Towards Diverse Perspective Learning with Selection over Multiple Temporal Poolings

Jihyeon Seong*[1], Jungmin Kim*[1], Jaesik Choi[1, 2]

[1]Korea Advanced Institute of Science and Technology (KAIST), South Korea
[2]INEEJI, South Korea
{jihyeon.seong, aldirl7, jaesik.choi}@kaist.ac.kr

Abstract

In Time Series Classification (TSC), temporal pooling methods that consider sequential information have been proposed. However, we found that each temporal pooling has a distinct mechanism, and can perform better or worse depending on time series data. We term this fixed pooling mechanism a single perspective of temporal poolings. In this paper, we propose a novel temporal pooling method with diverse perspective learning: Selection over Multiple Temporal Poolings (SoM-TP). SoM-TP dynamically selects the optimal temporal pooling among multiple methods for each data by attention. The dynamic pooling selection is motivated by the ensemble concept of Multiple Choice Learning (MCL), which selects the best among multiple outputs. The pooling selection by SoM-TP's attention enables a non-iterative pooling ensemble within a single classifier. Additionally, we define a perspective loss and Diverse Perspective Learning Network (DPLN). The loss works as a regularizer to reflect all the pooling perspectives from DPLN. Our perspective analysis using Layer-wise Relevance Propagation (LRP) reveals the limitation of a single perspective and ultimately demonstrates diverse perspective learning of SoM-TP. We also show that SoM-TP outperforms CNN models based on other temporal poolings and state-of-the-art models in TSC with extensive UCR/UEA repositories.

Introduction

Time Series Classification (TSC) is one of the most valuable tasks in data mining, and Convolutional Neural Network (CNN) with global pooling shows revolutionary success on TSC (Längkvist, Karlsson, and Loutfi 2014; Ismail Fawaz et al. 2019). However, global pooling in TSC poses a significant challenge, as it disregards the fundamental characteristic of time series data, which is the temporal information, by compressing it into a single scalar value (Lecun et al. 1998; Yu et al. 2014). To tackle this issue, temporal pooling methods were introduced, which preserve the temporal nature of the time series at the pooling level (Lee, Lee, and Yu 2021).

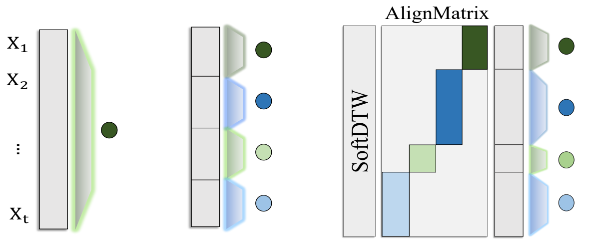
Temporal pooling involves employing operations such as 'maximum' (MAX) and 'average' (AVG), categorized by segmentation types: 'no segment,' 'uniform,' and 'dynamic.' These segmentation types correspond respectively to Global-Temporal-Pooling (GTP), Static-Temporal-Pooling (STP), and Dynamic-Temporal-Pooling (DTP) (Lee, Lee, and Yu 2021). We refer to each distinct pooling mechanism as a *perspective* based on the segmentation types. However, we discovered that the most effective temporal pooling varies depending on the characteristics of the time series data, and there is no universally dominant pooling method for all datasets (Esling and Agon 2012). This underlines the necessity for a learnable pooling approach adaptable to each data sample's characteristics.

In this paper, we propose Selection over Multiple Temporal Poolings (SoM-TP). SoM-TP is a learnable ensemble pooling method that dynamically selects heterogeneous temporal poolings through an attention mechanism (Vaswani et al. 2017). Aligned with our observation that a more suitable pooling exists for each data sample, a simple ensemble weakens the specified representation power (Lee et al. 2017). Therefore, SoM-TP applies advanced ensemble learning, motivated by Multiple Choice Learning (MCL), that selects the best among the multiple pooling outputs (Guzmán-rivera, Batra, and Kohli 2012).

MCL is a selection ensemble that generates M predictions from multiple instances, computes the oracle loss for the most accurate prediction, and optimizes only the best classifier. Capitalizing on the advantage of deep networks having access to intermediate features, SoM-TP ensembles diverse pooling features in a single classifier. To achieve non-iterative optimization, SoM-TP dynamically selects the most suitable pooling method for each data sample through attention, which is optimized by *Diverse Perspective Learning Network (DPLN)* and *perspective loss*. DPLN is a subnetwork that utilizes all pooling outputs, and the perspective loss reflects DPLN's result to make a regularization effect. Finally, the CNN model based on SoM-TP forms fine representations through diverse pooling selection, allowing it to capture both the 'global' and 'local' features of the dataset.

Recognizing the crucial role of pooling in selecting the most representative values from encoded features in CNNs, we have chosen CNNs as the suitable model for our study. We apply our new selection ensemble pooling to Fully Convolutional Networks (FCNs) and Residual Networks (ResNet), which show competitive performances in TSC as a CNN-based model (Wang, Yan, and Oates 2017; Ismail Fawaz et al. 2019). SoM-TP outperforms the exist-

*These authors contributed equally.



(a) GTP (b) STP (c) DTP

Figure 1: Perspectives of Temporal Poolings. Depending on segmentation types, each temporal pooling generates different pooling outputs and has different *perspectives*.

ing temporal pooling methods and state-of-the-art models of TSC both in univariate and multivariate time series datasets from massive UCR/UEA repositories. We also provide a detailed analysis of the diverse perspective learning result by Layer-wise Relevance Propagation (LRP) (Bach et al. 2015) and the dynamic selection process of SoM-TP. To the best of our knowledge, this is the first novel approach to pooling-level ensemble study in TSC.

Therefore, our contributions are as follows:

- We investigate data dependency arising from distinct perspectives of existing temporal poolings.
- We propose SoM-TP, a new temporal pooling method that fully utilizes the diverse temporal pooling mechanisms through an MCL-inspired selection ensemble.
- We employ an attention mechanism to enable a non-iterative ensemble in a single classifier.
- We define DPLN and perspective loss as a regularizer to promote diverse pooling selection.

Background

Different Perspectives between Temporal Poolings

Convolutional Neural Network in TSC In TSC, CNN outperforms conventional methods, such as nearest neighbor classifiers (Yuan et al. 2019) or COTE (Bagnall et al. 2015; Lines, Taylor, and Bagnall 2016), by capturing local patterns of time series (Ismail Fawaz et al. 2019).

