TSCMamba: Mamba Meets Multi-View Learning for Time Series Classification

Anonymous authors
Paper under double-blind review

Abstract

Time series classification (TSC) on multivariate time series is a critical problem. We propose a novel multi-view approach integrating frequency-domain and time-domain features to provide complementary contexts for TSC. Our method fuses continuous wavelet transform spectral features with temporal convolutional or multilayer perceptron features. We leverage the Mamba state space model for efficient and scalable sequence modeling. We also introduce a novel tango scanning scheme to better model sequence relationships. Experiments on 10 standard benchmark datasets demonstrate our approach achieves an average 6.45% accuracy improvement over state-of-the-art TSC models.

1 Introduction

Time series classification (TSC) is a fundamental task in diverse fields abundant with time series data. With the advancing of sensing technologies, multivariate time series (MTS) data have been ubiquitous, and thus TSC over MTS has attracted ever-increasing research attention.

MTS data is usually highly redundant, with significant inter-observation correlations in temporal neighborhoods. This redundancy is similar to that found in images (Bengio et al., 2013), where convolutional kernels are effective in extracting features. Convolutional kernels can be used in the time-domain to extract features from MTS data, but they are limited by their small receptive field, thus typically extracting temporally localized contents. They have limited ability to capture spectrally localized contents or global interactions in the MTS. To remedy this limitation, frequency-domain features can be extracted using transforms such as the Fourier transform, digital wavelet transforms, and the shapelet transform. These transforms can help capture global interactions and spectral patterns in the data and separate noise from signals or useful contents.

MTS data is often noisy (Kang et al., 2014), which may include Gaussian noise, speckle noise, seasonal noise, trend noise, and outliers. Noise can mask the underlying signals, make classification more difficult, and affect the robustness of the algorithms. Thus, it is important to obtain effective representations that are robust to noise. While Transformers can be effective for capturing global interactions (Vaswani et al., 2017), they have been found to be sensitive to noise in the data (Mahmood et al., 2021).

Existing methods for TSC typically focus on a single type of features, such as temporal-domain features or patterns, or frequency-domain patterns. However, these methods do not fully exploit the complementary characteristics that can be obtained from different domains. For example, time-domain features may readily capture temporally local patterns, while frequency-domain features may help capture spectrally local and sensible patterns. Multi-view learning has been successful in capturing complementary characteristics from complex data, robust to noise or disturbances, and has been widely used in image classification and clustering (Peng et al., 2024). However, effective multi-view strategies have not been well investigated for enhancing deep learning-based TSC in the literature.

To address this gap, we propose a novel approach for TSC that effectively integrates features and patterns from both time- and frequency domains. For time-domain features, we leverage an efficient convolutional kernel-based feature representation or project patterns using a fully connected neural network. For frequency-

domain patterns, we adopt features obtained from frequency-domain transformations, such as continuous wavelet transform (CWT). By integrating these features and patterns, we provide comprehensive multi-view contexts for TSC.

Convolutional kernels used in convolutional neural networks (CNNs) excel at capturing temporal dependencies between input features or inter-dependencies between channels. However, these kernels typically have limited lengths, resulting in a limited receptive field that captures local patterns. In contrast, multi-layer perceptrons (MLPs) are fully connected neural networks with a global receptive field, allowing them to learn global patterns and dependencies among input features. While MLPs can capture global patterns, they may not be well-suited for capturing temporally local patterns or temporal/spatial dependencies, as they treat each input feature independently and do not explicitly consider spatial relationships among neighboring features.

To capture spectral-domain patterns, we leverage CWT to decompose and represent a time series. CWT is an invertible wavelet transformation that can perfectly reconstruct the original data, preserving all information. This representation will provide a rich spectral view to complement the temporal views. While CWT has been used to extract features for fault diagnosis in the literature (Wang et al., 2020), it has not been used to develop a general model for TSC. To reduce the computational complexity of CWT, we will compute the CWT coefficients for each time series and use them as input for the spectral view.

The MTS data often has sensible global patterns, such as trend, seasonality, periodicity, cycles, and long-term dependencies. MTS may also comprise informational local patterns, such as peaks, troughs, jumps, local cycles, and plateaus with local trends. These global or local patterns may constitute key information to distinguish the classes of MTS. As different MTS datasets may have different global or local contents, it is critical to leverage sensible MTS patterns for TSC. Thanks to its multi-scale decomposition, the CWT representation for the spectral view contains global and local clues at various levels of the MTS. However, temporal views obtained from the CNN-based feature extraction and MLP-based feature maps may not provide global patterns or local cues as proper contexts for inference of class labels.

To effectively capture discriminative patterns in the temporal domain, we propose a switch mechanism that selects between CNN-based local features and MLP-based global patterns. We utilize the kernel-based feature transformation ROCKET (Dempster et al., 2020) to extract locally representative features while using a fully connected MLP to capture global patterns. The switch mechanism determines which temporal view (local or global) is more informative and integrates it with the CWT-domain features.

Consequently, we have diverse views, representing global temporal features, localized temporal features, and multi-scale spectral features. We will use them to provide discriminative and complementary contextual clues to an effective inference engine to capture long-term dependencies between these features. After obtaining features with these different views for time series data, we fuse them to provide rich contexts for subsequent sequence modeling. As discussed above, MLP is good at capturing global, but not local interactions. CNN can extract localized features but its receptive field is typically limited. We will use a switch gate to determine whether to fuse transform domain features with local features or global patterns in the temporal domain, which will be determined by tuning the model to select between these two approaches.

Having obtained salient contextual cues, we will use a state-of-the-art (SOTA) state-space model (SSM) driven inference engine, Mamba (Gu & Dao, 2023), to decide on the class labels. Mamba is an architecture for sequence modeling, which is a special case of SSM and similar to recursive neural networks (RNNs). The SSM is a classical concept in the fields of control and signal processing (Gu et al., 2021), which uses state variables to represent the system's internal condition and describes how the state variables change over time based on the inputs and the current state. By introducing a new selective updating mechanism of the hidden states, called selective state spaces, it can selectively update a subset of state dimensions based on the input at each time step. As a result, Mamba can efficiently capture long-range dependencies. Mamba-based models have demonstrated competitive performance on various tasks, such as language modeling (Gu & Dao, 2023; Dao & Gu, 2024), time series forecasting (Ahamed & Cheng, 2024b), DNA sequence modeling (Gu & Dao, 2023), tabular data learning (Ahamed & Cheng, 2024a), and audio generation (Shams et al., 2024).

Unlike popular Transformers, which are sequence modeling architectures with quadratic time complexity in sequence length, Mamba achieves linear time complexity. This makes it more suitable for processing long sequences and scaling to larger datasets. Therefore, by using Mamba, our TSC model is efficient in training and inference, with reduced computational costs and memory requirements compared to existing SOTA models. Moreover, we introduce a novel sequence scanning scheme for the Mamba block, which uses essentially the same memory footprint but demonstrates higher accuracy than the vanilla Mamba scanning. Through extensive experiments, we will demonstrate that our multi-view Mamba-based approach outperforms or matches the performance of existing SOTA models that typically have more computationally expensive models, with a superior average accuracy over a range of 10 standard benchmarking datasets.

