DySA-TGN: Dynamic Self-Adaptive Temporal Graph Neural Network for Multivariate Time Series Classification

Xinmeng Wan and Zhiying Long (✉)

School of Artificial Intelligence, Beijing Normal University, Beijing, China
wanxm@mail.bnu.edu.cn, zlong@bnu.edu.cn

Abstract. Multivariate Time Series Classification (MTSC) holds significant importance in various areas. Temporal Graph Neural Networks (TGNNs) that construct dynamic graphs by treating each variate as a node show great potential in MTSC. However, most TGNNs still require a pre-defined graph structure in each time slot and an additional sequence learning module. TodyNet which can learn dynamic graphs without pre-defined graph structure and separate sequence learning was the first TGNN model applied to MTSC. However, TodyNet faces two issues. First, dynamic graph structure learning is purely driven by classification tasks, without fully exploiting dynamic feature representation. Second, TodyNet only aggregates spatial and short-term temporal information without long-term information during information propagation. Based on these issues, we propose the Dynamic Self-Adaptive Temporal Graph Neural Network (DySA-TGN), featuring two core components: 1) The Dynamic Graph Construction based on multi-view feature fusing (DGC-MVFF) module dynamically constructs adjacency matrices by fusing time-varying and time-invariant features; 2) The Graph Isomorphism Network with temporal aggregation (GIN-TA) module aggregates spatial, time-decay and time-variation information to propagate information. Experimental results demonstrated a 5% performance improvement of DySA-TGN over state-of-the-art deep learning methods on 24 UEA benchmark datasets. DySA-TGN significantly enhanced classification accuracy and showed good generalization.

Keywords: Multivariate Time Series Classification · Temporal Graph Neural Network · Dynamic Graph Isomorphism Network · Graph Structure Learning.

1 Introduction

In the digital era, multivariate time series(MTS) data have become increasingly important in industries such as industrial production [1], healthcare [2], and transportation [3]. Because MTS data contain rich temporal information as well as multiple continuous variables, MTS-based data mining and decision-making are becoming a research hotspot [4]. In particular, multivariate time series classification(MTSC), which focuses on identifying discriminant patterns that dis-

tinguish MTS samples according to class labels, improves our understanding of the underlying patterns and progression of events [5].

However, MTSC faces significant challenges, namely the complexity of multiple variables [6], the presence of long-term dependencies [7] and the high-dimensional feature space [8]. In response to the above issues, previous studies have explored several ways to improve the performance of MTSC. Early strategies that focused on feature extraction and similarity measurement are limited by their scalability when dealing with high-dimensional or large-scale datasets [9, 10]. Recently, deep learning has become a key solution for MTSC problems due to its automatic feature learning capabilities. Several deep learning models [11–15] have been widely applied to MTSC. Among them, Graph Neural Networks(GNNs) are particularly effective for MTS data because they can learn variable dependencies by mapping MTS to graph structures. However, most existing graph-based methods treat time series as static graphs, failing to fully exploit the potential of GNNs to capture temporal dynamics [16].

In recent years, Temporal Graph Neural Networks(TGNNs) have been introduced into some MTS tasks, such as MTS forecasting and anomaly detection, to capture dynamic relationships between variables. The temporal graph is a set of graph structures constructed on the entire MTS data according to the temporal order, which means that the connectivity of the nodes and the weights of the edges change in different time slots. In TGNNs, spatial dependencies in each time slot are typically extracted by graph convolutional network(GCN), while temporal dependencies between different time slots are learned by recurrent neural networks(RNNs) [17] or convolutional layers [18]. Despite their effectiveness, these methods rely on predefined graph structure in each time slot and require additional sequence learning structures to learn temporal dependencies, which complicates the model structure. In addition, few previous studies have applied TGNNs to MTSC tasks.

In a recent study, TodyNet [16] was proposed as an innovative TGNN framework tailored for MTSC tasks. TodyNet used a dynamic graph learning mechanism to capture spatial dependencies without predefined graph structures and temporal dependencies without additional sequence learning structures. Although TodyNet achieved promising classification results, it still has some weaknesses. (1) First, the graph adjacency matrix for each time slot in TodyNet is randomly initialized and learned by gradient descent. The graph structure is learned purely based on the classification task, without using the spatial relationship between node features, which could limit the graph representation capacity. In addition, random initialization could increase the instability of the model. (2) Second, TodyNet uses a dynamic graph isomorphism network to aggregate information from the previous time slot as well as the previously hidden layer. The dynamic graph information propagation in TodyNet only considers the short-term influence of the previous neighbor time slot on the current information, while ignoring the long-term influence of the other previous time slots, which may not accurately reflect the dynamic temporal dependency between different time slots. Therefore, it is necessary to further investigate dynamic graph learning and information

propagation mechanisms that can accurately capture the spatial and temporal relationship between nodes and are suitable for the MTSC tasks.

In this study, we proposed a novel dynamic graph framework, Dynamic Self-Adaptive Temporal Graph Neural Network(DySA-TGN), which adaptively learned the dynamic graph spatial structure by fusing both the time-invariant and time-varying feature representation as well as propagated temporal information through the Graph Isomorphism Network with temporal aggregation. DySA-TGN consists of two main modules that are the Dynamic Graph Construction based on multi-view feature fusing(DGC-MVFF) module and the Graph Isomorphism Network with temporal aggregation(GIN-TA) module. To fully capture the dynamic spatial relationship between node feature vectors, the DGC-MVFF module assumes that each node feature vector could be embedded in two feature subspaces(views), which are the time-varying subspace and the time-invariant subspace. The former is specific to each time slot and varies dynamically with the time slot, while the latter is shared by all time slots and does not change with time. Based on this assumption, the DGC-MVFF module maps the original node feature vectors into two subspaces and constructs a dynamic graph structure based on the concatenated feature representations of the two views in each time slot. In addition, the GIN-TA module aggregates both the spatial information of the previous layer and the previous temporal information that includes the exponential time-decay information of all the previous time slots and the time-variation information between two adjacent time slots. These two modules are integrated into an end-to-end jointly optimized network, which can automatically adjust feature weights and thus flexibly adapt to different datasets and time series characteristics without the need for complex hyperparameter tuning.