The TSC problem is generally formulated as follows: a time series data $\mathbf{T} = \{(\mathbf{X}_1, y_1), ..., (\mathbf{X}_t, y_t)\}$, where $\mathbf{X} \in \mathbb{R}^{\mathbf{d} \times \mathbf{t}}$ of length $\mathbf{t}$ with $\mathbf{d}$ variables and $y \in \{1, ..., C\}$ from C classes. Then the convolution stack Φ of out channel dimension $\mathbf{k}$ encodes features as hidden representations with temporal position information $\mathbf{H} = \{h_0, ..., h_t\} \in \mathbb{R}^{\mathbf{k} \times \mathbf{t}}$ (Lee, Lee, and Yu 2021; Wang, Yan, and Oates 2017).

$$\mathbf{H} = \Phi(\mathbf{T}) \tag{1}$$

After convolutional layers, global pooling plays a key role with two primary purposes: 1) reducing the number of parameters for computational efficiency and preventing overfitting, and 2) learning position invariance. For this purpose, pooling combines the high-dimensional feature outputs into low-dimensional representations (Gholamalinezhad and Khosravi 2020). However, global pooling presents an issue of losing temporal information, which has led to the development of temporal pooling methods (Lee, Lee, and Yu 2021). We investigate different mechanisms of temporal poolings, which we refer to as *perspective*.

Global Temporal Pooling GTP pools only one representation $\mathbf{p_g} = [p_1] \in \mathbb{R}^{\mathbf{k} \times \mathbf{1}}$ in the entire time range. GTP ignores temporal information by aggregating the $\mathbf{H}$ to $\mathbf{p}_g = h$: the global view.

$$\mathbf{p}_g = pool_g(\mathbf{H}) \tag{2}$$

GTP effectively captures globally dominant features, such as trends or the highest peak, but has difficulty capturing multiple points dispersed on a time axis. To solve this constraint, temporal poolings based on sequential segmentation have been proposed: STP and DTP (Lee, Lee, and Yu 2021). Both have multiple local segments with the given number $\mathbf{n} \in \mathbb{Z}^+$: the local view.

Static Temporal Pooling STP divides the time axis equally into $\mathbf{n}$ segments with a length $\ell = \frac{t}{n}$, where $\bar{\mathbf{H}} = \{\mathbf{h}_{\mathbf{0}:\ell}, \mathbf{h}_{\ell:\mathbf{2}\ell}, ..., \mathbf{h}_{(\mathbf{n-1})\ell:\mathbf{n}\ell}\}$ and $\mathbf{p_s} = [p_1, ..., p_n] \in \mathbb{R}^{\mathbf{k} \times \mathbf{n}}$. Note that $\mathbf{h}_\ell$ retains temporal information, but there is no consideration of the temporal relationship between time series in the segmentation process: the uniform local view.

$$\mathbf{p}_s = pool_s(\bar{\mathbf{H}}) \tag{3}$$

STP functions well on a recursive pattern, such as a stationary process. However, forced uniform segmentation can divide important consecutive patterns or create unimportant segmentations. This inefficiency causes representation power to be distributed to non-informative regions.

Dynamic Temporal Pooling DTP is a learnable pooling layer optimized by soft-DTW (Cuturi and Blondel 2017) for dynamic segmentation considering the temporal relationship. By using the soft-DTW layer, $\mathbf{H}$ is segmented in diverse time lengths $\bar{\ell} = [\ell_1, \ell_2, ..., \ell_n]$, where $\mathbf{t} = \sum \bar{\ell}$. Finally, the optimal pooled vectors $\mathbf{p}_d = [p_{\ell_1}, ..., p_{\ell_n}] \in \mathbb{R}^{\mathbf{k} \times \mathbf{n}}$ are extracted from each segment of $\bar{\mathbf{H}}_{\bar{\ell}}$, where $\bar{\mathbf{H}}_{\bar{\ell}} = \{\mathbf{h}_{\ell_1}, \mathbf{h}_{\ell_2}, ..., \mathbf{h}_{\ell_n}\}$; the dynamic local view.

$$\mathbf{p}_d = pool_d(\bar{\mathbf{H}}_{\bar{\ell}}) \tag{4}$$

DTP has the highest complexity in finding different optimal segmentation lengths, enabling the pooling to fully represent segmentation power. However, since DTP is based on temporally aligned similarity of hidden features with a constraint that a single time point should not be aligned with multiple consecutive segments, the segmentation can easily divide informative change points that need to be preserved in time series patterns (Appendix. DTP Algorithm).

Limitation of Single Perspective Traditional temporal pooling methods only focus on a single perspective when dealing with hidden features $\mathbf{H}$. A global perspective cannot effectively capture multiple classification points, while a local perspective struggles to emphasize a dominant classification point. Consequently, datasets that require the simultaneous capture of dominant and hidden local features from diverse viewpoints inevitably exhibit lower performance when using a single perspective. Motivated by these limitations, we propose a novel pooling approach that fully leverages diverse perspectives.

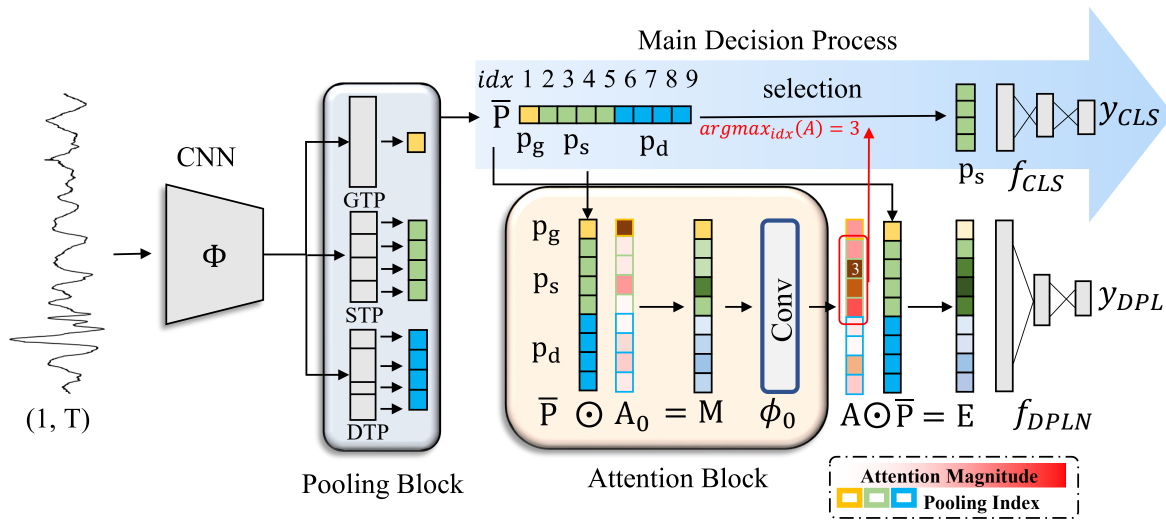


Figure 2: SoM-TP Architecture. Diverse Perspective Learning based on selection-ensemble is achieved as follows: The aggregated output of all pooling, $\bar{\mathbf{P}}$ is passed to the attention block to calculate the attention score $\mathbf{A}$. In the attention block, a weighted pooling output $\mathbf{M}$ is formed by the multiplication of $\bar{\mathbf{P}}$ and a learnable weight vector $\mathbf{A_0}$. After $\mathbf{M}$ passes through the convolutional layer ϕ_0, the attention score $\mathbf{A}$ is drawn out as an encoded weight vector. Using the index of the highest attention score (here, index 3), pooling for the CLS network is selected. Next, the parameters are updated with the following procedure: 1) DPLN uses the ensembled vector $\mathbf{E}$, whereas CLS network uses only the selected pooling output (here, $\mathbf{p_s}$); 2) Each network predicts y_{CLS} and y_{DPL} respectively, and y_{DPL} is used in the perspective loss to work as a regularizer; 3) With these two outputs, the model is optimized with diverse perspectives while selecting the proper pooling method for each batch.