In summary, the contribution of this paper includes but is not limited to the following:

- We propose a novel multi-view approach for TSC, which seamlessly integrates frequency-domain and time-domain features to provide complementary and discriminative contexts for classification. In particular, we propose to fuse features from both domains with a gating scheme. Our approach can effectively leverage local and global patterns that characterize the MTS classes, enhancing the discriminative power.
- We leverage a concurrent SSM technique, Mamba, for sequence modeling to capture long-term dependencies within the MTS with linear efficiency and scalability. Moreover, we propose an innovative Mamba-based scanning scheme, called tango scanning, to scan the integrated features to identify salient contents within complex contexts. This scanning is demonstrated to be more effective in modeling the relationships in the sequence than that of vanilla Mamba block for TSC.
- We first use the CWT for multi-scale representation of MTS in the general task of TSC. This joint temporal and spectral transformation allows for efficient frequency-domain feature extraction and subsequent fusion of time-domain features with frequency-domain features.
- Our extensive experiments validate the proposed approach, which demonstrates superior performance to various existing SOTA models over 10 standard benchmarking datasets. On average, an improvement of 6.45% in accuracy is obtained.

With these contributions, we expect our novel approach to help advance real-world TSC applications in diverse fields of research and everyday life. In the following sections, we first briefly review related works, then presents our approach in detail, and finally demonstrate extensive experimental results and ablation studies to conclude the paper.

2 Related Works

In this section, we provide a brief review of relevant methods for TSC in the literature, focusing on works using machine learning or deep learning. We group existing methods into 4 categories: traditional methods like DTW, deep learning approaches using CNNs or RNNs, Transformer architectures, and methods based on state-space models.

Traditional TSC methods include techniques like Dynamic Time Warping (DTW) (Berndt & Clifford, 1994), which measures the similarity between time series by aligning them in a non-linear way of dynamic programming. Tree-based methods like XGBoost (Chen & Guestrin, 2016) have also been applied to the TSC task.

In recent years, deep learning approaches have become increasingly popular for TSC. Various MLP-based methods have been proposed, including DLinear by Zeng et al. (2023) and LightTS by Zhang et al. (2022). DLinear constructs a simple model based on MLP, while LightTS uses light sampling-oriented MLP. These models are generally efficient in computations. Convolutional neural networks (CNNs) have been adapted for TSC, such as ROCKET (Dempster et al., 2020) which uses random convolutional kernels for fast and accurate classification. CNN has been also used by Franceschi et al. (2019) for learning representations of multivariate time series in an unsupervised way, which is then further leveraged for TSC. Besides CNNs, recursive neural networks (RNNs) such as long-short-term memory (LSTM) (Hochreiter & Schmidhuber,

1997) and a variant, a gated recursive unit (GRU), has also been adopted for TSC. Moreover, CNN and RNN have been combined to handle TSC effectively (Lai et al., 2018).

Transformers by Vaswani et al. (2017), originally used for natural language processing, have been adapted for time series modeling. Reformer by Kitaev et al. (2020) introduces efficiency improvements to handle longer sequences. Numerous Transformer variants have been proposed to better model the unique characteristics of time series, such as handling non-stationarity with on-stationary Transformers by Liu et al. (2022), combining exponential smoothing with ETSformer in Woo et al. (2022), and using decomposition and auto-correlation with Autoformer in Wu et al. (2021). Other variants include Pyraformer (Liu et al., 2021), which uses pyramidal attention to reduce complexity, Flowformer (Wu et al., 2022b), which linearizes Transformers using conservation flows, and Informer (Zhou et al., 2022), which focuses on efficient long sequence forecasting by exploiting frequency-enhanced decomposition. Notably, TimesNet (Wu et al., 2022a) models the temporal 2D variations in time series data using a hierarchical structure of temporal blocks. By combining 2D convolutions, multi-head self-attention, and a novel positional encoding scheme, it can capture both local patterns and long-range dependencies in time series and obtain state-of-the-art performance. Despite the impressive performance of Transformer-based models, Zeng et al. (2023) have shown that MLP-based models can be more effective in many scenarios.

Recently, a method called LSSL by Gu et al. (2022) has been proposed for TSC with a structured SSM called S4. It employs a special parameterization called the Diagonal Plus Low-Rank form to represent the state transition matrix, enabling efficient computation over long sequences.

3 Methodology

In this section, we describe all the steps of our proposed method step by step. The subsections describe the necessary components and procedures of TSCMamba.

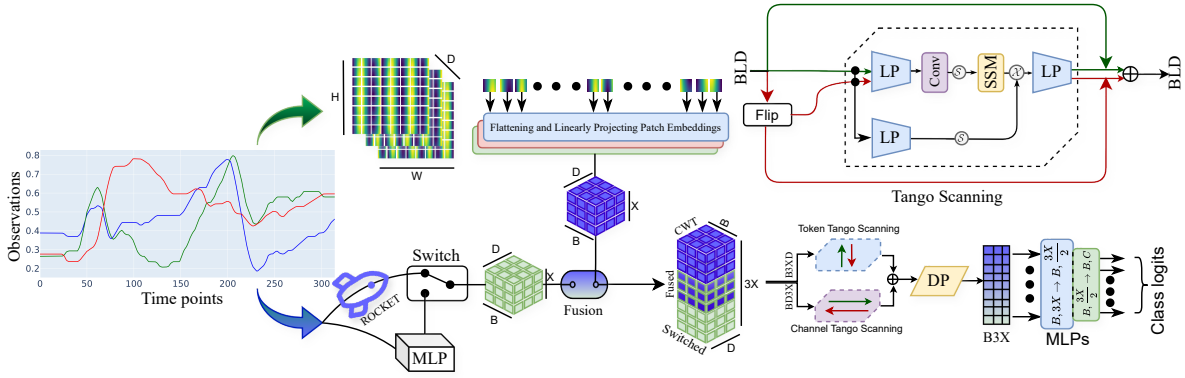

Figure 1: Schematic illustration of the proposed method (TSCMamba). This diagram illustrates the architecture of our approach, featuring tango scanning. Here, DP refers to Depth-wise Pooling (DP) and LP refers to Linear Projection (LP). A switch gate selectively activates the utilization of either ROCKET or MLP-derived features. The MLP module, depicted in the bottom right, comprises two layers with an optional dropout mechanism interspersed for regularization

3.1 Spectrogram Representation

We have chosen Continuous Wavelet Transform (CWT) to represent raw signals in spectrograms. CWT potentially surpasses space-time Fourier transform, fast Fourier transform (FFT), and digital wavelet transform (DWT) by providing superior time-frequency localization and multi-resolution analysis. CWT's adaptable wavelets enhance feature extraction and noise reduction while better-handling edge effects. This makes CWT particularly suitable for analyzing non-stationary signals. CWT's continuous and detailed representa-

tion may offer a significant advantage over the discrete nature of DWT. This renders CWT highly effective for precise time-frequency analysis.

