GFTLSTM: Dynamic Graph Neural Network Model Based on Graph Framelets Transform

Shengpeng Yang[1], Siwei Zhou[1], Shasha Yang[1(✉)], and Jiandong Shi[2]

[1] Key Laboratory of Intelligent Education Technology and Application of Zhejiang Province, Zhejiang Normal University, Jinhua, China
{yangshengp,yangss}@zjnu.edu.cn
[2] School of Computer Science and Technology (School of Artificial Intelligence), Zhejiang Normal University, Jinhua, China

Abstract. There is currently a surge of interest in graph representation learning, with researchers increasingly focusing on methods and applications involving graph neural networks (GNNs). However, traditional GNN models for static graph data analysis have limitations in their ability to extract evolutionary patterns from dynamic graphs, which are more commonly observed in real-world data. Additionally, existing dynamic GNN models tend to prioritize low-frequency information while neglecting high-frequency information. To address these limitations, we introduce a dynamic graph neural network model that leverages graph framelet transforms, capitalizing on the benefits of traditional wavelets in multi-resolution analysis. The initial step involves constructing the graph framelet transform based on graph wavelet research, subsequently implementing multi-resolution graph convolution with both low-pass and high-pass filtering. Then, we incorporate the convolution operation into a long short-term memory (LSTM) network. As a result, we develop a dynamic GNN model founded on the graph framelet transform, which effectively uncovers the evolutionary information embedded within dynamic graphs. In our experiment evaluation, we compare our model with 11 widely used dynamic graph representation learning algorithms across three public datasets of discrete dynamic graph representation learning tasks (encompassing six groups of experimental data). Our model outperforms the alternatives in terms of accuracy on the majority of the datasets.

Keywords: graph representation learning · dynamic graph · graph framelets transform

1 Introduction

With the rapid development of digital technologies such as the Internet, the Internet of Things, and artificial intelligence, trillions of bytes of data are generated every day across all industries [14]. These diverse and complex data record

M. Clayton et al. (Eds.): INTETAIN 2023, LNICST 560, pp. 63–75, 2024.
https://doi.org/10.1007/978-3-031-55722-4_6

various aspects of people's daily lives. Analyzing this data and mining potentially valuable information is crucial for scientific study, commerce, and other industries [12,32]. For example, in recommendation systems, users' preferences can be discovered through their browsing history, resulting in targeted product recommendations that increase shopping efficiency [8,13]. Similarly, in intelligent transportation systems, road conditions can be analyzed using vehicle-to-everything information to provide optimal path suggestions for destinations, ultimately saving users' time [5,7].

Since deep-learning-based data analysis methods have made significant advancements in academia, many models that use deep learning techniques to analyze and process different types of data are currently used in a wide range of industries. However, the majority of these models primarily focus on Euclidean structured data. In the era of big data, characterized by diverse data representations, there is an increasing prevalence of non-Euclidean structured graph data represented by social networks, telecommunication networks and transportation networks [11,22,27,31], which process data into graph forms and contain rich relational information.

Real-world graph data does not exhibit an unchangeable static graph structure; instead, it displays an extremely complex temporal evolution, as observed in social network graphs on Facebook and user-video interaction graphs on YouTube, etc. These data not only contain various individual entity features but also relationships between various users that change over time, creating a dynamic graph structure that has caught the interest of numerous academics. The work in [19] proposes the Evolving-GCN model which uses a recurrent neural network (RNN) to train the parameters of GCN to capture dynamic graph features. The authors of [21] introduce an attention mechanism into dynamic graph representation learning to construct the temporal and spatial information of graph evolution at different times. A K-Core-based model [16] is integrated into GCN and RNN to further consider the local similar information from different aspects in the same snapshot of dynamic graphs, thus improving the efficiency of subsequent tasks. The Netwalk model presented in [28] re-encodes dynamic graph data based on a deep self-encoder, updating node features in real-time using a graph random walk. It gathers nodes with similar features closely in space to perform graph clustering analysis. By utilizing a node labeling function, the StrGNN model in [3] incorporates closed subgraph sampling for neighbor nodes of the target node within a specific dynamic graph time window. This novel subgraph sampling approach aims to find the shortest path between the target node and the source node. By considering both the global and local information of dynamic graph neural networks in the same snapshot as well as dynamic evolution information in various snapshots, the authors in [17] achieve the best results to date on several dynamic graph anomaly detection benchmark datasets.