The main contributions of this study can be summarized as follows:

- The proposed DGC-MVFF module, which uses a dynamic graph structure learning mechanism based on time-varying and time-invariant feature representations, can fully capture the dynamic spatial relationships between graph nodes from MTS data and has good adaptation to data changes.
- The proposed GIN-TA module, which integrates both the exponential time-decay information and the time-variation information to propagate graph information, has a good ability to capture the short and long-term temporal dependencies as well as the dynamic variation effects.
- The proposed DySA-TGN model with few hyperparameters achieved superior classification performance to the state-of-the-art models and showed good generalization and interpretability on MTSC tasks.

2 Related studies

Multivariate Time Series Classification MTSC has a long history, starting with similarity-based and feature-based models such as DTW-1NN-I/D [9] and WEASEL+MUSE [10]. Traditional methods typically require feature engineering to manage large feature spaces. Deep learning models have been widely

applied to learn relevant features embedded in MTS data for classification, as deep learning-based approaches can learn feature representations and capture non-linear dependencies. Among various deep learning models, RNNs [12] and CNNs [14] have been applied to MTSC tasks by capturing temporal dependencies without considering the pairwise relationships between variables.

GNN-based Methods For MTS data, GNNs are a type of popular deep learning model because graph structures that represent the spatial relationship between variables can be learned from MTS data [19–21]. However, the previous studies apply GNNs by creating a single graph structure over the entire time series, overlooking dynamic processes. Therefore, TGNNs [22] have been proposed that are specifically designed for time-varying graph structure. METRO [23] unified spatio-temporal modeling with multi-scale temporal graphs and cross-scale fusion, while DGCRIN [24] modeling with graph recurrent cells integrating graph convolution and GRU. Although these methods excel at learning spatio-temporal dependencies, they may not be suitable for MTSC tasks due to the overly complex framework leading to overfitting. Recently, TodyNet [16] has been developed as a novel temporal graph fusion framework for MTSC. However, TodyNet is purely driven by a cost function to learn dynamic graphs without considering the distribution of real data and does not aggregate long-term temporal information during information propagation.

3 Methodology

In this section, we introduced the proposed DySA-TGN model. We first introduced the symbols used in MTSC. Then, we described the overall framework of DySA-TGN in Section 3.2 and provided a detailed description of each component of our model in Section 3.3 through Section 3.6.

3.1 Problem Formulation

A MTS sample is denoted as $X = \{x_1, x_2, ..., x_N\} \in \mathbb{R}^{N \times L}$, where $L \in \mathbb{N}^+$ is the time length of each time series and $N \in \mathbb{N}^+$ is the number of variates. Suppose the L time points of each variable are divided into S time slots. A temporal graph based on MTS is defined as $G^T = \{G^{T_1}, G^{T_2}, ..., G^{T_S}\}$, where $G^{T_i} = (V^{T_i}, E^{T_i})$ represents a subgraph defined in time slot T_i. V^{T_i} represents N nodes at time slot T_i and E^{T_i} represents the connections between nodes at time slot T_i.

3.2 Model Architecture Overview

The architecture of the proposed DySA-TGN model is illustrated in Fig. 1. The DySA-TGN model consists of four core modules which are Temporal Convolution(TC), DGC-MVFF, GIN-TA, and Temporal Graph Pooling(TGP).

First, a MTS sample is input to TC to extract local temporal features. Second, the data is segmented into time slots and processed by the DGC-MVFF

module to learn dynamic graph structures. Third, GIN-TA performs information propagation by integrating both spatial and temporal information. Fourth, the TGP module reduces the number of nodes while preserving the dynamic properties. Finally, the output module produces classification results through mean pooling and fully connected layers. The detailed design of the model is described in the following sections.

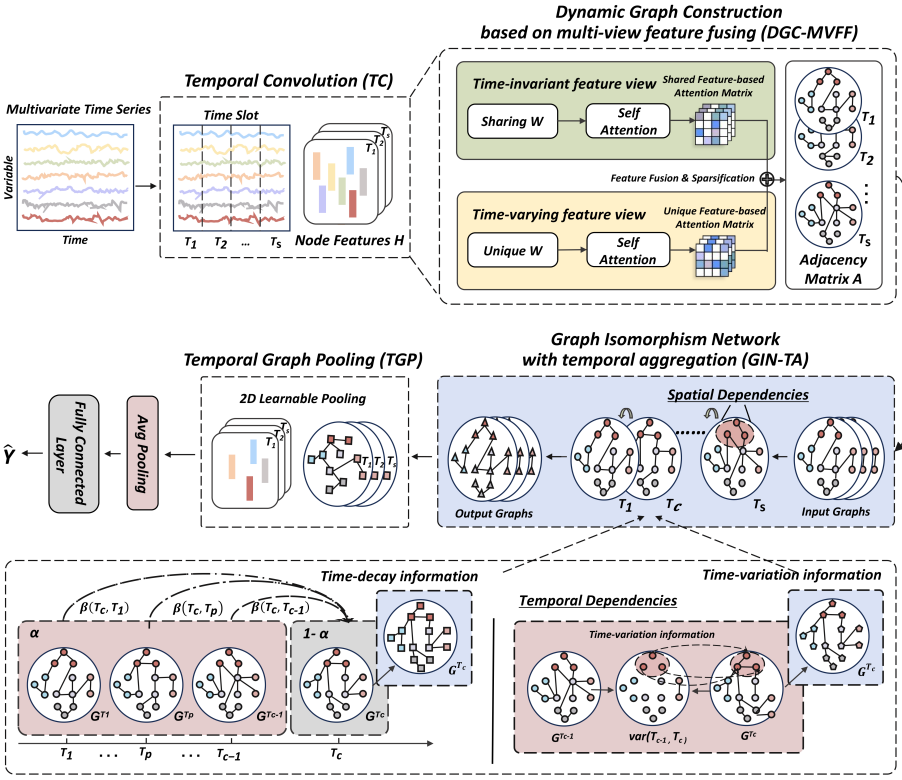

Fig. 1. The framework of DySA-TGN.