Among a variety of wavelets, the Morlet wavelet in Equation 1 is employed in this paper due to its capturing both amplitude and phase effectively:

$$\psi(t) = (\pi^{-1/4})(1 - \frac{t^2}{\sigma^2}) \exp(-\frac{t^2}{2\sigma^2}) \cos(2\pi f t), \tag{1}$$

where σ is the scale parameter controlling the width of the wavelet, and f is the frequency parameter that controls the frequency of the cosine function. In this paper, we adopt $\sigma^2 = 1$ and $f = 5/(2\pi)$ to balance computational cost and the expressiveness of the obtained wavelet features. However, keeping these parameters learnable may potentially benefit the classification accuracy. The smooth and symmetric shape of the Morlet wavelet minimizes distortions and edge effects, resulting in a clear and interpretable time-frequency representation. Using the wavelet function, we obtain a 2-D representation of the size $L_1 \times L_1$ for each channel of an original MTS input sample of size L. In this paper, we adopt $L_1 = 64$ for computational efficiency and expressiveness of the obtained wavelet features. We summarize this CWT feature extraction process in S-Algorithm 1. Since conversion from time signals to CWT representation is not learnable, we move this to the data pre-processing part, while regarding only the patch embedding module to be learnable. This helps our model to achieve lower FLOPs and faster training.

With the resultant CWT representation of size $D \times L_1 \times L_1$, we further perform patch embedding using a Conv2D layer (kernel size=stride=p, padding=0), where p=8 is patch size. Later with flattened patches, we utilize a feed-forward network (FFN) to obtain patches of size $D \times X$ for each MTS sample. The FFN consists of one fully connected layer with an input dimension $(\frac{L_1}{p})^2$ and an output dimension X, as shown for the projected space in Figure 1. It is used to extract features within the CWT representation. For each batch of size B, the resultant tensor for representing CWT features is denoted by $\mathcal{W} \in \mathcal{R}^{B \times D \times X}$.

3.2 Temporal Feature Extraction

To complement the frequency-domain features, we extract time-domain features. As previously discussed, different MTS datasets may have global or local features or patterns that discriminate between different classes. Capturing these features is essential for accurate classification. We leverage two different approaches to capture such features.

Extracting Local Features with Convolutional Kernels in Unsupervised Fashion. Convolutional kernels usually have limited receptive fields, thus focusing on the extraction of local features. Since an MTS dataset may have local features at multiple temporal scales, it is sensible to capture local features within various widths of receptive fields. To this end, we employ the ROCKET approach (Dempster et al., 2020) to extract local features within various local neighborhoods for each channel in an unsupervised fashion. Here, it is to be noted that we only utilize the time domain to extract the kernel-based features, we do not utilize the class labels of the corresponding features. Therefore, our improvement in performance does not solely rely on the ROCKET method, rather it works as a performance booster in certain datasets.

ROCKET is a randomized algorithm that uses a set of randomized convolutional kernels to extract features from time series data. The method is suitable for capturing local features at various scales due to its randomized nature and the use of kernels with different sizes and strides. The procedure first randomly generates a set of convolutional kernels, each with a specific size and stride. Next, it convolves each kernel with the time series data to generate a feature map. The procedure is summarized in S-Algorithm 2. ROCKET generates random convolutional kernels, including random length and dilation. It transforms the time series with two features per kernel. The global max pooling and the proportion of positive values (PPV). This fits one set of parameters for individual series.

We apply the ROCKET feature extraction method to each channel of length L to form a feature vector of length X. We input our training data as a result, with the input tensor of size $B \times D \times L$, we obtain a tensor $\mathcal{V}_L \in \mathcal{R}^{B \times D \times X}$ that represents the local features. We utilized the sktime implementation of ROCKET to achieve this (Király et al., 2024; Löning et al., 2019).

Global Feature Extraction with MLP. MLP has a receptive field covering the entire input, allowing the resulting feature vectors to capture the global characteristics of the MTS data. Independently for each channel of the input MTS of size $D \times L$, we utilize a one-layer MLP with linear activation to obtain a feature vector of size $D \times X$. Therefore, with the input tensor of size $B \times D \times L$, we obtain a tensor $\mathcal{V}_G \in \mathcal{R}^{B \times D \times X}$ representing the global features.

3.3 Fusing Multi-View Representations

After obtaining the spectrogram features using CWT and the temporal features at both local and global levels, we fuse these features to effectively exploit the complementary information in these multi-view representations. Through our empirical study, we observe that for many MTS data, either local features or global features in the temporal domain play a dominant role in discriminating between classes for TSC. This observation motivates us to fuse the spectrogram features with either global or local features in the temporal domain. Denote the temporal features by $\mathcal{V}$, which is either $\mathcal{V}_G$ or $\mathcal{V}_L$. Then, the fused feature map $\mathcal{V}_W$ will be calculated as follows:

$$\mathcal{V}_W = \mathcal{W} \otimes \mathcal{V}, \tag{2}$$

where $\otimes$ represents an element-wise operation. In this paper, this is either a multiplicative or additive operation such that

$$\{\mathcal{V}_W\}_{ijk} = \lambda \mathcal{V}_{ijk} * (2-\lambda)\mathcal{W}_{ijk}, \quad \text{or,} \quad \{\mathcal{V}_W\}_{ijk} = \lambda \mathcal{V}_{ijk} + (2-\lambda)\mathcal{W}_{ijk}, \tag{3}$$

where $\lambda \geq 0$ is a learnable parameter that determines the balance between the spectrogram features and the temporal features, $1 \leq i \leq B$, $1 \leq j \leq D$, $1 \leq k \leq X$. We set the λ as a learnable parameter while the initial value of $\lambda = 1.0$. Therefore, the optimal value of λ will be determined during the training process. The initial value of $\lambda = 1.0$ ensures a balanced focus initially between temporal and spectral domain features.

After obtaining the fused temporal-spectral features, we composite it with the tensors containing the multi-view features into a new tensor $\mathcal{U} = \mathcal{W} \| \mathcal{V}_W \| \mathcal{V} \in \mathcal{R}^{B \times D \times 3X}$, where $\|$ is a concatenation operation. We use a switching mechanism to make the choice between $\mathcal{V} = \mathcal{V}_G$ or $\mathcal{V} = \mathcal{V}_L$. This mechanism is implemented as a learnable binary mask that selects either the global or local temporal features during the training process. The final state of this switch will be determined by the optimization in the training process and tuned based on datasets for optimal performance.

3.4 Inferring with Time-Channel Tango Scanning

With the integrated temporal-spectral contextual representations contained in tensor $\mathcal{U}$, we can now learn salient representations to capture important relationships between features, particularly long-term dependencies. To achieve this, we construct tokens by treating each feature vector in $\mathcal{U}$ along the time and channel dimensions as a separate token. Subsequently, we leverage Mamba, a type of SSM, for modeling the token sequences. Mamba is designed for capturing discriminative contents by selectively scanning the token sequences. This selective scan ability allows the model to focus on the most informative parts of the sequence while ignoring less relevant information. By doing so, Mamba can effectively capture long-term dependencies and identify salient features that are most useful for classification.