In this paper, we propose a representation learning method for dynamic graphs because the existing feature extraction methods for Euclidean structures do not apply to non-Euclidean structures and the proposed models based on static graph representation learning are unable to capture the evolution features in dynamic graphs. GFTLSTM is a dynamic graph neural network model based

on graph framelets transform. Given that wavelet analysis is more applicable to changing data, we attempt to combine wavelet analysis with dynamic graphs in the frequency domain before data reconstruction to achieve the effect of ReLU, after which we combine the long and short-term memory neural network model to capture the temporal features. We validate the effectiveness of our method by experimenting with two different parameter training methods on three benchmark datasets, finding that GFTLSTM outperforms the most dynamic graph neural network models in terms of performance. Furthermore, the ablation study and ReLU activation function comparison study also verify the model's performance. We summarize the contributions of this work as follows:

- We propose a dynamic graph neural network model based on graph framelets, representing an attempt to combine dynamic graphs with wavelet theory for processing node signals in the frequency domain.
- We consider both low-frequency and high-frequency information of the dynamic graph in the same snapshot, combining the low-pass filter with the high-pass filter and achieving activation functions in the frequency domain.
- As a new dynamic graph representation learning method, experiments on three public benchmarks demonstrate that GFTLSTM can capture dynamic information and outperform more current baseline models in dynamic graphs.

2 Dynamic Graphs

Based on how data is processed in time series, there are two main categories of dynamic graphs. The first category, discrete dynamic graphs, records graph data in fixed time snapshot as shown in Fig. 1(a); the second category, continuous dynamic graphs, records graph data in continuous time, as illustrated in Fig. 1(b). Based on the structure of discrete dynamic graph, they can be further classified into the following three types according to the dynamic characteristics of nodes and edges: (1) The first discrete dynamic graph is an ordered set of node features and graph structure, referred to as a static graph structure with a temporal signal, as shown in Fig. 2(a). In this type of dynamic graph, node features change over time, but the graph structure remains constant across all snapshots; (2) The second discrete dynamic graph is an ordered set of node features and

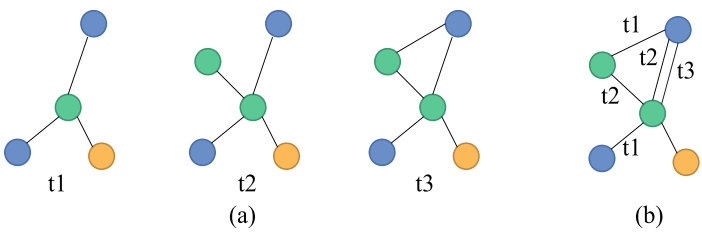

Fig. 1. (a) discrete dynamic graphs (b) continuous dynamic graphs.

graph structure, referred to as a dynamic graph structure with static features. In this case, node features remain unchanged, while the graph structure evolves over time, as seen in Fig. 2(b); (3) The third discrete dynamic graph is also an ordered set of node features and graph structure, wherein both node features and graph structure vary across different time snapshots. This type is known as a dynamic graph structure with a temporal signal, as shown in Fig. 2(c);

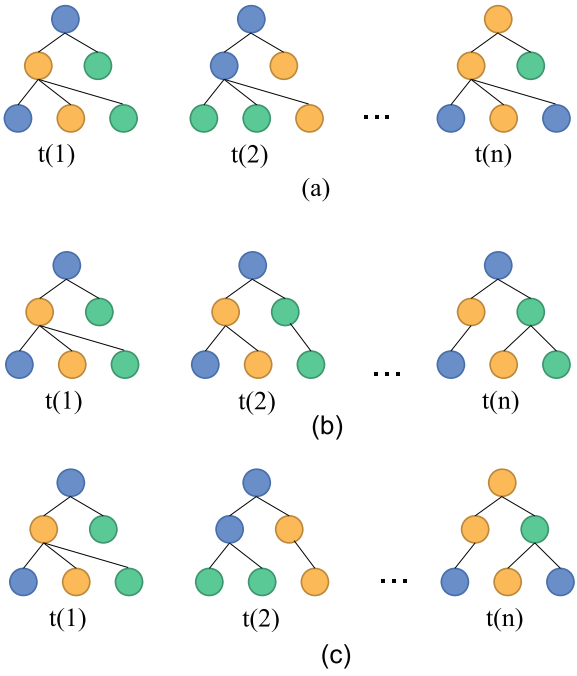

Fig. 2. (a) static graph with temporal signal (b) dynamic graph with static signal (c) dynamic graph with temporal signal.