3.3 Temporal Convolution (TC)

The temporal convolutional(TC) module uses three convolutional layers of different sizes k_s to extract temporal features from the input $X \in \mathbb{R}^{N \times L}$. The padding techniques were used to keep the length of the time series L unchanged. After feature extraction, the features are temporally segmented into S time slots, which can be represented by the learning embeddings $H = \{h^{T_1}, h^{T_2}, \ldots, h^{T_S}\} \in \mathbb{R}^{S \times N \times L_T}$. L_T is the length of each time slot. The segmented features H are fed into the DGC-MVFF module to generate graph structures for each time slot.

3.4 Dynamic Graph Construction based on multi-view feature fusing (DGC-MVFF)

This module is based on our innovative assumption that the feature vector of each node can be embedded into a time-varying feature subspace and a time-invariant feature subspace. The time-varying feature subspace contains feature representations that are specific to each time slot. The time-invariant feature subspace contains feature representations that have universal intrinsic relevance across time slots, although they have different manifestations. Therefore, the DGC-MVFF module is able to capture the underlying feature representations of MTS data from the two feature views and improve model adaptability.

The DGC-MVFF module receives the segmented embedding $H = \{H^{T_1}, H^{T_2}, ..., h^{T_S}\} \in \mathbb{R}^{S \times N \times L_T}$ generated by the TC module as input and learns the time-invariant feature representations common to the time slots and the time-varying feature representations specific to each time slot. The two-view feature representations are merged to construct a dense adjacency matrix. The module sparsifies the adjacency matrix and outputs the structure information $A \in (0,1)^{S \times N \times N}$.

Time-invariant and Time-varying Feature Learning For a given time slot T_m, the segmented feature matrix $H^{T_m} = \{h_1^{T_m}, ..., h_N^{T_m}\} \in \mathbb{R}^{N \times L_T}$ consists of the feature vector of all the N nodes in the time slot. Based on the self-attention mechanism [25], H^{T_m} is mapped into the time-invariant subspaces by two trainable time-invariant weight matrices $W_q, W_k \in \mathbb{R}^{L_T \times d_m}$ that are shared by all the time slots (see Equation 1). $Q_{tin}^{T_m}, K_{tin}^{T_m} \in \mathbb{R}^{N \times d_m}$ represent the time-invariant feature representations.

$$Q_{tin}^{T_m} = H^{T_m} \cdot W_q \quad K_{tin}^{T_m} = H^{T_m} \cdot W_k \tag{1}$$

Meanwhile, H^{T_m} is mapped into time-varying subspaces by two trainable time-varying weight matrices $W_Q^{T_m}, W_K^{T_m} \in \mathbb{R}^{L_T \times d_m}$ that are unique to the time slot T_m (see Equation 2). $Q_{tva}^{T_m}, K_{tva}^{T_m} \in \mathbb{R}^{N \times d_m}$ represent the time-varying feature representation at time slot T_m.

$$Q_{tva}^{T_m} = H^{T_m} \cdot W_Q^{T_m} \quad K_{tva}^{T_m} = H^{T_m} \cdot W_K^{T_m} \tag{2}$$

Feature Fusion and Dynamic Graph Construction The time-invariant and time-varying feature representations are merged as $Q^{T_m}, K^{T_m} \in \mathbb{R}^{N \times 2d_m}$ to construct a dense adjacency matrix $A^{T_m} \in \mathbb{R}^{N \times N}$ by using the scaled dot-product attention function (see Equation 3 and 4).

$$Q^{T_m} = \text{CONCAT}\left(Q_{tin}^{T_m}, Q_{tva}^{T_m}\right) \quad K^{T_m} = \text{CONCAT}\left(K_{tin}^{T_m}, K_{tva}^{T_m}\right) \tag{3}$$

$$A^{T_m} = \exp\left(\frac{Q^{T_m} \cdot (K^{T_m})^{\mathsf{T}}}{\sqrt{d_m}}\right) \tag{4}$$

The element $a_{ij}^{T_m}$ in A^{T_m} represents the correlation between node v_j and node v_i at time slot T_m. The adjacency matrix computed by Equation 4 represents a kind of fully connected graph, which can introduce noise into the graph data and increase the computational complexity [26]. Therefore, A^{T_m} is further transformed into a sparse adjacency matrix by a soft threshold operator $\sigma(\theta) \in (0, 1)$ with a learnable parameter θ (see Equation 5).

$$\hat{a}_{ij}^{T_m} = \begin{cases} a_{ij}^{T_m} - \sigma(\theta), & \text{if } a_{ij}^{T_m} > \sigma(\theta) \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

Where $a_{ij}^{T_m}$ is the edge in dense adjacency matrices and $\hat{a}_{ij}^{T_m}$ is the edge in sparse adjacency matrices. Unlike fixed absolute [16] or percentile thresholds [27] that require manual hyperparameter tuning, the soft threshold method adaptively learns the optimal sparsity level during training. After learning the sparse adjacency matrices in all the time slots, the temporal graph G^T is obtained and is fed into the GIN-TA module for message passing and feature aggregation.

3.5 Graph Isomorphism Network with temporal aggregation (GIN-TA)

The proposed GIN-TA module propagates information by integrating both the previous spatial layer and the dynamic temporal information. GIN-TA assumes that the state of the current time slot depends not only on the node information of all previous time slots but also on the variation between the two adjacent time slots. Based on this assumption, GIN-TA aggregates both the time-decay and time-variation information.

Time-decay and Time-variation Information The time-decay information is modeled as the dependency between any two time slots, which decays exponentially as the time interval increases. For a given time slot T_c, the weight of its connection to any previous time slot $T_p (p \in \{1, ..., c-1\})$ can be calculated by $\beta(T_c, T_p) = \exp\left(-\sigma(T_c - T_p)\right)$, which is called time-decay factor. The node information of each previous time slot is weighted by the time-decay factor and summed to form time-decay information $H_{td}^{T_c}$ (see Equation 6 and Fig. 1).

$$H_{td}^{T_c} = \sum\nolimits_{T_p=1}^{c-1} \beta(T_c, T_p) \cdot H^{T_p} \tag{6}$$

The time-variation information is modeled as the difference of the node feature representations between two neighbor time nodes. For a given time slot T_i, the time-variation of embedding $H_{\text{var}}^{T_i}$ is defined by Equation 7. There is no time-variation for the first time slot.

$$H_{\text{var}}^{T_i} = \{\, h_{\text{var},1}^{T_i}, ..., h_{\text{var},N}^{T_i} \,\} = \{\, |h_1^{T_i} - h_1^{T_{i-1}}|, ..., |h_N^{T_i} - h_N^{T_{i-1}}| \,\} \tag{7}$$