Compared to other SSMs, Mamba has the advantage of being computationally efficient and able to handle long sequences. It achieves this by using a sparse attention mechanism that reduces the complexity of token-to-token interactions. This makes Mamba particularly well-suited for processing time series data, where the sequences can be lengthy and contain complex temporal dependencies.

Vanilla Mamba Block: Inside a Mamba block, two fully-connected layers in two branches calculate linear projections. The output of the linear mapping in the first branch passes through a 1D causal convolution and SiLU activation $\mathcal{S}(\cdot)$ (Elfwing et al., 2018), then a structured SSM. The continuous-time SSM maps an input function or sequence $u(t)$ to output $v(t)$ through a latent state $h(t)$:

$$dh(t)/dt = A\ h(t) + B\ u(t), \quad z(t) = C\ h(t), \tag{4}$$

where $h(t)$ is N-dimensional, with N also known as a *state expansion factor*, $u(t)$ is D-dimensional, with D being the *dimension factor* for an input token, $z(t)$ is an output of dimension D, and A, B, and C are coefficient matrices of proper sizes. This dynamic system induces a discrete SSM governing state evolution and outputs given the input token sequence through time sampling at $\{k\Delta\}$ with a Δ time interval. This discrete SSM is

$$h_k = \bar{A}\ h_{k-1} + \bar{B}\ u_k, \quad z_k = C\ h_k, \tag{5}$$

where h_k, u_k, and z_k are respectively samples of $h(t)$, $u(t)$, and $z(t)$ at time $k\Delta$,

$$\bar{A} = \exp(\Delta A), \quad \bar{B} = (\Delta A)^{-1}(\exp(\Delta A) - I)\Delta B. \tag{6}$$

For SSMs, diagonal A is often used. Mamba makes B, C, and Δ linear time-varying functions dependent on the input. In particular, for a token u, $B, C \leftarrow Linear_N(u)$, and $\Delta \leftarrow softplus(parameter + Linear_D(Linear_1(u)))$, where $Linear_p(u)$ is a linear projection to a p-dimensional space, and $softplus$ activation function. Furthermore, Mamba also has an option to expand the model dimension factor D by a controllable dimension expansion factor E. Such coefficient matrices enable context and input selectivity properties (Gu & Dao, 2023) to selectively propagate or forget information along the input token sequence based on the current token. Denote the discretization operation by $\Delta = \tau_\Delta(parameter + s_\Delta)$, where τ_Δ and s_Δ are both functions of the input. For the special case of univariate sequences, the selectivity property has been mathematically proved (Gu & Dao, 2023), as shown in the following:

Theorem 1. *(Gu and Dao 2023) When $N = 1, A = -1, B = 1$, $s_\Delta = Linear(x)$, and $\tau_\Delta = softplus$, then the selective SSM recurrence takes the form of*

$$h_k = (1 - g_k)\ h_{k-1} + g_k\ u_k, \quad \textit{and} \quad g_k = \sigma(Linear(u_k)), \tag{7}$$

where g_k is the gate.

This theorem states that the hidden state is a convex combination of the current input token and the previous hidden state, with the combination coefficient controlled by the current input token. Moreover, it is pointed out that the parameter g_k is responsible for selecting the input contents u_k from the sequence, plays a role similar to a gating mechanism in the RNN model, thus connecting the selective SSM to the traditional RNN.

After obtaining the SSM output, it is multiplicatively modulated with the output from the second branch before another fully connected projection. The second branch in the Mamba block simply consists of a linear mapping followed by a SiLU.

Tango Scanning: The selectivity ability of Mamba depends on the ordering of the tokens in the sequence because the hidden state at time n is constructed causally from history tokens as determined by the ordering of the tokens. If the history tokens do not contain informational contexts, Mamba may provide less effective predicted output. To alleviate this potential limitation of causal scanning, we construct a dedicated module to extend a vanilla Mamba block, as shown in Figure 1. Each module comprises one vanilla Mamba block. On the input side, the module accepts a sequence in a forward fashion as input and then inverts the sequence to accept it as input again. At the output side, the output of the vanilla Mamba block with forward sequence and that with the inverted sequence are added element-wise. The operations are represented as follows. Denote an input token sequence by $v = [v_1, \cdots, v_M]$, where $v_i \in \mathcal{R}^D$, and $v \in \mathcal{R}^{D \times M}$ is the matrix representation of the token sequence with M being the sequence length. We will first get a reverse-flipped sequence $v^{(r)}$ by inverting the ordering of the elements in v. Tango scanning performs the following operations to obtain the output sequence $s^{(o)}$:

$$v^{(r)} = Reverse(v) = [v_M, v_{M-1}, \cdots, v_1], \tag{8}$$

$$a = Mamba(v), \quad a^{(r)} = Mamba(v^{(r)}), \tag{9}$$

$$s^{(o)} = v \oplus a \oplus v^{(r)} \oplus a^{(r)}, \tag{10}$$

where $Reverse(\cdot)$ denotes the flipping operation of a sequence, $Mamba(\cdot)$ denotes a vanilla Mamba block, and $\oplus$ denotes element-wise addition. The last equation 10 represents the element-wise addition for information fusion. Notably, the same Mamba block is used for the forward sequence v and the reverse-flipped sequence

$v^{(r)}$. By doing so, the SSM in this block will be trained to update the hidden state variable more effectively than using simply the forward scanning of the vanilla Mamba. Because of the sharing of one Mamba block (and thus one SSM) with two sequences that are flips of each other, we regard it as a dancer's one body with two concerted legs and hence call it tango scanning.

Unlike the bi-directional Mamba block in Behrouz & Hashemi (2024); Schiff et al. (2024) that uses two separate SSMs with one for forward direction and the other for backward direction, our tango scanning block uses only a single Mamba block. Importantly, for the inverted sequence in our tango scanning module, the output from the Mamba block is not re-inverted back to the original order before performing element-wise addition. In other words, a tango scanning module only involves sequence inversion once. On the contrary, the bi-directional Mamba block (Behrouz & Hashemi, 2024; Schiff et al., 2024) needs to re-invert the output from the vanilla Mamba block. Empirically, we will demonstrate that our tango scanning can effectively update the hidden state variable while maintaining essentially the same memory footprint as the vanilla Mamba block.