3 Multi-resolution Framelets Analysis in Dynamic Graphs

In a specific discrete dynamic graph snapshot $\mathcal{G}_{\mathrm{t}} = (V_t, E_t, \omega)$, the graph transform framework employs several frequency domain filter sets $\eta = \{a; b^1, ..., b^n\}$, where a, b, n in the filter sets denote the number of low-passes, high-pass and high-pass filters, respectively, to extract approximate information and other details from the graph signal [10,30]. The eigenvalues and eigenvectors consisting of the N nodes in dynamic graph $\mathcal{G}_{\mathrm{t}}$ are obtained from the Laplacian matrix at each snapshot, denoted as $(\lambda_\ell, \mu_\ell)_j^N$, where j represents the degree of signal scaling when passing the signal into the frequency domain. When $n = 1, ..., r$,

the signal of node p is transformed by the low-pass filter $\varphi_{j,p}$ and high-pass filters $\phi^r_{j,p}$ in a snapshot can be expressed as follows

$$\varphi_{j,p}(v) = \sum_{\ell=1}^{N} \widehat{\alpha}\left(\frac{\lambda_\ell}{2^j}\overline{\mu_\ell(p)}\mu_\ell(v)\right) \tag{1}$$

$$\phi_{j,p}(v) = \sum_{\ell=1}^{N} \widehat{\beta^{(n)}}\left(\frac{\lambda_\ell}{2^j}\overline{\mu_\ell(p)}\mu_\ell(v)\right) \tag{2}$$

where we exploit the Chebyshev polynomial approximation for the filters $\widehat{\alpha}$ and $\widehat{\beta^{(n)}}$. The process of projecting the original signal f to $\varphi_{j,p}$ and $\phi^r_{j,p}$ can be denoted as $<\varphi_{j,p}, f>$ and $<\phi^r_{j,p}, f>$, respectively, to obtain the final wavelet coefficients $v_{j,p}$ and $w_{j,p}$. Since the data used for deep learning training is in tensor form, the transformation of the graph signal can be achieved by the signal decomposition operator $\mathcal{W}$ and the signal reconstruction operator $\mathcal{V}$, described as

$$\mathcal{W} = \{\mathcal{W}_{\mathrm{r,j}} \mid \mathrm{r} = 1, \ldots, \mathrm{n}; \mathrm{j} = 1, \ldots \mathrm{J}\} \cup \{\mathcal{W}_{0,\mathrm{j}}\} \tag{3}$$

$$\mathcal{W}_{\mathrm{r},1} f = \mathcal{T}_{\mathrm{r}}^{\mathrm{k}}\left(2^{-\mathrm{J}}\mathcal{L}\right) f \tag{4}$$

$$\mathcal{W}_{\mathrm{r,j}} f = \mathcal{T}_{\mathrm{r}}^{\mathrm{k}}\left(2^{\mathrm{K+j-1}}\mathcal{L}\right)\mathcal{T}_{0}^{\mathrm{k}}\left(2^{\mathrm{K+j-2}}\mathcal{L}\right)\cdots\mathcal{T}_{0}^{\mathrm{k}}\left(2^{-\mathrm{K}}\mathcal{L}\right) f \tag{5}$$

where $j = 2...J$, $\mathcal{T}_{\mathrm{r}}^{\mathrm{k}}$ is the r-degree Chebyshev polynomial, $\mathcal{L}$ is the graph Laplacian matrix, K is a constant satisfying $\lambda_{max} \leq 2^K\pi$, $\mathcal{W}_{0,\mathrm{J}} f = \{v_{J,p}\}_{p\in V_t}$ are the low-pass coefficients and $\mathcal{W}_{\mathrm{r,j}} f = \left\{w^r_{j,p}\right\}_{p\in V_t}$ are high-pass coefficients of f, r is the number of high-pass filters. When the signal in the frequency domain is reconstructed, the reconstruction operator $\mathcal{V}$ is an arrangement reorganization of $\mathcal{W}$.

4 Dynamic Graph Neural Network Model Based on Graph Framelets Transform

A graph neural network is a powerful deep learning method. Most existing graph neural networks, such as GCN [9] and GAT [26], are based on spatial modeling. These models use spatial message passing to calculate information from neighboring nodes, then converge and integrate information from source to target nodes through graph convolution techniques. They are trained by multilayer network stacking, ultimately achieving full graph feature learning. However, graph convolution models in the spatial domain are essentially low-pass filters and shallow graph convolution operations do not effectively propagate node labels. Deep graph convolution stacking can lead to excessive feature smoothing, and after multiple training iterations, similar information is sharpened more than once, making it difficult to distinguish between different classes of nodes. Using the Fourier transform to pass the signal into the frequency domain is another modeling technique. The spatial domain signal is projected onto the Fourier basis

to determine the magnitude of the signal in the spectrogram. After performing a series of operations, the signal is transmitted back into the spatial domain by the inverse Fourier transform indiscriminately. This frequency domain-based data analysis and modeling method has significant limitations for non-smooth signals with frequencies that change over time.