Information Aggregation Finally, both the time-decay and time-variation information were integrated into the Graph Isomorphism Network(GIN) [28] to form the GIN-TA module. For the node v of the layer l in the time slot T_i, the feature representation $h_v^{(l,T_i)}$ of the node can be aggregated by Equation 8.

$$h_v^{(l,T_i)} = \mathrm{MLP}^{(l,T_i)}\left((1-\alpha)\cdot h_v^{(l-1,T_i)} + \alpha\Big(\sum\nolimits_{T_p=1}^{i-1}\beta(T_i,T_p)\cdot h_v^{(l-1,T_p)}\Big) + \left|h_v^{(l-1,T_i)} - h_v^{(l-1,T_{i-1})}\right| + \sum_{u\in N(v)}\tilde{a}_{v,u}^{T_i}\cdot h_u^{(l-1,T_i)}\right) \tag{8}$$

α is a learnable parameter that controls the weight of the effect of all the previous time slots and is adaptive to the variation across different datasets. The edge weight $\tilde{a}_{v,u}^{T_i} \in (0,1)$ is the normalized form of $a_{v,u}^{T_i} \in A^{T_i}$. The matrix form of GIN-TA is in Equation 9.

$$H^l = \mathrm{MLP}\Big((\tilde{A} + (1-\alpha)\cdot I)\cdot H^{l-1} + H_{td}^{l-1} + H_{var}^{l-1}\Big) \tag{9}$$

where $H^l \in \mathbb{R}^{S\times N\times L_T}$ is the output tensor of the l-th layer. $\tilde{A}$ is the normalized representation of $A \in (0,1)^{S\times N\times N}$. I is the identity matrix. H^l aggregating spatial information, time-decay information, and time-variation information from the previous layer and previous time slots.

3.6 Temporal Graph Pooling (TGP)

We use TGP [16] for temporal graph pooling to enhance feature and semantic representation with minimal information loss. It applies a hierarchical pooling strategy after each GNN layer, using 2D convolutions for node aggregation, resulting in new node representations. The input tensor is $H^l \in \mathbb{R}^{N\times L}$. The kernel of the pooling layer $(1, k_s)$ adapts subsequent temporal convolutions to preserve the model's receptive field and information flow, outputting a reshaped tensor $\tilde{H}^l \in \mathbb{R}^{\tilde{N}\times L}$, where $\tilde{N}$ is the post-pooling node count. This approach efficiently pools graph structures while preserving time series dynamics.

3.7 Time-Complexity Analysis

In this subsection, we analyze the time complexity of the proposed DGC-MVFF and GIN-TA modules.

The time complexity of the DGC-MVFF module is $O(S\cdot(2Nd_mL/S + N^2d_m))$, where S is the number of dynamic graphs, N is the number of nodes, L is the length of input features, and d_m is the dimension of feature embedding. This complexity arises from computing both time-invariant and time-varying features, as well as calculating the adjacency matrix via self-attention, which can be simplified to $O(SN^2d_m)$.

The time complexity of the GIN-TA module includes three components: the aggregation complexity of GIN, the computation of time-decay information, and the calculation of time-variation information. Thus, the overall time complexity is $O(S \cdot (N^2 + SNL/S + NL/S))$, with the dominant term being $O(SN^2)$.

4 Experimental Studies

In this section, we conducted extensive experiments to evaluate the performance of DySA-TGN on various MTSC tasks and the impact of removing key components of the framework. Furthermore, we discussed the sensitivity of the hyperparameters and visualized the learning process of the graph structure.

4.1 Experimental Settings

Datasets The datasets used in this study are from the UEA Multivariate Time Series Classification Archive [29]. To ensure the accuracy of the experimental results, we selected datasets with consistent data lengths and no missing values from this archive. The 24 datasets have been widely used in several studies. We used the original datasets for comparisons with other methods to fairly assess performance. The basic information on these datasets is detailed in Table 1.

Table 1. Basic information of 24 UEA datasets.

Dataset	Code	Type	Train	Test	Series	Length	Classes
ArticularyWordRecognition	AWR	Motion	275	300	9	144	25
AtrialFibrillation	AF	ECG	15	15	2	640	3
BasicMotions	BM	HAR	40	40	6	100	4
Cricket	CR	HAR	108	72	6	1197	12
EigenWorms	EW	Motion	128	131	6	17984	5
Epilepsy	EP	HAR	137	138	3	206	4
EthanolConcentration	EC	HAR	261	263	3	1751	4
ERing	ER	Other	30	270	4	65	6
FaceDetection	FD	EEG	5890	3524	144	62	2
FingerMovements	FM	EEG	316	100	28	50	2
HandMovementDirection	HMD	EEG	160	74	10	400	4
Handwriting	HW	HAR	150	850	3	152	26
Heartbeat	HB	AS	204	205	61	405	2
Libras	LIB	HAR	180	180	2	45	15
LSST	LSST	Other	2459	2466	6	36	14
MotorImagery	MI	EEG	278	100	64	3000	2
NATOPS	NA	HAR	180	180	24	51	6
PenDigits	PD	Motion	7494	3498	2	8	10
Phoneme	PM	AS	3315	3353	11	217	39
RacketSports	RS	HAR	151	152	6	30	4
SelfRegulationsCP1	SRS1	EEG	268	293	6	896	2
SelfRegulationSCP2	SRS2	EEG	200	180	7	1152	2
StandWalkjump	SWJ	ECG	12	15	4	2500	3
UWaveGestureLibrary	UW	HAR	120	320	3	315	8

Baselines We select state-of-the-art machine learning and neural network-based methods as benchmarks for comparison. *(1) Machine learning methods:* ED-1NN [14], DTW-1NN-I/D [9], WEASEL+MUSE [10]. *(2) Deep learning methods:* TapNet [11], MLSTM-FCN [12], ShapeNet [13], MOS/OS-CNN [14], TodyNet [16] (the first to apply TGNN to MTSC), DKN [15] (the forefront of deep learning in MTSC).