Performing Tango Scanning in Time and Channel Dimensions: The MTS data have significant patterns, correlation structures, and temporal long-term dependencies. To model the relationship in the temporal dimension, we perform tango scanning temporally for every channel. The processed embedded representation with tensor size $B \times 3X \times D$ is transformed using Tango Scanning. Specifically, with each D-dimensional feature point across all channels regarded as a token, we have a token sequence with dimension factor D and length $3X$ as input to the Mamba block in the tango scanning module. This yields an output tensor of size $B \times 3X \times D$. That is, by denoting $u^{(t)} = [u_1^{(t)}, \cdots, u_{3X}^{(t)}]^T \in \mathcal{R}^{3X \times D}$ as the token sequence formed along the time direction for a time series (in the batch), we have

$$s^{(t)} = Tango\_Scanning(u^{(t)}), \tag{11}$$

where $s^{(t)} = [s_1^{(t)}, \cdots, s_{3X}^{(t)}]^T \in \mathcal{R}^{3X \times D}$. By leveraging Mamba, we will extract salient features and context cues from the input token sequence. Particularly, the output sequence s_t captures the between-time-point interactions along the temporal direction.

Because the MTS data often have significant correlations along the channel dimension, we will also model relationships across channels. To this end, we first form our tensor to have size $B \times D \times 3X$ and then we transform it using our tango scanning. Specifically, the whole univariate sequence of each channel is used as a token with a dimension factor $3X$ for the Mamba block in the tango scanning module. Thus, we form a token sequence of length D, with each token having dimension $3X$. This token sequence will be input to our tango scanning module, yielding an output tensor of size $B \times D \times 3X$. That is, by denoting $u^{(c)} = [u_1^{(c)}, \cdots, u_D^{(c)}]^T \in \mathcal{R}^{D \times 3X}$ as the token sequence formed along the channel dimension, we have

$$s^{(c)} = Tango\_Scanning(u^{(c)}), \tag{12}$$

where $s^{(c)} = [s_1^{(c)}, \cdots, s_D^{(c)}]^T \in \mathcal{R}^{D \times 3X}$. Note that the Tango Scanning module used in Eq. (12) is different from the one used in Eq. (11) and utilizes a separate Mamba module. The output sequence s_c captures the between-channel interactions along the temporal direction. It is critical to account for the inter-relationships across channels when the MTS data have many channels.

After obtaining the outputs from the time-wise scanning u_t and the channel-wise scanning u_c, we will perform another fusion at the Mamba-transformed sequence level:

$$z = (s^{(t)})^T \oplus s^{(c)}, \tag{13}$$

where $\oplus$ denotes element-wise addition of matrices, , and $(s^{(t)})^T$ is the transpose of $s^{(t)}$. The resultant fused sequence is of size $D \times 3X$.

3.5 Output Class Representation

The fused tensor of size $B \times D \times 3x$ will be used to distill class information (class logits). First, we perform depth-wise pooling (DP) (Figure 1) to aggregate information across channels. Specifically, given a fused

sequence $z \in \mathcal{R}^{D \times 3X}$, we have

$$\bar{z} = DP(z), \tag{14}$$

where $DP(\cdot)$ denotes the depth-wise pooling and the output $\bar{z} \in \mathcal{R}^{3x}$. DP can be either average pooling or max pooling. We regard these two pooling operations as two possible values of DP. Given a dataset, the specific pooling will be determined in the training stage.

Subsequently, we will employ an FFN of two layers with an optional dropout mechanism interspersed for regularization:

$$\bar{z}^{(1)} = MLP(\bar{z}), \quad \bar{z}^{(2)} = MLP(\bar{z}^{(1)}), \tag{15}$$

where $\bar{z}$ is projected into vectors $\bar{z}^{(1)} \in \mathcal{R}^{3x/2}$ and $\bar{z}^{(2)} \in \mathcal{R}^{C}$. The class labels will be determined based on $\bar{z}^{(2)}$. To train the proposed network, which we call TSCMamba, we employ a cross-entropy loss (CE) on the output of the second layer of the FFN, $\bar{z}^{(2)}$.

4 Experiments and Result Analysis

In this section, we present the results of our experiments on benchmark datasets for time series classification tasks. We evaluate the performance of our proposed method, TSCMamba, and compare it with several state-of-the-art baseline models. The results demonstrate the effectiveness of TSCMamba in handling complex time series classification tasks.

4.1 Datasets

We evaluated the performance of our proposed method, TSCMamba, on 10 benchmark datasets for time series classification tasks following TimesNet (Wu et al., 2022a). These datasets are commonly used in the literature and are representative of various domains, including image, audio, and sensor data. We present dataset statistics in S-Table 1. These datasets are sourced from a diverse set of domains and contain a diverse range of classes, channels, and time-sequences leading to a robust benchmark for evaluating classification tasks. Moreover, some datasets contain more data in the Test set than the Train set (EC, HW, HB, JV, SCP1, UG) making the time-series classification a harder task. More domain-related information can be found in Bagnall et al. (2018).

4.2 Experimental Environment

All experiments were conducted using the PyTorch framework (Paszke et al., 2019) with NVIDIA 4× V100 GPUs (32GB each). The model was optimized using the ADAM algorithm (Kingma & Ba, 2014) with Cross-Entropy loss following TimesNet (Wu et al., 2022a). Moreover, the baseline results are utilized from TimesNet (Wu et al., 2022a) paper for a fair comparison (Same train-test set across the methods). The batch size, epochs, and initial learning rate varied on the datasets for optimal performance. Moreover, the

Table 1: Classification Accuracy (%) for Various Models. The . symbol in Transformer models denotes the specific type of ∗former used. The best average result is in **bold** and second best is <u>underlined</u>.

Datasets	Methods																			
	DTW (1994)	XGBoost (2016)	Rocket (2020)	LSTM (1997)	LSTNet (2018)	LSSL (2022)	TCN (2019)	Trans (2017)	Re. (2020)	In. (2021)	Pyra (2021)	Auto. (2021)	Station. (2022)	FED. (2022)	ETS. (2022)	Flow. (2022b)	DLinear (2023)	LightTS. (2022)	TimesNet (2022a)	TSCMamba Ours
EC	32.3	43.7	45.2	32.3	39.9	31.1	28.9	32.7	31.9	31.6	30.8	31.6	32.7	31.2	28.1	33.8	32.6	29.7	35.7	62.0
FD	52.9	63.3	64.7	57.7	65.7	66.7	52.8	67.3	68.6	67.0	65.7	68.4	68.0	66.0	66.3	67.6	68.0	67.5	68.6	69.4
HW	28.6	15.8	58.8	15.2	25.8	24.6	53.3	32.0	27.4	32.8	29.4	36.7	31.6	28.0	32.5	33.8	27.0	26.1	32.1	53.3
HB	71.7	73.2	75.6	72.2	77.1	72.7	75.6	76.1	77.1	80.5	75.6	74.6	73.7	73.7	71.2	77.6	75.1	75.1	78.0	76.6
JV	94.9	86.5	96.2	79.7	98.1	98.4	98.9	98.7	97.8	98.9	98.4	96.2	99.2	98.4	95.9	98.9	96.2	96.2	98.4	97.0
PS	71.1	98.3	75.1	39.9	86.7	86.1	68.8	82.1	82.7	81.5	83.2	82.7	87.3	80.9	86.0	83.8	75.1	88.4	89.6	90.2
SCP1	77.7	84.6	90.8	68.9	84.0	90.8	84.6	92.2	90.4	90.1	88.1	84.0	89.4	88.7	89.6	92.5	87.3	89.8	91.8	92.5
SCP2	53.9	48.9	53.3	46.6	52.8	52.2	55.6	53.9	56.7	53.3	53.3	50.6	57.2	54.4	55.0	56.1	50.5	51.1	57.2	66.7
SA	96.3	69.6	71.2	31.9	100.0	100.0	95.6	98.4	97.0	100.0	99.6	100.0	100.0	100.0	100.0	98.8	81.4	100.0	99.0	99.0
UG	90.3	75.9	94.4	41.2	87.8	85.9	88.4	85.6	85.6	85.6	83.4	85.9	87.5	85.3	85.0	86.6	82.1	80.3	85.3	93.8
Avg.	67.0	66.0	72.5	48.6	71.8	70.9	70.3	71.9	71.5	72.1	70.8	71.1	72.7	70.7	71.0	73.0	67.5	70.4	<u>73.6</u>	**80.05**