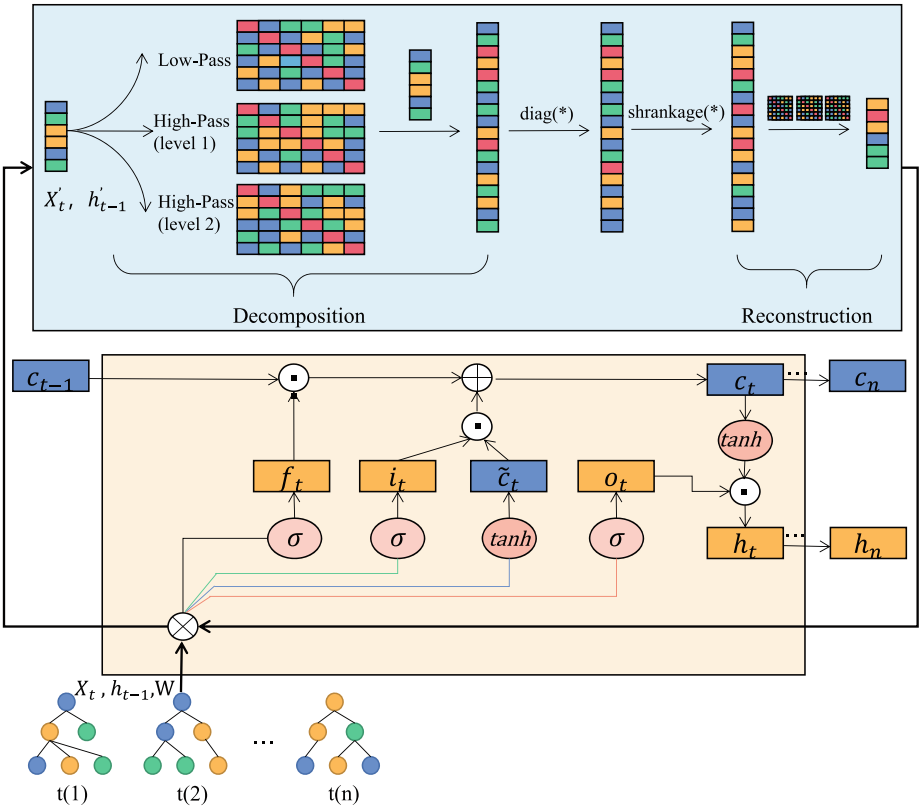

Fig. 3. The framework of GFTLSTM.

Due to the aforementioned shortcomings of modeling based on the spatial and frequency domains, we consider the wavelet transform, which is not only suitable for handling non-smooth signal characteristics (also a feature of dynamic graphs) but can also compensate for the drawbacks of the Fourier transform. We construct a frequency convolution model for dynamic graph signal extraction, GFTLSTM, based on the wavelet transform principle. The model structure is shown in Fig. 3. First, we project the spatial domain signal onto the wavelet basis, then filter the low-frequency and high-frequency signals differently and further compress the signal in the frequency space using the *Shrinkage* function

while denoising. The final step is to merge the filtering results with a long short-term memory neural network to capture the evolving features and relationships in dynamic graphs.

4.1 Multi-resolution Graph Convolution Based on Graph Framelets Transform

Based on the above graph framelets transform steps, we can perform convolution operations in the frequency domain space, formulated as follows:

$$\mathcal{V}\left(Shrinkage\left(diag\left(\theta\right)\left(\mathcal{W}X_t^{'}\right)\right)\right), X_t^{'} = X_t W \tag{6}$$

$$\mathcal{V}\left(Shrinkage\left(diag\left(\theta\right)\left(\mathcal{W}h_{t-1}^{'}\right)\right)\right), h_{t-1}^{'} = h_{t-1} W \tag{7}$$

where $X \in R^{N\times d}$ represents the node feature in dynamic graphs at the current moment t, $W \in d \times d^{'}$ denotes the trainable weight matrix, $X_t^{'}$ is the input feature matrix at this snapshot t after the transformation of the weight matrix W, θ is a network filter that multiplies each component value with the frequency domain framelet coefficients $\mathcal{W}X_t^{'}$ and $\mathcal{W}h_{t-1}^{'}$ to achieve the filtering operation, $Shrinkage$ denotes the signal compression function.