Implementation Details In the DySA-TGN model, the number of graphs S is set to 4, and the embedding dimension for graph structure learning d_m is 32. The temporal convolution consists of 3 layers with kernel sizes k_s of 11,3,3. The batch size b_s is set to 16, the learning rate l_r is set to 10^{-4}, and the pooling rate p_r is 0.2. Each dataset is trained for 500 epochs. All experiments are conducted using PyTorch 2.2.1 in Python 3.9.1. The computational infrastructure includes an Ubuntu 20.04 operating system and a GPU NVIDIA GeForce RTX 3090.

Model Evaluation Methods We evaluate the performance of different methods on test datasets by calculating accuracy, average precision, top-1 count, and 1-to-1 win/draw/loss counts.

In addition, we use the equation $Rank_C = \frac{\sum_{i=1}^{D} rank_i^C}{D}$ to obtain average rank comparison results, where a rank closer to 1 indicates better classification performance. In the formula, $Rank_C$ is the average rank of classifier C across D datasets, $rank_i^C$ is the rank achieved by classifier C on dataset i.

Finally, the Wilcoxon signed-rank tests measured the accuracy differences between the proposed DySA-TGN model and existing methods, as well as the differences before and after removing key modules on all datasets. The F-statistic tested whether different hyperparameter settings affected classification accuracy.

4.2 Classification Performance Evaluation

To evaluate the performance of DySA-TGN, we compared it to baseline models on all the benchmark datasets listed in Table 1. Table 2 shows the accuracy and evaluation metrics of all methods. The best and second-best results are highlighted in bold and underlined respectively.

Experimental results show that DySA-TGN outperforms most of the baseline models. DySA-TGN achieved the highest classification accuracy on 14 datasets and produced an average accuracy of 0.766, which was approximately 5% higher than the second-best method TodyNet(0.730). In 1:1 win/loss comparisons, DySA-TGN significantly outperformed all baseline models. Specifically, it outperformed the state-of-the-art methods by 10% on the LSST, FD, and FM datasets, and by 7.7% and 6.3% on the HB and EC datasets respectively.

Meanwhile, the results of the Wilcoxon signed-rank test for DySA-TGN versus the other baseline models are shown in the last row of Table 2. It can be seen that DySA-TGN showed significantly higher accuracy than all other baseline models ($p < 0.05$). The superior performance of DySA-TGN could indicate that the proposed dynamic graph structure learning and information propagation with temporal aggregation can effectively capture the dynamic spatial and temporal relationships between graph nodes.

Additionally, the average ranks of all the classifiers are shown in Fig. 2. It can be seen that DySA-TGN has the lowest overall average rank of 2.542 among all the methods, especially outperforming existing deep learning methods such as TodyNet and DKN, as well as traditional classifiers like ShapeNet and DTW. The results highlight the excellent generalization ability of DySA-TGN.

Table 2. The results of the 24 benchmark UEA datasets.

Dataset /Method	ED -1NN	DTW -1NN-I	DTW -1NN-D	Tap Net	MLSTM -FCN	Shape Net	WEASEL +MUSE	OS -CNN	MOS -CNN	Tody Net	DKN	DySA -TGN
AWR	0.970	0.980	0.987	0.987	0.987	0.990	0.973	0.988	0.991	0.987	**0.993**	**0.993**
AF	0.267	0.267	0.200	0.333	0.400	0.333	0.267	0.233	0.183	0.467	0.467	**0.533**
BM	0.675	**1.000**	0.975	**1.000**	**1.000**	**1.000**	0.950	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
CR	0.944	0.986	**1.000**	0.958	0.986	**1.000**	0.917	0.993	0.990	**1.000**	0.951	0.986
EW	0.550	0.603	0.618	0.489	0.878	**0.890**	0.504	0.414	0.508	0.840	0.628	0.870
EP	0.667	0.978	0.964	0.971	0.987	**1.000**	0.761	0.980	0.996	0.971	0.979	0.993
EC	0.293	0.304	0.323	0.323	0.312	0.133	0.373	0.240	0.415	0.350	0.372	**0.441**
ER	0.133	0.133	0.133	0.133	0.133	0.430	0.133	0.881	0.915	0.915	0.933	**0.937**
FD	0.519	0.513	0.529	0.556	0.602	0.545	0.545	0.575	0.597	0.627	0.631	**0.693**
FM	0.550	0.520	0.530	0.530	0.589	0.490	0.580	0.568	0.568	0.570	0.600	**0.660**
HMD	0.279	0.306	0.231	0.378	0.338	0.365	0.365	0.443	0.361	0.649	**0.662**	0.595
HW	0.371	0.509	0.607	0.357	0.451	0.605	0.286	0.668	**0.677**	0.436	0.231	0.596
HB	0.620	0.659	0.717	0.751	0.756	0.727	0.663	0.489	0.604	0.756	0.765	**0.824**
LIB	0.833	0.894	0.872	0.850	0.856	0.878	0.856	0.950	**0.965**	0.850	0.900	0.894
LSST	0.456	0.575	0.551	0.568	0.590	0.590	0.373	0.413	0.521	0.615	0.347	**0.680**
MI	0.510	0.390	0.500	0.590	0.610	0.500	0.510	0.535	0.515	0.640	0.620	**0.650**
NA	0.860	0.850	0.883	0.939	0.883	0.870	0.889	0.968	0.951	0.972	0.872	**0.978**
PD	0.973	0.939	0.977	0.980	0.977	0.948	0.978	0.985	0.983	0.987	0.948	**0.989**
PM	0.104	0.151	0.151	0.175	0.298	0.190	0.110	0.299	0.295	0.309	**0.525**	0.306
RS	0.868	0.842	0.803	0.868	0.882	**0.934**	0.803	0.877	0.929	0.803	0.879	0.875
SRS1	0.771	0.765	0.775	0.652	0.782	0.710	0.874	0.835	0.829	0.898	0.913	**0.922**
SRS2	0.483	0.533	0.539	0.550	0.578	0.460	0.472	0.532	0.510	0.550	0.600	**0.606**
SWJ	0.200	0.333	0.200	0.400	**0.533**	0.333	0.067	0.383	0.383	0.467	**0.533**	0.467
UW	0.881	0.869	0.903	0.894	0.906	0.916	0.891	**0.927**	0.926	0.850	0.897	0.906
Average Acc	0.574	0.621	0.624	0.635	0.680	0.660	0.589	0.674	0.692	0.730	0.719	0.766
Num.Top-1	0	1	1	1	2	5	0	2	3	2	5	14
Ours Wins	24	21	22	23	18	17	24	18	17	19	17	-
Ours Draws	0	3	0	1	3	1	0	1	1	2	2	-
Ours Losses	0	0	2	0	3	6	0	5	6	3	5	-
P-value	1.19E-07	5.96E-05	2.98E-06	2.70E-05	6.57E-04	1.02E-03	1.19E-07	1.48E-03	4.25E-03	5.44E-04	1.94E-02	-