Table 2: FLOPs comparison among the top performing methods. The values are represented in GigaFLOPS (G) or TeraFlops (T), where 1TFLOPs=1000GFLOPs and a lower value indicates better computational efficiency.

Methods	EC	FD	HW	HB	JV	PS	SCP1	SCP2	SA	UG
Flow. Wu et al. (2022b)	1.06T	37.97G	92.21G	246.37G	15.76G	94.02G	542.64G	697.74G	50.33G	190.82G
TimesNet. Wu et al. (2022a)	1.11T	161.93G	115.88G	182.69G	48.15G	**74.18G**	503.62G	2.33T	26.00G	247.73G
TSCMamba (Ours)	**1.69G**	**11.53G**	**27.24G**	**8.39G**	**12.33G**	2.84T	**3.42G**	**11.11**G	**0.78G**	**13.86G**

optimization was performed utilizing a cosine-annealing learning rate scheduler. We measure the prediction performance of our method using accuracy metric where larger values indicate better prediction accuracy.

Baseline Models In this study, we evaluate the performance of our proposed method, TSCMamba, against 19 state-of-the-art baselines, encompassing Transformer-based (Wu et al., 2022a; Vaswani et al., 2017; Kitaev et al., 2020; Zhou et al., 2021; Liu et al., 2021; Wu et al., 2021; Liu et al., 2022; Zhou et al., 2022; Woo et al., 2022; Wu et al., 2022b), CNN-based (Franceschi et al., 2019), RNN-based (Hochreiter & Schmidhuber, 1997; Lai et al., 2018; Gu et al., 2022), MLP-based (Zeng et al., 2023; Zhang et al., 2022), and classical machine learning-based methods (Berndt & Clifford, 1994; Chen & Guestrin, 2016; Dempster et al., 2020). Therefore the comparison among these methods following TimesNet (Wu et al., 2022a) provides strong recent baselines from various aspects of machine learning.

4.3 Predictive Performance Comparison

The comprehensive results are presented in Table 1. Notably, our approach achieves a substantial improvement of **6.45%** over the existing best baseline, TimesNet (Wu et al., 2022a), which is a large margin compared to TimesNet's own improvement of 0.6% over the previous best baseline, Flowformer (Wu et al., 2022b). This notable performance gain solidifies TSCMamba as a strong contender for the TSC task. We plan to release our code and checkpoints for Table 1 publicly. While in Table 1, we present our best-achieved results, we also demonstrate TSCMamba's generalizability across 5 runs with mean and error-bar (standard-deviation) in S-Figure 4. From Table 1, it is evident that some methods may perform well in some datasets (TimesNet (Wu et al., 2022a) on JV, SA), however, may lack the performance in a large margin on other datasets (TimesNet (Wu et al., 2022a) on EC, HW). On the contrary, our method keeps a balance across the datasets while showing a large improvement in average.

4.4 Computational Complexity

In this study, we compared the floating-point operations (FLOPs) of the top-performing methods presented in Table 1. To calculate FLOPs, we set a batch size of 16 across all baselines. For our method, we employed the best-performing hyperparameters, whereas for other baselines, we utilized the recommended parameters specified in the official TimesNet code (Wu et al., 2022a) and Flowformer (Wu et al., 2022b). We leveraged the source code from Ye (2023) to calculate FLOPs. The overall FLOPs, including both forward and backward passes, are presented in Table 2. Notably, our method achieves substantial improvements in terms of FLOPs across all datasets, with the exception of PEMS-SF (PS). This anomaly can be attributed to the projected space (X) used to achieve the best result for this dataset, which was set to 1024, thereby impacting the total FLOPs for this dataset only.

5 Ablation

5.1 Component-wise Ablation

In this section, we conduct an ablation study to investigate the contribution of individual components in our proposed method. The results are presented in Table 3. A notable observation is that the performance

Table 3: Ablation experiments on particular components in our method.

Mamba	Avg.Pool	ROCKET	AF	EC	FD	HW	HB	JV	PS	SCP1	SCP2	SA	UG
✓	✓	✓	✓	62.0	57.0	53.3	74.1	93.0	90.2	92.5	66.7	94.1	93.8
✗	✓	✓	✓	33.1	63.2	34.1	73.2	85.4	81.5	86.7	57.2	74.0	89.1
✓	✗	✓	✓	31.6	64.2	52.0	74.1	94.1	63.0	86.7	60.6	96.7	92.8
✓	✓	✗	✓	31.6	69.4	24.8	76.6	97.0	87.3	91.8	58.3	97.6	86.2
✓	✓	✓	✗	30.0	51.5	49.3	72.7	91.4	84.4	88.7	58.9	90.0	90.3

degrades in a large margin across all datasets when the Mamba modules are not utilized, highlighting the importance of incorporating Mamba in our approach. Specifically, when Mamba is not employed (2nd row), the intermediate values are bypassed by scanning operations and directly fed into the DP and MLPs for class logit prediction. In the 3rd row, we present depth-wise max-pooling instead of average-pooling, resulting in an input shape of B,3X. Furthermore, when ROCKET-extracted features are not utilized (4th row), we resort to MLP-extracted features, where the former are non-learnable and the latter are learnable. Additionally, in the 5th row, we explore the effect of replacing additive fusion (AF) with multiplicative fusion (MF), as detailed in Sec.3.3. Notably, while Table 3 largely mirrors the best performance reported in Table 1 across most datasets, the SpokenArabicDigits (SA) dataset exhibits optimal performance when employing depth-wise max-pooling and MLP-based features.

5.2 Hyperparameter Sensitivity

In this section, we discuss the key settings for the Mamba model, as detailed in Gu & Dao (2023). The model operates with four main settings: model dimension (d_model), SSM state expansion factor (d_state), local convolution width (d_conv), and block expansion factor (expand). In our experiments, we automatically set the model dimension based on the input data, while we adjust the other three settings. The importance of fine-tuning these parameters is evident from our tests, shown in Figure 2, which clearly demonstrate their impact on our model's performance. In addition to Mamba's hyperparameters, we also tuned the dimension of the projected space (X) mentioned in 1.