Activation functions for signal compression are typically applied in the spatial domain, which means the signal must be transferred back to the spatial domain without any loss before using the activation function. The conventional processing approach found in most activation functions does not meet our requirement for multi-resolution data analysis. In contrast, applying different filters to low-frequency and high-frequency signals in the frequency domain and then transferring them back to the spatial domain using the reconstruction operator accommodates the need for multi-resolution analysis while maintaining good performance. As shown in Fig. 3, three signal decomposition operators $\mathcal{W} = \{\mathcal{W}_{0,2}; \mathcal{W}_{1,1}, \mathcal{W}_{1,2}\}$ are obtained after portraying the signal with $j = 1, 2$ in a high-pass filter, followed by a low-pass filter and a high-pass filter. If the dimension of each decomposition operator is $N \times N$, then $W^{|X|} \in R^{3N\times d}$. After filtering the signal in the frequency domain through filter θ, the signal compression function $soft - shrinkage$ and $hard - shrinkage$ are used instead of the activation function to truncate the high-frequency components. The reconstruction operator $\mathcal{V} = (\mathcal{W})^T$ is then used to transfer the signal to the spatial domain. We perform the following signal compression function $Shrinkage$

$$Shrinkage - soft\left(x\right) = sgn\left(x\right)\left(|x| - \lambda\right)_+, \forall x \in R \tag{8}$$

$$Shrinkage - hard\left(x\right) = x\left(|x| - \lambda\right), \forall x \in R \tag{9}$$

4.2 Long Short-Term Memory Neural Network

One of the recurrent neural network models is the long short-term memory network (LSTM), which is primarily utilized to address gradient vanishing and gradient explosion issues during the training of long sequences [6]. We combine the

graph convolution operation with the long short-term memory neural network, the multiplication operation of the input matrix X_t and temporal information h_{t-1} with the weight matrix W in the long and short-term memory network is replaced by the multi-resolution graph convolution based on the graph framelets transform. The specific processing is shown as follows:

$$i_t = \sigma\left(W_{xi} * gX_t + W_{hi} * gh_{t-1} + w_{ci} \odot c_{t-1} + b_i\right) \tag{10}$$

$$f_t = \sigma\left(W_{xf} * gX_t + W_{hf} * gh_{t-1} + w_{ci} \odot c_{t-1} + b_f\right) \tag{11}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot tanh\left(W_{xc} * gX_t + W_{hc} * gh_{t-1} + b_c\right) \tag{12}$$

$$o_t = \sigma\left(W_{xo} * gX_t + W_{ho} * gh_{t-1} + w_{co} \odot c_{t-1} + b_o\right) \tag{13}$$

$$h_t = o_t \odot tanh\left(c_t\right) \tag{14}$$

where $*g$ represents the convolution operation, which is the graph framelets transform, b is the bias matrix, c_{t-1} represents the internal state of the previous snapshots, recording all the historical information in the time sequence up to the previous snapshot; i_t serves as an input gate to retain important data and reduce the amount of information being input for unimportant data; f_t acts as a forget gate, selectively forgetting information about the internal state of the previous snapshots, while remembering crucial information; c_t indicates the internal state at the current snapshot, updating the information that needs to be recorded. The role of the output gate o_t is to control how much information needs to be output to the external state h_t from the internal state c_t at the current snapshot.

5 Experiments

5.1 Datasets

To evaluate the performance of the proposed GFTLSTM, we conduct experiments using three public datasets: Chickenpox Hungary [20], Pedal Me Deliveries [20], Wikipedia Math [20].

Chickenpox Hungary: This is a spatiotemporal dataset about officially reported cases of chickenpox in Hungary between 2005 and 2015, with nodes being counties, edges indicating adjacencies between counties, and a prediction target of the number of illnesses in the following week.

Pedal Me Deliveries: This is a dataset about the number of weekly bicycle package deliveries by Pedal Me in London during 2020 and 2021. Nodes in the graph represent geographical units and edges are proximity-based mutual adjacency relationships.

Wikipedia Math: This is a dataset about the popular mathematics topics on Wikipedia where the edges denote the links from one page to another and the features describe the number of daily visits between 2019 and 2021 March.

5.2 Experiment Settings

We conduct experiments on each of the three dynamic graph datasets using two training methods: incremental and cumulative. For the incremental training method, we perform the $Shrinkage - hard$ function, while for the cumulative training method, we use the $Shrinkage - soft$ function.

Incremental Training Method: The losses and training weights are updated for each time series snapshot in the dynamic graph.

Cumulative Training Method: The losses in each temporal snapshot in the dynamic graph are accumulated and then backpropagated.

The hyperparameters in the model are summarized in Table 1, where *epoch* denotes the total number of training iterations; hiddenNN denotes the number of neurons in the intermediate layers; lr denotes the learning rate; $dropout(p)$ is a function that randomly deactivates the neurons with probability p; λ is used as a threshold value to control the affected signal components. In the high-frequency coefficients, the value of all signal components is set to 0 if their absolute value is less than the threshold value, thus compressing the signal.