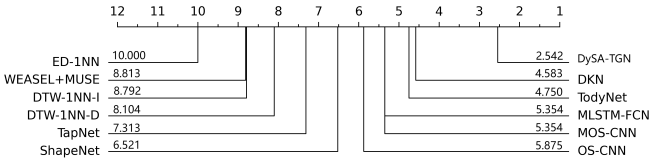

Fig. 2. The ranking of classifiers.

4.3 Ablation Experiments

In this section, we investigated the internal mechanisms of DySA-TGN, focusing on the roles of the constructed DGC-MVFF and GIN-TA modules within the entire neural network. The ablation studies were conducted on 24 UEA datasets without altering the settings. Our discussions include the following scenarios:

1) Effectiveness of DGC-MVFF:

- DGC-MVFF: DySA-TGN without DGC-MVFF module.
- DGC-TI: DySA-TGN without time-invariant feature view of DGC-MVFF.
- DGC-TV: DySA-TGN without time-varying feature view of DGC-MVFF.

2) Effectiveness of GIN-TA:

- GIN-TA: DySA-TGN without GIN-TA module.
- GIN-TD: DySA-TGN without time-decay information of GIN-TA.
- GIN-TV: DySA-TGN without time-variation information of GIN-TA.

3) Effectiveness of Temporal Graph:

- Graph: DySA-TGN without GNN, passing TC output to the output layer.
- T-Graph: DySA-TGN without TGNN, using a graph over the whole series.

The results of the ablation study are presented in Table 3. Pairwise Wilcoxon signed-rank tests were performed to assess the statistical significance of the accuracy differences between DySA-TGN and the other baseline methods. The p-values are listed at the bottom of Table 3. The results showed that the accuracies of all the DySA-TGN models with the key components removed were significantly lower than the original DySA-TGN model ($p < 0.05$), suggesting that all the key components were necessary for the DySA-TGN model.

Table 3. Accuracies of the ablation experiments on the 24 UEA datasets.

	DGC-MVFF			GIN-TA			Temporal Graph		
Dataset	- DGC-MVFF	- DGC-TI	- DGC-TV	- GIN-TA	- GIN-TD	- GIN-TV	- Graph	- T-Graph	DySA-TGN
AWR	0.983	0.987	0.993	0.980	0.993	0.983	0.743	0.967	0.993
AF	0.467	0.533	0.533	0.400	0.467	0.533	0.400	0.467	0.533
BM	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
CR	0.944	0.972	0.986	1.000	0.986	0.972	0.889	0.986	0.986
EW	0.763	0.836	0.810	0.527	0.847	0.864	0.466	0.481	0.870
EP	0.957	0.986	0.986	0.949	0.949	0.971	1.000	0.978	0.993
EC	0.380	0.403	0.415	0.411	0.418	0.502	0.342	0.342	0.441
ER	0.911	0.900	0.896	0.800	0.917	0.911	0.774	0.900	0.937
FD	0.678	0.649	0.663	0.679	0.649	0.659	0.560	0.568	0.693
FM	0.650	0.620	0.600	0.650	0.650	0.620	0.600	0.670	0.660
HMD	0.500	0.608	0.635	0.446	0.568	0.595	0.405	0.459	0.595
HW	0.442	0.566	0.578	0.354	0.444	0.413	0.138	0.315	0.596
HB	0.805	0.829	0.815	0.805	0.800	0.795	0.756	0.824	0.824
LIB	0.811	0.861	0.856	0.789	0.867	0.839	0.561	0.900	0.894
LSST	0.661	0.674	0.653	0.635	0.665	0.642	0.533	0.653	0.680
MI	0.640	0.630	0.610	0.600	0.620	0.640	0.640	0.610	0.650
NA	0.944	0.961	0.950	0.911	0.967	0.989	0.728	0.972	0.978
PD	0.989	0.989	0.985	0.984	0.985	0.988	0.975	0.991	0.989
PM	0.310	0.310	0.284	0.305	0.292	0.310	0.302	0.340	0.306
RS	0.816	0.855	0.842	0.842	0.855	0.836	0.678	0.895	0.875
SRS1	0.908	0.918	0.908	0.905	0.901	0.898	0.911	0.847	0.922
SRS2	0.589	0.594	0.578	0.561	0.578	0.606	0.583	0.589	0.606
SWJ	0.333	0.467	0.400	0.400	0.400	0.467	0.600	0.533	0.467
UW	0.875	0.897	0.903	0.863	0.918	0.896	0.531	0.800	0.906
Average Acc	0.723	0.752	0.745	0.700	0.739	0.747	0.630	0.712	0.766
P-value	4.59E-05	6.34E-04	8.32E-04	5.95E-05	1.06E-04	6.19E-03	1.62E-04	7.06E-03	-

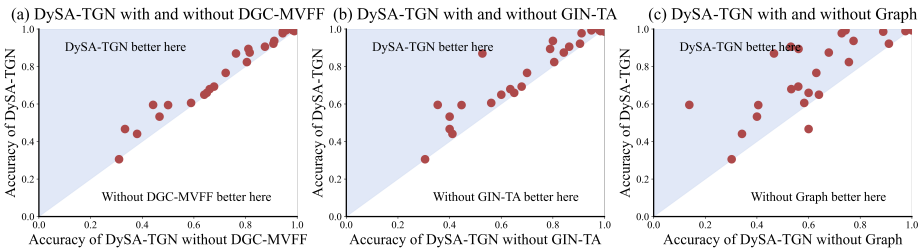

Fig. 3. Scatter plot of ablation experiments.

