.; Peng, J.; Huang, F.; Wang, J.; Chen, J.; and Xiao, Y. 2023. Micn: Multi-scale local and global context modeling for long-term series forecasting. In *The Eleventh International Conference on Learning Representations*.

Wang, S.; Wu, H.; Shi, X.; Hu, T.; Luo, H.; Ma, L.; Zhang, J. Y.; and ZHOU, J. 2024. TimeMixer: Decomposable Multiscale Mixing for Time Series Forecasting. In *International Conference on Learning Representations (ICLR)*.

Wu, H.; Hu, T.; Liu, Y.; Zhou, H.; Wang, J.; and Long, M. 2023. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. In *The Eleventh International Conference on Learning Representations*.

Wu, H.; Wu, J.; Xu, J.; Wang, J.; and Long, M. 2022. Flowformer: Linearizing Transformers with Conservation Flows. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, 24226–24242. PMLR.

Wu, Z.; Pan, S.; Long, G.; Jiang, J.; and Zhang, C. 2019. Graph WaveNet for Deep Spatial-Temporal Graph Modeling. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, 1907–1913. International Joint Conferences on Artificial Intelligence Organization.

Xiong, Y.; and Yeung, D.-Y. 2002. Mixtures of ARMA models for model-based time series clustering. In *2002 IEEE International Conference on Data Mining, 2002. Proceedings.*, 717–720. IEEE.

Yang, J.; Nguyen, M. N.; San, P. P.; Li, X.; and Krishnaswamy, S. 2015. Deep Convolutional Neural Networks on Multichannel Time Series for Human Activity Recognition. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-15*, 3995–4001. AAAI Press.

Younis, R.; and Ahmadi, Z. 2024. HyperTime: A Dynamic Hypergraph Approach for Time Series Classification. In *2024 IEEE International Conference on Data Mining (ICDM)*, 570–579.

Yue, Z.; Wang, Y.; Duan, J.; Yang, T.; Huang, C.; Tong, Y.; and Xu, B. 2022. Ts2vec: Towards universal representation of time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 8980–8987.

Zeng, A.; Chen, M.; Zhang, L.; and Xu, Q. 2023. Are transformers effective for time series forecasting? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 11121–11128.

Zha, D.; Lai, K.-H.; Zhou, K.; and Hu, X. 2022. Towards similarity-aware time-series classification. In *Proceedings of the 2022 SIAM International Conference on Data Mining (SDM)*, 199–207. SIAM.

Zhang, X.; Gao, Y.; Lin, J.; and Lu, C.-T. 2020. Tapnet: Multivariate time series classification with attentional prototypical network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 6845–6852.

Zhang, X.; Zeman, M.; Tsiligkaridis, T.; and Zitnik, M. 2022. Graph-Guided Network for Irregularly Sampled Multivariate Time Series. In *International Conference on Learning Representations*.

Zhang, Y.; and Yan, J. 2023. Crossformer: Transformer Utilizing Cross-Dimension Dependency for Multivariate Time Series Forecasting. In *The Eleventh International Conference on Learning Representations*.

Zhou, T.; Ma, Z.; Wen, Q.; Wang, X.; Sun, L.; and Jin, R. 2022. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International Conference on Machine Learning*, 27268–27286. PMLR.

Zhou, X.; Liu, S.; Chen, A.; and Chen, H. 2024. Learning in CubeRes Model Space for Anomaly Detection in 3D GPR Data. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, 5662–5670.

Zhou, X.; Liu, S.; Chen, A.; Chen, Q.; Xiong, F.; Wang, Y.; and Chen, H. 2023. Underground Anomaly Detection in GPR Data by Learning in the C3 Model Space. *IEEE Transactions on Geoscience and Remote Sensing*, 61: 1–11.