Table 1. Experiment Settings.

Datasets	Training methods	Epoch	HiddenNN	Lr	Dropout(p)	λ
Chickenpox Hungary	Incremental	100	200	0.0001	P = 0.3	1e−4
Chickenpox Hungary	Cumulative	100	32	0.01	P = 0.3	1e−4
PedalMe London	Incremental	100	200	0.0001	P = 0.3	1e−2
PedalMe London	Cumulative	100	32	0.0001	P = 0.3	1e−4
Wikipedia Math	Incremental	100	200	0.0001	P = 0.3	1e−4
Wikipedia Math	Cumulative	100	32	0.01	P = 0.3	1e−4

5.3 Evaluation Metrics

We use mean squared error (MSE) and standard deviation as the evaluation metrics. MSE is the expected value of the squared difference between the sample estimate and the true sample value which can be defined as

$$L_q = \frac{1}{N} \sum_{N}^{i=1} (q_i - o_i)^2 \tag{15}$$

where $o = \{o_1, ..., o_N\}$ represents the model's output and q_i denotes the true sample values. The precision of the data is indicated by the standard deviation, which is a measure of the dispersion around the average value, described as

$$S = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \overline{x})^2}{n-1}} \tag{16}$$

where $\overline{x}$ denotes the sample average value and x_i is the i-th sample data. A high standard deviation indicates that most of the values are different from their average value, conversely, they are close to the average value.

5.4 Performance Comparison

Using supervised learning methods, the dynamic graph data are input into our model in a specific temporal sequence, and then the prediction results for the next snapshot are output. The experiments are repeated 10 times for each dataset under two different training methods, and the prediction results of other models are compared longitudinally. MSE is introduced to evaluate the performance of the model and the standard deviation is calculated for auxiliary validation. The experiment results are shown in Table 2.

Table 2. The predictive performance of GFTLSTM and other spatiotemporal neural networks evaluated by average mean squared error and standard deviations around the average mean squared error from 10 experiment repetitions.

	Chickenpox Hungary		PedalMe London		Wikipedia Math	
	Incremental	Cumulative	Incremental	Cumulative	Incremental	Cumulative
DCRNN [15]	1.124 ± 0.015	1.123 ± 0.014	1.463 ± 0.019	1.450 ± 0.024	0.679 ± 0.020	0.803 ± 0.018
GConvGRU [23]	1.128 ± 0.011	1.132 ± 0.023	1.622 ± 0.032	1.944 ± 0.013	$\mathbf{0.657 \pm 0.015}$	0.837 ± 0.021
GConvLSTM [23]	1.121 ± 0.014	1.119 ± 0.022	1.442 ± 0.028	1.433 ± 0.020	0.777 ± 0.021	0.868 ± 0.018
GC-LSTM [4]	1.115 ± 0.014	1.116 ± 0.023	1.455 ± 0.023	1.468 ± 0.025	0.779 ± 0.023	0.852 ± 0.016
DyGrAE [24,25]	1.120 ± 0.021	1.118 ± 0.015	1.455 ± 0.031	1.456 ± 0.019	0.073 ± 0.009	0.816 ± 0.016
EGCN-H [19]	1.113 ± 0.016	1.104 ± 0.024	1.467 ± 0.026	1.436 ± 0.017	0.775 ± 0.022	0.857 ± 0.022
EGCN-O [19]	1.124 ± 0.009	1.119 ± 0.020	1.491 ± 0.024	1.430 ± 0.023	0.750 ± 0.014	0.823 ± 0.014
A3T-GCN [1]	1.114 ± 0.008	1.119 ± 0.018	1.469 ± 0.027	1.475 ± 0.029	0.781 ± 0.011	0.872 ± 0.017
T-GCN [29]	1.117 ± 0.011	1.111 ± 0.022	1.479 ± 0.012	1.481 ± 0.029	0.764 ± 0.011	0.846 ± 0.020
MPNN LSTM [18]	1.116 ± 0.023	1.129 ± 0.021	1.485 ± 0.028	1.458 ± 0.013	0.795 ± 0.010	0.905 ± 0.017
AGCRN [2]	1.120 ± 0.010	1.116 ± 0.017	1.469 ± 0.030	1.465 ± 0.026	0.788 ± 0.011	0.832 ± 0.020
GFTLSTM	$\mathbf{1.099 \pm 0.001}$	$\mathbf{1.067 \pm 0.012}$	$\mathbf{1.437 \pm 0.027}$	$\mathbf{1.401 \pm 0.096}$	0.708 ± 0.013	$\mathbf{0.653 \pm 0.027}$

The bolded values in Table 2 indicate the best experiment results of the various prediction models. It can be observed that the GFTLSTM model proposed in this paper outperforms most of the comparison models in processing discrete dynamic graph data and achieves the best results in most experiments. Although the GFTLSTM model doesn't achieve state-of-the-art results in the Wikipedia Math dataset using the incremental training method, it still outperforms the majority of the models. This demonstrates that the GFTLSTM model can better capture the evolutionary features of dynamic graphs, providing a valuable reference for dynamic graph representation learning.