Multiple Choice Learning for Deep Temporal Pooling

The traditional ML-based ensemble method focuses on aggregating multiple outputs. However, the aggregation of the simple ensemble makes outputs smoother, due to the generalization effect (Lee et al. 2017). To solve this limitation, MCL has been proposed as an advanced ensemble method that selects the best among multiple outputs using oracle loss (Guzmán-rivera, Batra, and Kohli 2012). More formally, MCL generates M solutions $\hat{Y}_i = (\hat{y}_i^1, ..., \hat{y}_i^M)$, and learns a mapping $g : \mathcal{X} \rightarrow \mathcal{Y}^M$ that minimizes oracle loss $min_m \ell(y_i, \hat{y}_i^m)$. The ensemble mapping function g consists of multiple predictors, $g(x) = \{f_1(x), f_2(x), ..., f_M(x)\}$ (Lee et al. 2016).

The effects of diverse solution sets in MCL can be summarized as addressing situations of 'Ambiguous evidence' and 'Bias towards the mode'. 'Ambiguous evidence' refers to situations with insufficient information to make a definitive prediction. In such cases, presenting a small set of reasonable possibilities can alleviate the over-confidence problem of deep learning (Nguyen, Yosinski, and Clune 2015), rather than striving for a single accurate answer. The other situation is 'Bias towards the mode', indicating the model's tendency to learn a mode-seeking behavior to reduce the expected loss across the entire dataset. When only a single prediction exists, the model eventually learns to minimize the average error. In contrast, MCL generates multiple predictions, allowing some classifiers to cover the lower-density regions of the solution space without sacrificing performance on the high-density regions (Lee et al. 2016).

MCL faces computational challenges in deep networks due to the iterative optimization process of the oracle loss, with a complexity of $\mathcal{O}(N^2)$. Although sMCL has partially alleviated this issue through stochastic gradient descent, the method still requires identifying the best output among multiple possibilities (Lee et al. 2016). CMCL, which is another approach to address MCL's over-confidence problem, cannot be applied at the feature level due to optimization at the output level (Lee et al. 2017). In summary, the integration of MCL into the pooling level is not feasible due to the structural constraints imposed by the oracle loss design. To overcome this challenge, we establish a model structure to incorporate the concept of MCL into a pooling-level ensemble.

Selection over Multiple Temporal Pooling

SoM-TP Architecture and Selection Ensemble

Diverse Perspective Learning (DPL) is achieved by dynamically selecting heterogeneous multiple temporal poolings in a single classifier. The overview architecture, as illustrated in Figure 2, consists of four parts: 1) a common feature extractor with CNN Φ; 2) a pooling block with multiple temporal pooling layers; 3) an attention block with an attention weight vector $\mathbf{A}_0$, and a convolutional layer ϕ_0, as well as 4) Fully Connected layers (FC): a classification network (CLS) f_{CLS} and a *DPLN* f_{DPLN}. Through these modules, SoM-TP can cover the high probability prediction space, which is aligned with MCL where multiple classifiers are trained to distinguish specific distributions (Lee et al. 2016).

DPL Attention SoM-TP ensembles multiple temporal pooling within a single classifier. The advantage of using a single classifier lies in the absence of the need to compare prediction outputs, making it non-iterative and computationally efficient, in contrast to MCL. Through the attention mechanism (Vaswani et al. 2017), we achieve a 'comparison-free' ensemble by dynamically selecting the most suitable temporal pooling for each batch.

DPL attention is an extended attention mechanism that simultaneously considers two factors: the overall dataset and each data sample. In Figure 2, the attention weight vector $\mathbf{A}_0$ is learned in the direction of adding weight to specific pooling, which has a minimum loss for every batch. Consequently, $\mathbf{A}_0$ has a weight that reflects the entire dataset.

Next, a convolutional layer ϕ_0 is used to reflect a more specific data level. As shown in Algorithm 1, the weighted pooling vector $\mathbf{M}$ which is the element-wise multiplication between $\mathbf{A}_0$ and $\bar{\mathbf{P}}$ is given as an input to ϕ_0. By encoding $\mathbf{M}$ through ϕ_0, $\mathbf{A}$ can reflect more of each batch characteristic, not just following the dominant pooling in terms of the entire dataset. As a result, this optimized pooling selection by attention can solve the 'Bias towards the mode' problem by assigning the most suitable pooling for each data batch, which is aligned with learning the multiple experts in MCL.

Algorithm 1: SoM-TP selecting algorithm

Function `Attention Block(`$\mathbf{H}$`):`
- ▷ select proper temporal pooling by attention $\mathbf{A} \in \mathbb{R}^{1\times 3\mathbf{n}}$
- ▷ attention weight $\mathbf{A}_0 \in \mathbb{R}^{1\times 3\mathbf{n}}$ is initialized as zero
- ▷ GTP, STP, DTP: $pool_g, pool_s, pool_d$
- ▷ convolutional encoding layer: ϕ_0
- **Function** `Pooling Block(`$\mathbf{H}$`):`
  - ▷ convolutional hidden feature: $\mathbf{H} \in \mathbb{R}^{k\times t}$
  - ▷ static segmented hidden feature: $\bar{\mathbf{H}} = \{\mathbf{h}_{\mathbf{0}:\ell}, \mathbf{h}_{\ell:\mathbf{2}\ell}, ..., \mathbf{h}_{(\mathbf{n-1})\ell:\mathbf{n}\ell}\}$, $\ell = \frac{t}{n}$
  - ▷ dynamic segmented hidden feature: $\bar{\mathbf{H}}_{\bar{\ell}} = \{\mathbf{h}_{\ell_1}, \mathbf{h}_{\ell_2}, ..., \mathbf{h}_{\ell_n}\}$, $\bar{\ell} = [\ell_1, \ell_2, ..., \ell_n]$, where $\mathbf{t} = \sum \bar{\ell}$
  - ▷ pooling outputs: $\mathbf{p}_g, \mathbf{p}_s, \mathbf{p}_d = pool_g(\mathbf{H}), pool_s(\bar{\mathbf{H}}), pool_d(\bar{\mathbf{H}}_{\bar{\ell}})$
  - **return** $\mathbf{p}_g, \mathbf{p}_s, \mathbf{p}_d$
- $\bar{\mathbf{P}} = [\mathbf{p}_g, \mathbf{p}_s, \mathbf{p}_d]$
- $\mathbf{M} = \mathbf{A}_0 \odot \bar{\mathbf{P}}$
- $\mathbf{A} = \phi_0(\mathbf{M})$, where $x \in \mathbf{A}$
- $idx = \begin{cases} \text{argmax}_i(y), \text{ where } y_i = \frac{exp(x_i)}{\sum_i^{\mathbf{n}} exp(x_i)} \\ \text{argmax}_j(y), \text{ where } y_j = \frac{\sum_j x_{(j|n)}}{\mathbf{n}} \end{cases}$
- **return** $\mathbf{p} = \bar{\mathbf{P}}(idx), \mathbf{E} = \mathbf{A} \odot \bar{\mathbf{P}}$