We observed that the effect of different modules varied across different datasets. To demonstrate the effectiveness of our proposed module more intuitively, the results of three major ablation experiments were further visualized. Figure 3 shows the scatter plots of classification accuracy for DySA-TGN versus ablated models. For all three ablated models, almost all points fall within the colored regions, indicating that DySA-TGN achieved higher accuracy than DySA-TGN without DGC-MVFF, GIN-TA, and Graph models on almost all datasets.

The consistently high accuracy of the DySA-TGN with DGC-MVFF and GIN-TA modules across all datasets may indicate that the two proposed modules have a better self-adaptive ability to different MTS datasets than the models without them. The good self-adaptive ability of DGC-MVFF could be attributed to the time-varying and time-invariant feature representations learned from MTS data, which can accurately capture the dynamic spatial relationship between variables. In addition, the good adaptive ability of the GIN-TA module may indicate the effectiveness of aggregating the long-term and short-term decay information and the time variation information in MTS tasks.

Therefore, the results of the ablation study showed that each individual component of DySA-TGN plays an essential role in MTSC tasks. When multiple modules work together, our model can extract more abstract concepts and thus capture different features more effectively than the other models.

4.4 Hyperparameter Sensitivity

We conducted a sensitivity analysis of the main hyperparameters (embedding dimension d_m, kernel size k_s, time slot number S, pooling ratio p_r) of DySA-TGN to evaluate the model performance and reduce the biases introduced by dataset characteristics. The experiment selected 14 datasets including AWR, BM, CR, EP, ER, FM, HMD, HW, LIB, NA, RS, SRS1, SRS2, and UW. The box plot in Fig. 4 shows the variation in classification accuracy across all datasets. The red curves represent the variation of the average accuracy of DySA-TGN with the increase of the four parameters. Based on the highest classification accuracy, the optimal parameters of d_m, k_s, S, p_r were set to 32, [11,3,3], 4, 0.2 respectively.

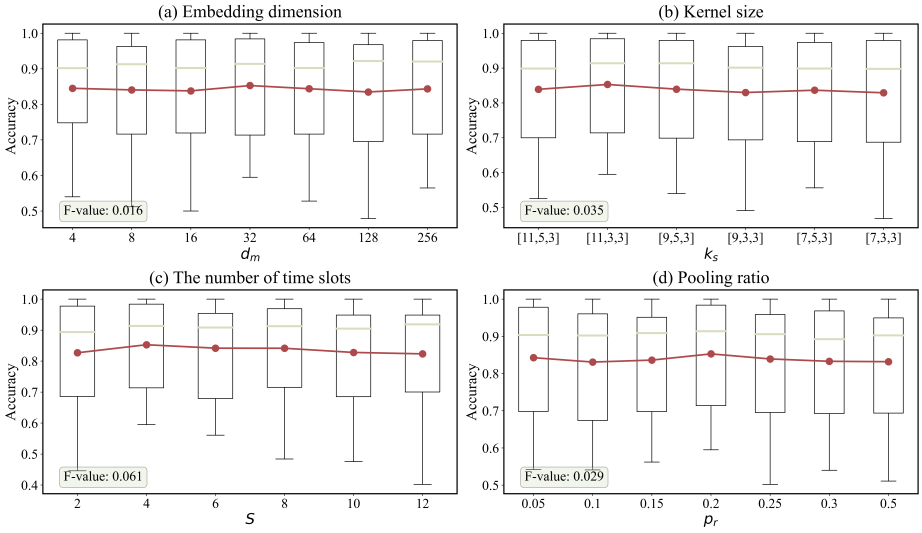

Fig. 4. Hyperparameter evaluation.

We also performed the F-test to check whether the accuracy has the same variance under different hyperparameter choices. The F-values are labeled in the lower left corner of the graph. The results showed that the F-values under different hyperparameter settings are all below the F-threshold ($F_{th} = 2.20$ for d_m and p_r and 2.33 for k_s and S), indicating no significant difference between the results under different hyperparameter choices. The results may indicate that DySA-TGN is insensitive to parameter variation.

4.5 Visualization Analysis

We further analyzed the convergence performance of the model by visualizing the training process, while validating the effectiveness of the proposed key component DGC-MVFF by visualizing the dynamic graph structure.

Model Convergence and Generalization Performance This section aims to analyze the differences in convergence and generalization between DySA-TGN and TodyNet [16]. We trained the models on the 14 datasets mentioned in Section 4.4 and recorded the classification accuracy and loss values of the test datasets. The variation of mean accuracy and loss with training epochs is shown in Fig. 5. In contrast to TodyNet, DySA-TGN shows a faster-increasing rate of accuracy and decreasing rate of loss in the first 50 epochs, which may indicate that DySA-TGN had a faster convergence speed. In addition, DySA-TGN achieved higher accuracy and lower loss than TodyNet after 100 epochs in the test dataset. The results demonstrated the stronger generalization capability of DySA-TGN, enabling it to perform better on unseen data than TodyNet.

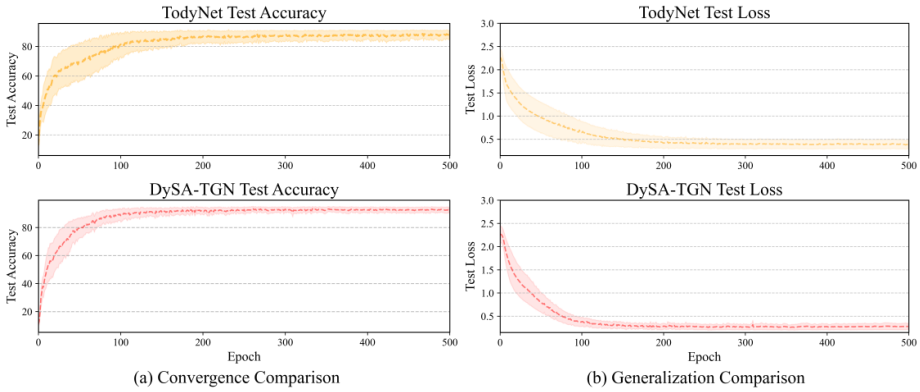

Fig. 5. Comparison between DySA-TGN and TodyNet.