5.5 Ablation Study

Ablation experiments are conducted to verify the effectiveness of the *Shrinkage* function introduced in the frequency domain space to replace the spatial domain

activation function ReLU in the model. In these experiments, GFTLSTM-S represents the model proposed in this paper after removing the $Shrinkage - soft$ and $Shrinkage - hard$ functions. After conducting 10 trials, the experiment results are presented in Table 3.

Table 3. The predictive performance of GFTLSTM-S evaluated using average mean squared error and standard deviations around the average mean squared error from 10 experiment repetitions.

	Chickenpox Hungary		PedalMe London		Wikipedia Math	
	Incremental	Cumulative	Incremental	Cumulative	Incremental	Cumulative
GFTLSTM-S	1.103 ± 0.002	1.078 ± 0.009	1.449 ± 0.020	1.569 ± 0.167	0.717 ± 0.019	0.669 ± 0.017

As shown in the table above, the MSE of the experiment results exhibits varying degrees of improvement after removing the *Shrinkage* function in the frequency domain space. This result indicates that the model performance becomes worse, which verifies the usefulness of introducing the *Shrinkage* function to the model.

5.6 Comparison Experiment

To further verify the effectiveness of multi-resolution analysis in the model, the ReLU activation function is introduced in the spatial domain and the *Shrinkage* function in the frequency domain is removed to obtain the GFTLSTM_R model. The experiment results are shown in Table 4 after 10 repetitions.

Table 4. The predictive performance of GFTLSTM_R evaluated by average mean squared error and standard deviations around the average mean squared error from 10 experiment repetitions.

	Chickenpox Hungary		PedalMe London		Wikipedia Math	
	Incremental	Cumulative	Incremental	Cumulative	Incremental	Cumulative
GFTLSTM_R	1.103 ± 0.003	1.080 ± 0.012	1.472 ± 0.031	1.466 ± 0.119	0.720 ± 0.014	0.658 ± 0.018

As shown in Table 4, a comparison of the performance of the GFTLSTM model as detailed in Table 2 shows that the value of GFTLSTM_R in terms of MSE or standard deviation metrics also increases, indicating that the experiment results become worse again. Therefore, for the discrete dynamic graph dataset, better results are obtained by using the *Shrinkage* function in the frequency domain space instead of ReLU as the activation function. This verifies the value of using the *Shrinkage* function for multi-resolution analysis in dynamic graphs.

6 Conclusion

In this work, we propose a general framework for dynamic graph representation learning, representing a first attempt to combine wavelet analysis theory, multi-resolution analysis and dynamic graphs. We evaluate the model's effectiveness on several dynamic graph datasets. In future work, we hope to apply the model to dynamic graph structures with static features, dynamic graph structures with temporal signals, and continuous dynamic graphs within specific application scenarios. This will further verify the model's generality across various dynamic graph representation learning tasks.

References

1. Bai, J., et al.: A3T-GCN: attention temporal graph convolutional network for traffic forecasting. ISPRS Int. J. Geo Inf. **10**(7), 485 (2021)
2. Bai, L., Yao, L., Li, C., Wang, X., Wang, C.: Adaptive graph convolutional recurrent network for traffic forecasting. Adv. Neural. Inf. Process. Syst. **33**, 17804–17815 (2020)
3. Cai, L., et al.: Structural temporal graph neural networks for anomaly detection in dynamic graphs. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, Changsha, pp. 3747–3756. ACM (2021)
4. Chen, J., Wang, X., Xu, X.: GC-LSTM: graph convolution embedded LSTM for dynamic network link prediction. Appl. Intell., 1–16 (2022)
5. Grattarola, D., Alippi, C.: Graph neural networks in TensorFlow and keras with spektral. IEEE Comput. Intell. Mag. **16**(1), 99–106 (2021)
6. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
7. Javaid, S., Sufian, A., Pervaiz, S., Tanveer, M.: Smart traffic management system using internet of things. In: Proceedings of the 20th International Conference on Advanced Communication Technology, Wonju, pp. 393–398. Springer (2018)
8. Karimi, M., Jannach, D., Jugovac, M.: News recommender systems-survey and roads ahead. Inf. Process. Manage. **54**(6), 1203–1227 (2018)
9. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2017)
10. Li, M., Ma, Z., Wang, Y.G., Zhuang, X.: Fast Haar transforms for graph neural networks. Neural Netw. **128**, 188–198 (2020)
11. Li, M., Sonoda, S., Cao, F., Wang, Y.G., Liang, J.: How powerful are shallow neural networks with bandlimited random weights? In: International Conference on Machine Learning, pp. 19960–19981. PMLR (2023)
12. Li, M., Wang, D.: 2-d stochastic configuration networks for image data analytics. IEEE Trans. Cybern. **51**(1), 359–372 (2021)
13. Li, M., Zhang, L., Cui, L., Bai, L., Li, Z., Wu, X.: BLoG: bootstrapped graph representation learning with local and global regularization for recommendation. Pattern Recogn. **144**, 109874 (2023)
14. Li, S., Xu, L.D., Zhao, S.: The internet of things: a survey. Inf. Syst. Front. **17**, 243–259 (2015)
15. Li, Y., Yu, R., Shahabi, C., Liu, Y.: Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In: Proceedings of the 6th International Conference on Learning Representations. OpenReview.net, Vancouver (2018)