Diverse Perspective Learning Network and Perspective Loss

To optimize DPL attention, we introduce DPLN and perspective loss. DPLN is a sub-network that utilizes the weighted aggregation ensemble $\mathbf{E}$. The main role of DPLN is regularization through perspective loss. In contrast, the CLS network, which predicts the main decision, does not directly utilize attention $\mathbf{A}$. Instead, it employs a chosen pooling feature output (denoted as $\mathbf{p_s}$ in Figure 2), determined by the index with the biggest score in $\mathbf{A}$.

Perspective Loss Perspective loss serves as a cost function to maximize the utilization of the sub-network, DPLN, through network tying between the two FC networks. Ultimately, it aims to prevent the CLS network from converging to one dominant pooling and to maintain the benefits of ensemble learning continuously.

To achieve its purpose, perspective loss is designed as the sum of DPLN cross-entropy loss and the Kullback-Leibler (KL) divergence between y_{CLS} and y_{DPL}. Specifically, KL divergence works similarly to CMCL's KL term, which regulates one model to be over-confident through a uniform distribution, while the KL term of perspective loss regulates based on DPLN (Lee et al. 2017).

$$\begin{aligned} &KL(y_{CLS}, y_{DPL}) = y_{DPL} \cdot log\frac{y_{DPL}}{y_{CLS}}, \\ &\mathcal{L}_{DPLN}(\{\mathcal{W}_\Phi\}, \{\mathbf{W}^{(dpln)}\}) = -\frac{1}{t}\sum_{n=1}^{t} logP(y = y_n|\mathbf{X}_n), \\ &\mathcal{L}_{perspective} = KL(y_{CLS}, y_{DPL}) + \mathcal{L}_{DPLN}, \end{aligned} \tag{5}$$

where input time series $\{(\mathbf{X}_1, y_1), ..., (\mathbf{X_t}, y_t)\}$, Φ with learnable parameter $\mathcal{W}_\Phi$ of CNN, $y_{CLS} \in \mathbb{R}^{1\times c}$ from the 'CLS network' $\mathbf{W}^{(cls)}$, and $y_{DPL} \in \mathbb{R}^{1\times c}$ from the DPLN $\mathbf{W}^{(dpln)}$. Then, we set first f_{DPLN} weight matrix $\mathbf{W}_0^{(dpln)} = [\mathbf{w}_1^{(\mathbf{p}_g)}, ..., \mathbf{w}_{\mathbf{2n}}^{(\mathbf{p}_s)}, ..., \mathbf{w}_{\mathbf{3n}}^{(\mathbf{p}_d)}] \in \mathbb{R}^{k\times 3\cdot\mathbf{n}}$, where $\mathbf{w}^{(\mathbf{p})} \in \mathbb{R}^k$ is weight matrix of each latent dimension k of pooling $\mathbf{p}_i$, whereas $\mathbf{W}_0^{(cls)} = [\mathbf{w}_1^{(c)}, ..., \mathbf{w}_{\mathbf{n}}^{(c)}] \in \mathbb{R}^{k\times\mathbf{n}}$ is the first f_{CLS} weight matrix (Lee, Lee, and Yu 2021). Note that the results of GTP are repeated $\mathbf{n}$ times to give the same proportion for each pooling by attention weight.

Therefore, the final loss function of the SoM-TP is designed as follows,

$$\begin{aligned} &\mathcal{L}_{CLS}(\{\mathcal{W}_\Phi\}, \{\mathbf{W}^{(cls)}\}) = -\frac{1}{t}\sum_{n=1}^{t} logP(y = y_n|\mathbf{X}_n), \\ &\mathcal{L}_{cost}(\{\mathcal{W}_\Phi\}, \{\mathbf{W}\}) = \mathcal{L}_{CLS} + \lambda \cdot \mathcal{L}_{perspective}, \end{aligned} \tag{6}$$

where $\{\mathbf{W}^{(cls)}, \mathbf{W}^{(dpln)}, \mathbf{A}_0, \phi_0\} \in \mathbf{W}$ are learnable parameters. Prioritizing classification accuracy, the loss $\mathcal{L}_{CLS}$ is computed, and $\mathcal{L}_{perspective}$ is added with λ decay.

As a result, SoM-TP can address the 'Ambiguous evidence' problem through DPLN and perspective loss. In a pooling ensemble, the 'Ambiguous evidence' can be conceived as a scenario where a single pooling is not dominant. Even though SoM-TP selects only one pooling, y_{DPL} in the perspective loss enables the model to consider the importance of other poolings.

CNN	POOL (type)		UCR (uni-variate)					UEA (multi-variate)				
	MAX		ACC	F1 macro	ROC AUC	PR AUC	Rank	ACC	F1 macro	ROC AUC	PR AUC	Rank
FCN	GTP		0.6992	0.6666	0.8662	0.7406	3.6	0.6558	0.6213	0.7841	0.6854	3.9
	STP		0.7462	0.7133	0.8924	0.7889	2.7	0.6801	0.6603	0.8001	0.6984	3.0
	DTP	euc	0.7406	0.7123	0.8897	0.7782	3.1	0.6795	0.6559	0.8129	0.7163	3.4
		cos	0.7335	0.7062	0.8879	0.7768	2.9	0.6702	0.6314	0.8061	0.7022	3.1
	SoM-TP		**0.7556**	**0.7241**	**0.9026**	**0.8000**	**2.6**	**0.6920**	**0.6621**	0.8105	0.7099	**2.4**
ResNet	GTP		0.7227	0.6952	0.8837	0.7654	3.5	0.6423	0.6083	0.7798	0.6769	3.5
	STP		0.7420	0.7126	0.8880	0.7864	2.9	0.6717	0.6383	0.7962	0.6934	2.8
	DTP	euc	0.7456	0.7197	0.8939	0.7846	3.0	0.6567	0.6271	0.7981	0.6968	3.1
		cos	0.7452	0.7198	0.8945	0.7829	3.0	0.6534	0.6377	0.7895	0.6832	3.0
	SoM-TP		**0.7773**	**0.7489**	**0.9182**	**0.8261**	**2.4**	**0.6769**	**0.6387**	**0.8016**	**0.7033**	**2.5**

*This table is for pooling type MAX. Please refer to Table 7 in Appendix for pooling type AVG.