5.3 Effectiveness of Tango Scanning

Although, our way of scanning (Tango Scanning) at first glance may seem counterintuitive to only forward-based scanning and with another additional reverse flip based scanning (Schiff et al., 2024), it provides substantial improvements in accuracy. This approach, as demonstrated its effectiveness by Figure 3, outperforms traditional forward-scanning and other reverse-flip-based scanning for TSC tasks, making it a valuable strategy for complex scanning scenarios. Our tango scanning is explained in more detail in section 3.4.

6 Conclusion and Future Work

We present TSCMamba, an innovative approach for Time Series Classification (TSC) designed to enhance performance while maintaining lower Floating Point Operations (FLOPs). TSCMamba leverages multi-view learning to analyze different views of time-series data, including local and/or global features extracted from time and frequency domains, thereby capturing the essential, discriminative patterns of real-world time-series data. Moreover, the proposed tango scanning mechanism demonstrates TSCMamba's superiority over conventional scanning methods through extensive experimental validation. Our comprehensive experiments highlight TSCMamba's exceptional performance in terms of both accuracy and computational efficiency, consistently outperforming current state-of-the-art methods. These results suggest that TSCMamba can serve as a robust and efficient solution for a wide range of TSC applications. Looking ahead, our future work will focus on further enhancing TSCMamba by incorporating self-supervised learning techniques and extending its capabilities to multiple-task learning, in addition to the classification task. We also plan to explore the adaptability of TSCMamba across more diverse and complex time-series datasets, aiming

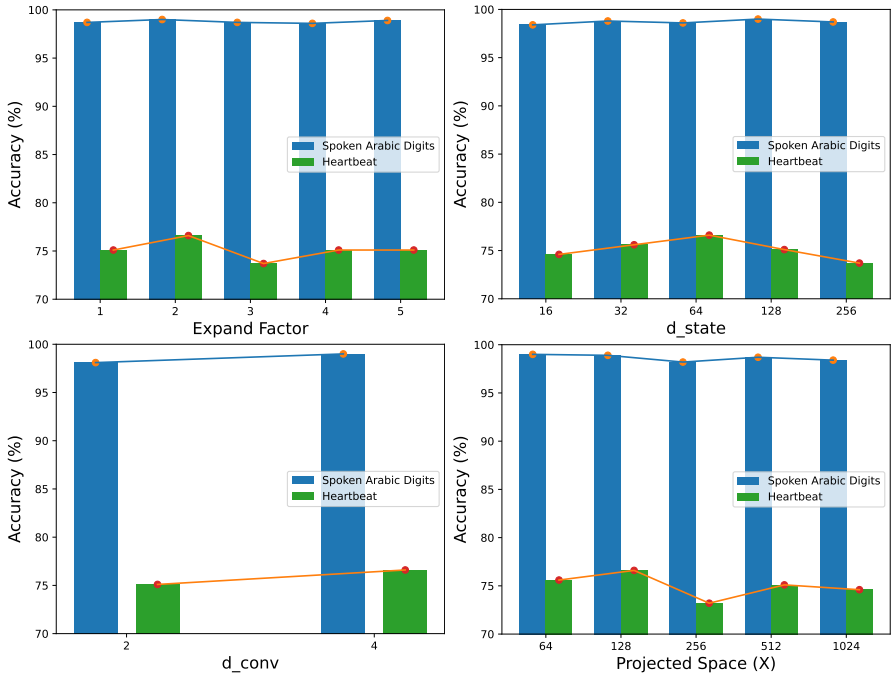

Figure 2: Sensitivity analysis of TSCMamba's hyper-parameters on Time Series Classification (TSC) performance. The plot shows the impact of varying (from left to right, top to bottom) block expansion factor, SSM state expansion factor, local convolutional width, and dimension of the projected space (X) on model performance, highlighting the relative importance of each component in achieving optimal TSC results.

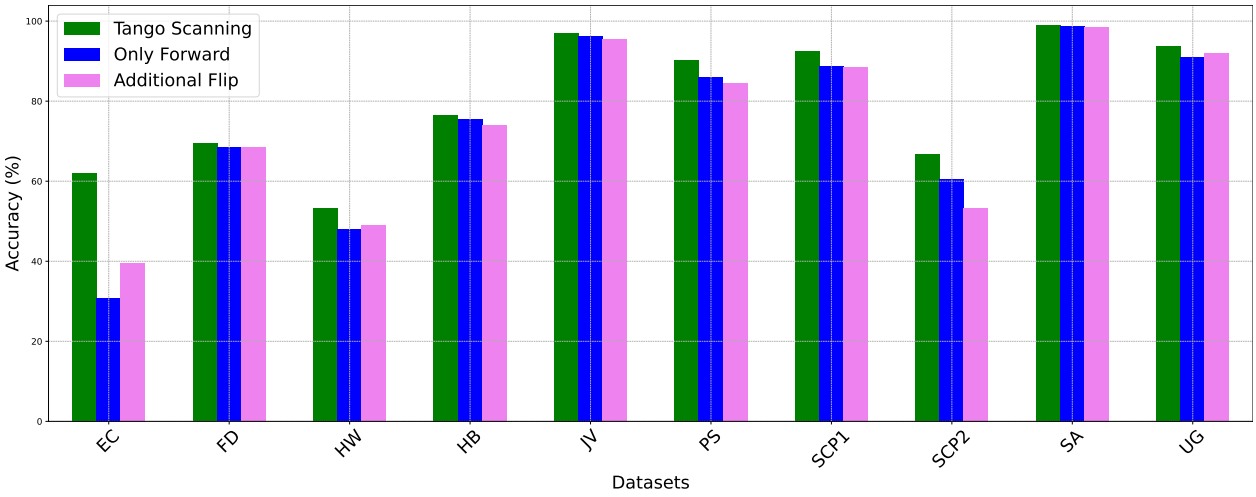

Figure 3: Effectiveness of our tango scanning compared against only forward-based scanning protocol and additional flip-based scanning protocol in reverse scanning.

to establish its versatility and robustness in various real-world scenarios. The promising results suggest TSCMamba's high potential for time series applications.

References

Md Atik Ahamed and Qiang Cheng. MambaTab: A simple yet effective approach for handling tabular data. *arXiv preprint arXiv:2401.08867*, 2024a.

Md Atik Ahamed and Qiang Cheng. TimeMachine: A time series is worth 4 mambas for long-term forecasting. *arXiv preprint arXiv:2403.09898*, 2024b.

Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. The UEA multivariate time series classification archive, 2018. *arXiv preprint arXiv:1811.00075*, 2018.

Ali Behrouz and Farnoosh Hashemi. Graph Mamba: Towards learning on graphs with state space models. *arXiv preprint arXiv:2402.08678*, 2024.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

Donald J. Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *KDD Workshop*, 1994.

Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. *KDD*, 2016.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *International Conference on Machine Learning (ICML)*, 2024.

Angus Dempster, Franccois Petitjean, and Geoffrey I. Webb. Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Min. Knowl. Discov.*, 2020.

Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11, 2018.

Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. Unsupervised scalable representation learning for multivariate time series. In *NeurIPS*, 2019.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Albert Gu, Isys Johnson, Karan Goel, Khaled Saab, Tri Dao, Atri Rudra, and Christopher Ré. Combining recurrent, convolutional, and continuous-time models with linear state-space layers. *NeurIPS*, 2021.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *ICLR*, 2022.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 1997.

Yanfei Kang, Danijel Belušić, and Kate Smith-Miles. Detecting and classifying events in noisy time series. *Journal of the Atmospheric Sciences*, 71(3):1090–1104, 2014.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Franz Király, Markus Löning, Tony Bagnall, Matthew Middlehurst, Sajaysurya Ganesh, Martin Walter, George Oastler, Anirban Ray, Jason Lines, ViktorKaz, Lukasz Mentel, Benedikt Heidrich, Sagar Mishra, chrisholder, Daniel Bartling, Leonidas Tsaprounis, RNKuhns, Ciaran Gilbert, Mirae Baichoo, Hazrul Akmal, Patrick Rockenschaub, Taiwo Owoseni, Guzal, eenticott shell, Sami Alavi, jesellier, Armaghan, Kishan Manani, and Patrick Schäfer. sktime/sktime: v0.30.0, June 2024. URL `https://doi.org/10.5281/zenodo.11460888`.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *ICLR*, 2020.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *SIGIR*, 2018.

Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X Liu, and Schahram Dustdar. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *ICLR*, 2021.

Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary transformers: Rethinking the stationarity in time series forecasting. In *NeurIPS*, 2022.

Markus Löning, Anthony Bagnall, Sajaysurya Ganesh, Viktor Kazakov, Jason Lines, and Franz J Király. sktime: A unified interface for machine learning with time series. *arXiv preprint arXiv:1909.07872*, 2019.

Kaleel Mahmood, Rigel Mahmood, and Marten Van Dijk. On the robustness of vision transformers to adversarial examples. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7838–7847, 2021.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32, 2019.

Chong Peng, Kehan Kang, Yongyong Chen, Zhao Kang, Chenglizhao Chen, and Qiang Cheng. Fine-grained essential tensor learning for robust multi-view spectral clustering. *IEEE Transactions on Image Processing*, 33:3145–3160, 2024.

Yair Schiff, Chia-Hsiang Kao, Aaron Gokaslan, Tri Dao, Albert Gu, and Volodymyr Kuleshov. Caduceus: Bi-directional equivariant long-range dna sequence modeling. *arXiv preprint arXiv:2403.03234*, 2024.

Siavash Shams, Sukru Samet Dindar, Xilin Jiang, and Nima Mesgarani. Ssamba: Self-supervised audio representation learning with mamba state space model. *arXiv preprint arXiv:2405.11831*, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

Zhenya Wang, Ligang Yao, and Yongwu Cai. Rolling bearing fault diagnosis using generalized refined composite multiscale sample entropy and optimized support vector machine. *Measurement*, 156:107574, 2020.

Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven C. H. Hoi. ETSformer: Exponential smoothing transformers for time-series forecasting. *arXiv preprint arXiv:2202.01381*, 2022.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with Auto-Correlation for long-term series forecasting. In *NeurIPS*, 2021.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. TimesNet: Temporal 2d-variation modeling for general time series analysis. In *The Eleventh International Conference on Learning Representations*, 2022a.

Haixu Wu, Jialong Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Flowformer: Linearizing transformers with conservation flows. In *ICML*, 2022b.

xiaoju Ye. calflops: a flops and params calculate tool for neural networks in pytorch framework, 2023. URL `https://github.com/MrYxJ/calculate-flops.pytorch`.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 11121–11128, 2023.

T. Zhang, Yizhuo Zhang, Wei Cao, J. Bian, Xiaohan Yi, Shun Zheng, and Jian Li. Less is more: Fast multivariate time series forecasting with light sampling-oriented mlp structures. *arXiv preprint arXiv:2207.01186*, 2022.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *AAAI*, 2021.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *ICML*, 2022.

7 Appendix

In the appendix section, we present additional supplementary materials including algorithms, dataset statistics, etc. To distinguish from original materials, we add the prefix **S-** to the supplementary materials.

7.1 Algorithms

In this section, we present the two schematic algorithms for conversion of raw signals to CWT and ROCKET feature extraction in S-Algorithm 1 and S-Algorithm 2.respectively.

S-Algorithm 1 Convert Raw Signals to CWT representation

Input: Raw signals of shape (N, D, L)
Output: Tensor of shape (N, D, L_1, L_1) # We set $L_1 = 64$ in this paper.

for each signal i in N **do**
  for each dimension d in D **do**
    $signal \leftarrow \text{Raw}[i, d, :]$
    $coeff, freq \leftarrow \text{CWT}(signal)$
    $cwt\_resized \leftarrow \text{resize}(coeff, (L_1, L_1), \text{mode="constant"})$
    $\text{Tensor}[i, d, :, :] \leftarrow cwt\_resized$
  end for
end for

S-Algorithm 2 Feature Extraction with ROCKET for Random Convolutional Kernel Transform

Input: Time series data of length $= L$, number of kernels, $n = \frac{X}{2}$
Output: Feature vector of shape $(2 \times n) = X$

$kernels \leftarrow$ list of n random kernels of random length l, weight w, bias b, dilation d, padding p.
$feature\_maps \leftarrow$ empty list
for each kernel k in $kernels$ **do**
  $h_{ppv}, h_{max} \leftarrow \text{convolve}(k, x)$
  $feature\_maps.\text{append}(h_{ppv}, h_{max})$
end for
return feature_ maps

7.2 Dataset Statistics

We utilized 10 datasets following TimesNet (Wu et al., 2022a). Datasets with their corresponding number of channels (D), sequence length, Train samples, Test samples, number of classes, and domain information are presented in S-Table 1.

S-Table 1: Publicly available datasets with their statistics utilized in this paper.

Datasets	Channels	Length	Train	Test	Classes	Domain
EthanolConcentration (EC)	3	1751	261	263	4	Alcohol Industry
FaceDetection (FD)	144	62	5890	2524	2	Face (250Hz)
Handwriting (HW)	3	152	150	850	26	Smart Watch
Heartbeat (HB)	61	405	204	205	2	Clinical
JapaneseVowels (JV)	12	29	270	370	9	Audio
PEMS-SF (PS)	963	144	267	173	7	Transportation
SelfRegulationSCP1 (SCP1)	6	896	268	293	2	Health (256Hz)
SelfRegulationSCP2 (SCP2)	7	1152	200	180	2	Health (256Hz)
SpokenArabicDigits (SA)	13	93	6599	2199	10	Voice (11025Hz)
UWaveGestureLibrary (UG)	3	315	120	320	8	Gesture

7.3 Accuracy on Multiple Runs

S-Figure 4: Performance of TSCMamba over 5 random runs. The mean performance is shown as green bars, with the standard deviation represented by red error bars that are very small. (Best viewed when zoomed in.)