16. Liu, J., Xu, C., Yin, C., Wu, W., Song, Y.: K-core based temporal graph convolutional network for dynamic graphs. IEEE Trans. Knowl. Data Eng. **34**(8), 3841–3853 (2020)
17. Liu, Y., et al.: Anomaly detection in dynamic graphs via transformer. IEEE Trans. Knowl. Data Eng. **01**, 1 (2021)
18. Panagopoulos, G., Nikolentzos, G., Vazirgiannis, M.: Transfer graph neural networks for pandemic forecasting. In: Proceedings of the 35th AAAI Conference on Artificial Intelligence, California, pp. 4838–4845. AAAI Press (2021)
19. Pareja, A., et al.: EvolveGCN: evolving graph convolutional networks for dynamic graphs. In: Proceedings of the 34th AAAI Conference on Artificial Intelligence, New York, pp. 5363–5370. AAAI Press (2020)
20. Rozemberczki, B., et al.: Pytorch geometric temporal: Spatiotemporal signal processing with neural machine learning models. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, Changsha, pp. 4564–4573. ACM (2021)
21. Sankar, A., Wu, Y., Gou, L., Zhang, W., Yang, H.: DySAT: deep neural representation learning on dynamic graphs via self-attention networks. In: Proceedings of the 13th International Conference on Web Search and Data Mining, Houston, pp. 519–527. ACM (2020)
22. Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. IEEE Trans. Neural Networks **20**(1), 61–80 (2008)
23. Seo, Y., Defferrard, M., Vandergheynst, P., Bresson, X.: Structured sequence modeling with graph convolutional recurrent networks. In: Cheng, L., Leung, A.C.S., Ozawa, S. (eds.) ICONIP 2018. LNCS, vol. 11301, pp. 362–373. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-04167-0_33
24. Taheri, A., Berger-Wolf, T.: Predictive temporal embedding of dynamic graphs. In: Proceedings of the 9th IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, Vancouver, pp. 57–64. IEEE (2019)
25. Taheri, A., Gimpel, K., Berger-Wolf, T.: Learning to represent the evolution of dynamic graphs with recurrent models. In: Proceedings of the 28th World Wide Web Conference, Portland, pp. 301–307. ACM (2019)
26. Velickovic, P., et al.: Graph attention networks. Stat **1050**(20), 10–48550 (2017)
27. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y.: A comprehensive survey on graph neural networks. IEEE Trans. Neural Netw. Learning Syst. **32**(1), 4–24 (2020)
28. Yu, W., Cheng, W., Aggarwal, C.C., Zhang, K., Chen, H., Wang, W.: NetWalk: a flexible deep embedding approach for anomaly detection in dynamic networks. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, pp. 2672–2681. ACM (2018)
29. Zhao, L., et al.: T-GCN: a temporal graph convolutional network for traffic prediction. IEEE Trans. Intell. Transp. Syst. **21**(9), 3848–3858 (2019)
30. Zheng, X., et al.: How framelets enhance graph neural networks. In: Proceedings of the 38th International Conference on Machine Learning, Graz, pp. 12761–12771. PMLR (2021)
31. Zhou, H., She, C., Deng, Y., Dohler, M., Nallanathan, A.: Machine learning for massive industrial internet of things. IEEE Wirel. Commun. **28**(4), 81–87 (2021)
32. Zhou, J., et al.: Graph neural networks: a review of methods and applications. AI Open **1**, 57–81 (2020)