Table 1: SoM-TP Comparison with Single Perspective Temporal Poolings. The table presents the effectiveness of the selection ensemble of SoM-TP compared to traditional temporal poolings. The best performances where SoM-TP outperforms others are bolded and the best performances of other temporal poolings are underlined.

Optimization

For attention weight $\mathbf{A}_0$, SoM-TP proceeds with additional optimization: a dot product similarity to regulate $\mathbf{A}_0$. The similarity term is defined as,

$$\mathcal{L}_{attn} = -y_{CLS} \cdot y_{DPL}, \tag{7}$$

Due to the KL-divergence cost function in perspective loss, the CLS network and DPLN can be overly similar during the optimization process. As the additional regulation for output over-similarity, $\mathcal{L}_{attn}$ plays the opposite regulation to the perspective loss. Note that the dot-product similarity considers both the magnitude and direction of two output vectors.

Finally, the overall optimization process is as follows,

$$\begin{aligned} &\mathbf{A_0} \leftarrow \mathbf{A_0} - \eta \cdot \partial \mathcal{L}_{attn} / \partial \mathbf{A_0}, \\ &\mathcal{W}_\Phi \leftarrow \mathcal{W}_\Phi - \eta \cdot \partial \mathcal{L}_{cost} / \partial \mathcal{W}_\Phi, \\ &\mathbf{W} \leftarrow \mathbf{W} - \eta \cdot \partial \mathcal{L}_{cost} / \partial \mathbf{W}. \end{aligned} \tag{8}$$

As a result, even with a non-iterative optimization process, SoM-TP learns various perspectives through DPL attention, DPLN, and perspective loss. Consequently, Φ reflects f_{CLS} and f_{DPLN} relatively while minimizing the similarity between each network output.

Experiments

Experimental Settings

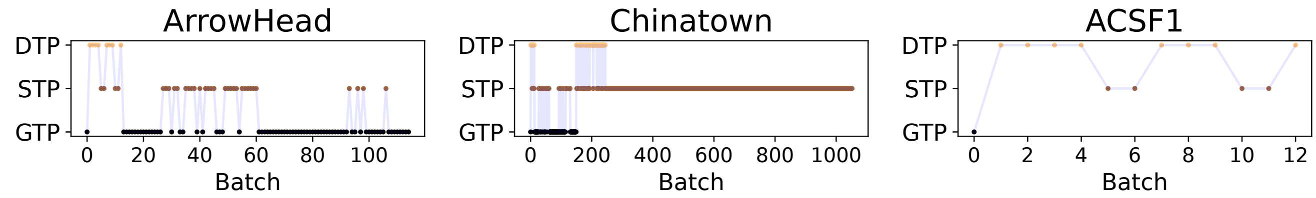
For the extensive evaluation, 112 univariate and 22 multivariate time series datasets from the UCR/UEA repositories are used (Bagnall et al. 2018; Dau et al. 2019); collected from a wide range of domains and publicly available. To ensure the validity of our experiments, we exclude a few datasets from the UCR/UEA repositories due to the irregular data lengths. While zero padding could resolve this, it might cause bias in some time series models.

All temporal pooling methods have the same CNN architecture. FCN and ResNet are specifically designed as a feature extractor (Wang, Yan, and Oates 2017), and temporal poolings are constructed with the same settings: normalization with BatchNorm (Ioffe and Szegedy 2015), activation function with ReLU, and optimizer with Adam (Kingma and Ba 2015). The validation set is made from 20% of the training set for a more accurate evaluation. In the case of imbalanced classes, a weighted loss is employed. The prototype number $\mathbf{n}$ is searched in a greedy way, taking into consideration the unique class count of each dataset. Specifically, we observe that selecting 4-10 segments based on the class count in each dataset enhances performance. Consequently, we use an equal number of segments in each dataset for segment-based poolings (Appendix. Table 5).

Baselines We conduct two experiments to evaluate the performance of SoM-TP. First, we compare it with traditional temporal poolings, GTP, STP, and DTP, to demonstrate the effectiveness of selection-ensemble in temporal poolings (Lee, Lee, and Yu 2021). Second, we compare SoM-TP with other state-of-the-art models that utilize advanced methods, including scale-invariant methods (ROCKET (Dempster, Petitjean, and Webb 2020), InceptionTime (Ismail Fawaz et al. 2020), OS-CNN (Tang et al. 2021), and DSN (Xiao et al. 2022)), sequential models (MLSTM-FCN (Karim et al. 2019), and TCN (Bai, Kolter, and Koltun 2018)), and Transformer-based models (Vanilla-Transformer (Vaswani et al. 2017), TST (Zerveas et al. 2021), and ConvTran (Foumani et al. 2024)). Models leveraging temporal information use attention or RNNs to emphasize long-term dependencies. On the other hand, scale-invariant learning models employ a CNN-based architecture with various kernel sizes to find the optimum through global average pooling.

Experimental Evaluation

Performance Analysis As shown in Table 1, SoM-TP shows superior performance for overall TSC datasets when compared to conventional temporal poolings. We calculate the average performance of the entire repository. We consider not only accuracy but also the F1 macro score, ROC-AUC, and PR-AUC to consider the imbalanced class. Quantitatively, SoM-TP outperforms the existing temporal pooling methods both in univariate and multivariate time series datasets. Through these results, we can confirm that the dynamic selection ensemble of SoM-TP boosts the performance of the CNN model.



(a) GTP most selected (b) STP most selected (c) DTP most selected

Figure 3: Dynamic Pooling Selection in SoM-TP. This figure represents the graph of dynamic selection in the FCN SoM-TP MAX on the UCR repository: ArrowHead, Chinatown, and ACSF1.