Effectiveness of DGC-MVFF One of the main advantages of DySA-TGN is its ability to learn task-aware dynamic graph structures from MTS. The distributions of weights of the learnable adjacency matrices of the Heartbeat dataset in

different time slots are shown in Fig. 6. Each time series of the Heartbeat dataset is divided into four time slots. The adjacency matrices showed both variations and similarities in the graph structure across the four time slots, which may support the existence of both time-varying and time-invariant feature representations in MTS data. The results also demonstrated the reasonability of the proposed DGC-MVFF module.

The above results show that DySA-TGN accelerates model convergence and improves classification accuracy by automatically optimizing graph structure and adapting to dynamic changes in the data.

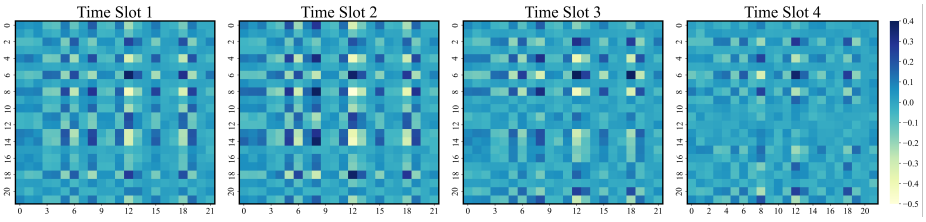

Fig. 6. The learning process of adjacency matrices for four time slots.

5 Conclusion

This study proposed an innovative framework DySA-TGN for Multivariate Time Series Classification(MTSC), which can adaptively learn dynamic graph structures by fusing time-varying and time-invariant feature representations, and perform information propagation by aggregating spatial information, time-decay information, and time-variation information. Experimental results showed that DySA-TGN outperformed existing state-of-the-art deep learning methods on 24 UEA benchmark datasets, significantly improving both the generalization ability and interpretability of the model.

Acknowledgments. This study was funded by the National Natural Science Foundation of China (grant number 62071050).

References

1. Yao, Y., et al.: Multivariate time-series prediction in industrial processes via a deep hybrid network under data uncertainty. IEEE Trans. Ind. Inf. pp. 1977–1987 (2022)
2. Lee, J.M., et al.: Modeling multivariate clinical event time-series with recurrent temporal mechanisms. Artif. Intell. Med. **112**, 102021 (2021)
3. Ji, J., et al.: Stden: Towards physics-guided neural networks for traffic flow prediction. In: Proc. of AAAI. vol. 36, pp. 4048–4056 (2022)
4. Jin, M., et al.: A survey on graph neural networks for time series: Forecasting, classification, imputation, and anomaly detection. IEEE Trans. Pattern Anal. Mach. Intell. (2024)

5. Mohammadi Foumani, N., et al.: Deep learning for time series classification and extrinsic regression: A current survey. ACM Comput. Surv. **56**(9), 1–45 (2024)
6. Yang, Q., et al.: 10 challenging problems in data mining research. Int. J. Inf. Technol. Decis. Making **5**(04), 597–604 (2006)
7. Li, G., et al.: Deep learning for anomaly detection in multivariate time series: Approaches, applications, and challenges. Inf. Fusion **91**, 93–102 (2023)
8. Lim, B., et al.: Time-series forecasting with deep learning: a survey. Phil. Trans. R. Soc. A **379**(2194), 20200209 (2021)
9. Chen, Y., et al.: Dtw-d: time series semi-supervised learning from a single example. In: Proc. of KDD. pp. 383–391 (2013)
10. Schäfer, P., et al.: Multivariate time series classification with weasel+ muse. arXiv preprint arXiv:1711.11343 (2017)
11. Zhang, X., et al.: Tapnet: Multivariate time series classification with attentional prototypical network. In: Proc. of AAAI. vol. 34, pp. 6845–6852 (2020)
12. Karim, F., et al.: Multivariate lstm-fcns for time series classification. Neural networks **116**, 237–245 (2019)
13. Li, G., et al.: Shapenet: A shapelet-neural network approach for multivariate time series classification. In: Proc. of AAAI. vol. 35, pp. 8375–8383 (2021)
14. Tang, W., et al.: Omni-scale cnns: a simple and effective kernel size configuration for time series classification. arXiv preprint arXiv:2002.10061 (2020)
15. Xiao, Z., et al.: Densely knowledge-aware network for multivariate time series classification. IEEE Trans. Syst. Man Cybern.: Syst. (2024)
16. Liu, H., et al.: Todynet: temporal dynamic graph neural network for multivariate time series classification. Inf. Sci. p. 120914 (2024)
17. Xu, D., et al.: Spatio-temporal attentive rnn for node classification in temporal attributed graphs. In: Proc. of IJCAI. pp. 3947–3953 (2019)
18. Li, M., et al.: Spatial-temporal fusion graph neural networks for traffic flow forecasting. In: Proc. of AAAI. vol. 35, pp. 4189–4196 (2021)
19. Scarselli, F., et al.: The graph neural network model. IEEE Trans. Neural Networks **20**(1), 61–80 (2008)
20. Duan, Z., et al.: Multivariate time-series classification with hierarchical variational graph pooling. Neural Networks **154**, 481–490 (2022)
21. Chauhan, J., et al.: Multi-variate time series forecasting on variable subsets. In: Proc. of KDD. pp. 76–86 (2022)
22. Longa, A., et al.: Graph neural networks for temporal graphs: State of the art, open challenges, and opportunities. arXiv preprint arXiv:2302.01018 (2023)
23. Cui, Y., et al.: Metro: a generic graph neural network framework for multivariate time series forecasting. Proc. VLDB Endow. **15**(2), 224–236 (2021)
24. Kong, X., et al.: Dynamic graph convolutional recurrent imputation network for spatiotemporal traffic missing data. Knowledge-Based Syst. **261**, 110188 (2023)
25. Vaswani, A.: Attention is all you need. Proc. of NeurIPS (2017)
26. Zheng, C., et al.: Robust graph representation learning via neural sparsification. In: Proc. of ICML. pp. 11458–11468. PMLR (2020)
27. Zhao, J., et al.: Heterogeneous graph structure learning for graph neural networks. In: Proc. of AAAI. vol. 35, pp. 4697–4705 (2021)
28. Xu, K., et al.: How powerful are graph neural networks? arXiv preprint arXiv:1810.00826 (2018)
29. Bagnall, A., et al.: The uea multivariate time series classification archive, 2018. arXiv preprint arXiv:1811.00075 (2018)