Methods	UCR				UEA			
	Baseline	SoM-TP wins	Tie	Rank	Baseline	SoM-TP wins	Tie	Rank
Vanilla-Transformer (Vaswani et al. 2017)	10	**99**	4	5.2	3	**18**	1	4.4
TCN (Bai, Kolter, and Koltun 2018)	28	**80**	5	4.0	4	**17**	1	4.3
TST (Zerveas et al. 2021)	32	**78**	3	3.8	6	**15**	1	3.6
ConvTran (Foumani et al. 2024)	35	**71**	7	3.1	8	**13**	1	3.1
MLSTM-FCN (Karim et al. 2019)	46	**60**	6	2.6	6	**12**	4	3.2
SoM-TP - MAX	-	-	-	**2.5**	-	-	-	**2.5**

Methods	UCR				UEA			
	Acc	F1-score	ROC AUC	PR AUC	Acc	F1-score	ROC AUC	PR AUC
ROCKET (Dempster, Petitjean, and Webb 2020)	0.7718	0.7478	0.8899	0.7841	0.6785	0.6592	0.7926	0.6940
InceptionTime (Ismail Fawaz et al. 2020)	0.7713	0.7455	0.9056	0.8164	0.6612	0.6360	0.7984	0.7106
OS-CNN (Tang et al. 2021)	0.7663	0.7324	0.9005	0.8139	0.6808	0.6547	**0.8118**	**0.7137**
DSN (Xiao et al. 2022)	0.7488	0.7230	0.8838	0.7968	0.5648	0.5433	0.7575	0.6265
SoM-TP - MAX	**0.7773**	**0.7489**	**0.9182**	**0.8261**	**0.6920**	**0.6621**	0.8105	0.7099

Table 2: SoM-TP Comparison with Advanced TSC Methods. This table compares the performance of SoM-TP with advanced TSC models that leverage temporal information and those that exploit scale-invariant properties, respectively. The best performances, where SoM-TP beat others, are bolded, and the best performances among other models are underlined.

Type	SoM-TP Modules				Rank			Acc
	$\mathbf{A}_0$	ϕ_0	DPLN	$\mathcal{L}_{attn}$	1	2	3	
only ϕ_0		✓			8	15	17	0.6966
only $\mathbf{A}_0$	✓				4	9	24	0.6963
DPL Attention	✓	✓			6	7	23	0.6974
DPLN w/o $\mathbf{A}_0$		✓	✓		6	14	29	0.7047
DPLN	✓	✓	✓		23	27	26	0.7399
SoM-TP	✓	✓	✓	✓	**42**	20	6	**0.7503**

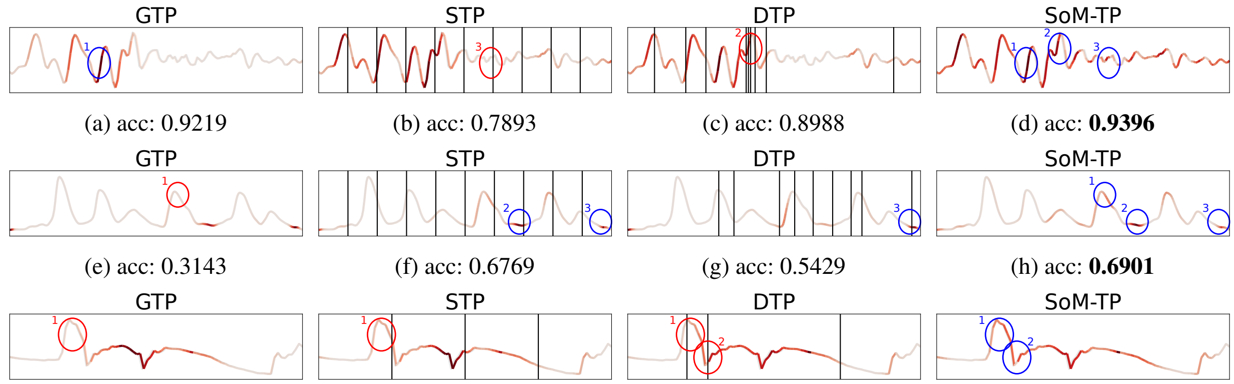
Table 3: SoM-TP Module Ablation Study.

Additionally, Table 2 compares SoM-TP with other state-of-the-art TSC models from two different approaches. In Table 2-1, ResNet SoM-TP MAX significantly outperforms other sequential models in terms of comparing models leveraging temporal information. As SoM-TP clearly outperforms all other models in accuracy metric, we demonstrate the robustness of performance by providing the number of datasets where SoM-TP achieves higher accuracy. Considering the lowest average rank of SoM-TP, we can conclude that dynamic pooling selection leverages the model to keep important temporal information in a more optimal way than other methods in the massive UCR/UEA repository.

Next, Table 2-2 highlights SoM-TP's comparable performance alongside scale-invariant methods, even with SoM-TP's significant computational efficiency. Since SoM-TP and scale-invariant methods have different learning approaches, it is suitable to consider various metrics. Regarding the time complexity of models, scale-invariant methods consider various receptive fields of a CNN, which results in longer training times. In contrast, SoM-TP achieves comparable performance with only one-third of the time.

Finally, in Table 3, we present the results of an ablation study on the modules of SoM-TP discussed in Section 3. When each module, including $\mathbf{A}_0$ and ϕ_0 constituting DPL Attention, DPLN, and $\mathcal{L}_{attn}$, is removed, it decreases SoM-TP's performance. In terms of dataset robustness, we can observe through the rank results that all modules contribute to promoting SoM-TP's Diverse Perspective Learning.

Perspective Analysis with LRP In Figure 3, we can observe that SoM-TP dynamically selects pooling during inference. ArrowHead, Chaintown, and ACSF1, in order, are datasets where GTP, STP, and DTP pooling selections are most frequently chosen. The DPL attention is trained to select the optimal pooling for each batch during the training process, and during inference, it continues to choose the most suitable pooling without DPLN (Appendix. Figure 7).



(a) acc: 0.9219 (b) acc: 0.7893 (c) acc: 0.8988 (d) acc: **0.9396**

(e) acc: 0.3143 (f) acc: 0.6769 (g) acc: 0.5429 (h) acc: **0.6901**

(i) acc: 0.3914 (j) acc: 0.8139 (k) acc: 0.7780 (l) acc: **0.8810**

Figure 4: Comparison of LRP Input Attribution on Single vs Diverse Perspective Learning. The figure shows LRP attribution results for FaceAll, FiftyWords, and MoteStrain datasets in the UCR repository, ordered in respective rows. Pooling choice significantly affects accuracy and attributions, reflecting different perspectives. Redder areas of time series indicate higher attribution, aligning with LRP's conservation rule of summing to 1. Blue circles denote well-captured regions, while red circles suggest dispersed focus or inadequate capture. Given the absence of a ground truth concept for input attributes in TSC, we infer these implicitly from the presented accuracy.

Model	Complexity	
	Pooling	Optimization
GTP	$\mathcal{O}(1)$	$\mathcal{O}(N)$
STP/DTP	$\mathcal{O}(L)$	
SoM-TP	$\mathcal{O}(L_{\mathbf{P}}) + \mathcal{O}(L_{mul}) = \mathcal{O}(L)$	
MCL	-	$\mathcal{O}(N^2)$

Table 4: Complexity Study.

For qualitative analysis, we employ Layer-wise Relevance Propagation (LRP) to understand how different temporal pooling perspectives capture time series patterns. LRP attributes relevance to input features, signifying their contribution to the output. Note that the conservation rule maintains relevance sum in backward propagation, ensuring that the sum of attribution is 1. We use the LRP z^+ rule for the convolutional stack Φ, and the ϵ rule for the FC layers.

In Figure 4, GTP focuses on globally crucial parts (a, e, i), while STP and DTP use local views within segmented time series for a more balanced representation (b, c, f, g, j, k). However, GTP's limitation lies in concentrating only on specific parts, neglecting other local aspects (e, i). Conversely, STP and DTP risk diluting the primary representation by reflecting all local segments (b, j) or cutting significant series due to forced segmentation (c, k). DTP often segments at change points, losing essential information (c, k).

SoM-TP addresses these issues by combining global and local views of each pooling method via diverse perspective learning. In Figure 4-(d), SoM-TP captures GTP's points (d-1) and enhances multiple representations by effectively capturing local patterns (d-2, d-3). In (h), SoM-TP identifies the common important points (h-2, h-3) and complements GTP's missed local points (h-1). Finally, in (l), SoM-TP captures GTP's missed local points (l-1) and fully utilizes important time series (l-2) cut by STP and DTP (j-1, k-1, k-2).

Complexity of SoM-TP We compare the complexity of independent temporal poolings: pooling and optimization complexity. We exclude the maximum or average operation, which is common for all pooling complexity.

As shown in Table 4, for the pooling complexity, GTP has $\mathcal{O}(1)$ while STP and DTP have $\mathcal{O}(L)$ from segmenting. SoM-TP has increased complexity as $\mathcal{O}(L_{\mathbf{P}} + L_{mul}) = \mathcal{O}(L)$ for computation of the attention score, where $\mathcal{O}(L_{\mathbf{P}})$ is for a sum of the three temporal poolings' complexity, making it $\mathcal{O}(L)$, and $\mathcal{O}(L_{mul})$ for the complexity of multiplication between $\bar{\mathbf{P}}$ and $\mathbf{A_0}$, and between $\bar{\mathbf{P}}$ and $\mathbf{A}$. As for the optimization complexity, SoM-TP and other temporal poolings have all $\mathcal{O}(N)$, while MCL has $\mathcal{O}(N^2)$ to generate and compare multiple outputs. Therefore, compared with independent pooling, SoM-TP has little degradation of complexity, while optimization is effectively achieved even with an ensemble.

Conclusion

This paper proposes SoM-TP, a novel temporal pooling method employing a selection ensemble to address data dependency in temporal pooling by learning diverse perspectives. Utilizing a selection ensemble inspired by MCL, SoM-TP adapts to each data batch's characteristics. Optimal pooling selection with DPL attention achieves a comparison-free ensemble. We define DPLN and perspective loss for effective ensemble optimization. In quantitative evaluation, SoM-TP surpasses other pooling methods and state-of-the-art TSC models in UCR/UEA experiments. In qualitative analysis, LRP results highlight SoM-TP's ability to complement existing temporal pooling limitations. We re-examine the conventional role of temporal poolings, identify their limitations, and propose an efficient data-driven temporal pooling ensemble as a first attempt.

Acknowledgements

This work was partly supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2022-0-00984, Development of Artificial Intelligence Technology for Personalized Plug-and-Play Explanation and Verification of Explanation; No. 2022-0-00184, Development and Study of AI Technologies to Inexpensively Conform to Evolving Policy on Ethics; No. 2021-0-02068, Artificial Intelligence Innovation Hub; No. 2019-0-00075, Artificial Intelligence Graduate School Program (KAIST)).

References

Alsallakh, B.; Yan, D.; Kokhlikyan, N.; Miglani, V.; Reblitz-Richardson, O.; and Bhattacharya, P. 2023. Mind the Pool: Convolutional Neural Networks Can Overfit Input Size. In *Proceedings of the 11th International Conference on Learning Representations (ICLR'23)*.

Bach, S.; Binder, A.; Montavon, G.; Klauschen, F.; Müller, K.-R.; and Samek, W. 2015. On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation. *PLOS ONE*, 10(7): 1–46.

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:1811.00075.

Bagnall, A.; Lines, J.; Hills, J.; and Bostrom, A. 2015. Time-Series Classification with COTE: The Collective of Transformation-Based Ensembles. *IEEE Transactions on Knowledge and Data Engineering*, 27(9): 2522–2535.

Bai, S.; Kolter, J. Z.; and Koltun, V. 2018. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. *arXiv:1803.01271*.

Bay, S. D.; Kibler, D.; Pazzani, M. J.; and Smyth, P. 2000. The UCI KDD archive of large data sets for data mining research and experimentation. *ACM SIGKDD Explorations Newsletter*, 2(2): 81–85.

Cui, Z.; Chen, W.; and Chen, Y. 2016. Multi-scale convolutional neural networks for time series classification. *arXiv preprint arXiv:1603.06995*.

Cuturi, M.; and Blondel, M. 2017. Soft-DTW: A Differentiable Loss Function for Time-Series. In *Proceedings of the 34th International Conference on Machine Learning (ICML'17)*.

Dau, H. A.; Bagnall, A.; Kamgar, K.; Yeh, C.-C. M.; Zhu, Y.; Gharghabi, S.; Ratanamahatana, C. A.; and Keogh, E. 2019. The UCR time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6): 1293–1305.

Dempster, A.; Petitjean, F.; and Webb, G. I. 2020. ROCKET: Exceptionally Fast and Accurate Time Series Classification Using Random Convolutional Kernels. *Data Mining and Knowledge Discovery*, 34(5): 1454–1495.

Esling, P.; and Agon, C. 2012. Time-Series Data Mining. *ACM Comput. Surv.*, 45(1).

Foumani, N. M.; Tan, C. W.; Webb, G. I.; and Salehi, M. 2024. Improving position encoding of transformers for multivariate time series classification. *Data Mining and Knowledge Discovery*, 38(1): 22–48.

Gao, Z.; Wang, Q.; Zhang, B.; Hu, Q.; and Li, P. 2021. Temporal-attentive covariance pooling networks for video recognition. In *Proceedings of the 35th Advances in Neural Information Processing Systems (NIPS'21)*.

Gholamalinezhad, H.; and Khosravi, H. 2020. Pooling Methods in Deep Neural Networks, a Review. arXiv:2009.07485.

Girdhar, R.; and Ramanan, D. 2017. Attentional pooling for action recognition. In *Proceedings of the 31th Advances in Neural Information Processing Systems (NIPS'17)*.

Guzmán-rivera, A.; Batra, D.; and Kohli, P. 2012. In *Proceedings of the 26th Advances in Neural Information Processing Systems (NIPS'12)*.

Hinton, G. E.; Sabour, S.; and Frosst, N. 2018. Matrix capsules with EM routing. In *Proceedings of the 6th International Conference on Learning Representations (ICLR'18)*.

Hou, Q.; Zhang, L.; Cheng, M.-M.; and Feng, J. 2020. Strip pooling: Rethinking spatial pooling for scene parsing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR'20)*.

Ioffe, S.; and Szegedy, C. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on Machine Learning (ICML'15)*.

Ismail Fawaz, H.; Forestier, G.; Weber, J.; Idoumghar, L.; and Muller, P.-A. 2019. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4): 917–963.

Ismail Fawaz, H.; Lucas, B.; Forestier, G.; Pelletier, C.; Schmidt, D. F.; Weber, J.; Webb, G. I.; Idoumghar, L.; Muller, P.-A.; and Petitjean, F. 2020. InceptionTime: Finding AlexNet for Time Series Classification. *Data Mining and Knowledge Discovery*, 34(6): 1936–1962.

Kachuee, M.; Fazeli, S.; and Sarrafzadeh, M. 2018. ECG Heartbeat Classification: A Deep Transferable Representation. *CoRR*, abs/1805.00794.

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

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of 3rd International Conference on Learning Representations (ICLR'15)*.

Lecun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*.

Lee, D.; Lee, S.; and Yu, H. 2021. Learnable Dynamic Temporal Pooling for Time Series Classification. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI'21)*.

Lee, K.; Hwang, C.; Park, K.; and Shin, J. 2017. Confident Multiple Choice Learning. In *Proceedings of the 34th International Conference on Machine Learning (ICML'17)*.

Lee, S.; Purushwalkam, S.; Cogswell, M.; Ranjan, V.; Crandall, D.; and Batra, D. 2016. Stochastic Multiple Choice Learning for Training Diverse Deep Ensembles. In *Proceedings of the 30th Advances in Neural Information Processing Systems (NIPS'16)*.

Lines, J.; Taylor, S.; and Bagnall, A. 2016. HIVE-COTE: The Hierarchical Vote Collective of Transformation-Based Ensembles for Time Series Classification. In *Proceedings of the 16th IEEE International Conference on Data Mining (ICDM'16)*.

Liu, H.; Simonyan, K.; and Yang, Y. 2019. DARTS: Differentiable Architecture Search. In *Proceedings of the 7th International Conference on Learning Representations (ICLR'19)*.

Längkvist, M.; Karlsson, L.; and Loutfi, A. 2014. A review of unsupervised feature learning and deep learning for time-series modeling. *Pattern Recognition Letters*, 42: 11–24.

Nguyen, A.; Yosinski, J.; and Clune, J. 2015. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR'15)*.

Pigou, L.; Van Den Oord, A.; Dieleman, S.; Van Herreweghe, M.; and Dambre, J. 2018. Beyond temporal pooling: Recurrence and temporal convolutions for gesture recognition in video. *International Journal of Computer Vision*, 126: 430–439.

Rakthanmanon, T.; Campana, B.; Mueen, A.; Batista, G.; Westover, B.; Zhu, Q.; Zakaria, J.; and Keogh, E. 2012. Searching and mining trillions of time series subsequences under dynamic time warping. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'12)*.

Rippel, O.; Snoek, J.; and Adams, R. P. 2015. Spectral representations for convolutional neural networks. In *Proceedings of the 28th Advances in Neural Information Processing Systems (NIPS'15)*.

Sabour, S.; Frosst, N.; and Hinton, G. E. 2017. Dynamic routing between capsules. In *Proceedings of the 31st Advances in Neural Information Processing Systems (NIPS'17)*.

Schäfer, P. 2015. The BOSS is concerned with time series classification in the presence of noise. *Data Mining and Knowledge Discovery*, 29: 1505–1530.

Tan, C. W.; Dempster, A.; Bergmeir, C.; and Webb, G. I. 2022. MultiRocket: multiple pooling operators and transformations for fast and effective time series classification. *Data Mining and Knowledge Discovery*, 36(5): 1623–1646.

Tang, W.; Long, G.; Liu, L.; Zhou, T.; Blumenstein, M.; and Jiang, J. 2021. Omni-Scale CNNs: a simple and effective kernel size configuration for time series classification. In *Proceedings of the 9th International Conference on Learning Representations (ICLR'21)*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L. u.; and Polosukhin, I. 2017. Attention is All you Need. In *Proceedings of the 31st Advances in Neural Information Processing Systems (NIPS'17)*.

Wang, J.; Shao, Z.; Huang, X.; Lu, T.; Zhang, R.; and Lv, X. 2021. Spatial–temporal pooling for action recognition in videos. *Neurocomputing*, 451: 265–278.

Wang, Z.; Yan, W.; and Oates, T. 2017. Time series classification from scratch with deep neural networks: A strong baseline. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN'17)*.

Xiao, Q.; Wu, B.; Zhang, Y.; Liu, S.; Pechenizkiy, M.; Mocanu, E.; and Mocanu, D. C. 2022. Dynamic Sparse Network for Time Series Classification: Learning What to "See". In *Proceedings of the 36th Advances in Neural Information Processing Systems (NIPS'22)*.

Yu, D.; Wang, H.; Chen, P.; and Wei, Z. 2014. Mixed Pooling for Convolutional Neural Networks. *Rough Sets and Knowledge Technology*, 364–375.

Yuan, J.; Douzal-Chouakria, A.; Varasteh Yazdi, S.; and Wang, Z. 2019. A large margin time series nearest neighbour classification under locally weighted time warps. *Knowledge and Information Systems*, 59(1): 117–135.

Zerveas, G.; Jayaraman, S.; Patel, D.; Bhamidipaty, A.; and Eickhoff, C. 2021. A Transformer-Based Framework for Multivariate Time Series Representation Learning. In *Proceedings of the 27th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'21)*.

Zhang, X.; Gao, Y.; Lin, J.; and Lu, C.-T. 2020. Tapnet: Multivariate time series classification with attentional prototypical network. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI'20)*.

Zhao, P.; Luo, C.; Qiao, B.; Wang, L.; Rajmohan, S.; Lin, Q.; and Zhang, D. 2022. T-SMOTE: Temporal-oriented Synthetic Minority Oversampling Technique for Imbalanced Time Series Classification. In *Proceedings of the 31st International Joint Conference on Artificial Intelligence (IJCAI'22)*.

Zoph, B.; and Le, Q. 2017. Neural Architecture Search with Reinforcement Learning. In *Proceedings of the 5th International Conference on Learning Representations (ICLR'17)*.