TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 13/18 pp141–149
DOI: 10.26599/TST.2021.9010081
Volume 28, Number 1, February 2023

Multivariate Time Series Forecasting with Transfer Entropy Graph

Ziheng Duan, Haoyan Xu, Yida Huang, Jie Feng, and Yueyang Wang*

Abstract: Multivariate Time Series (MTS) forecasting is an essential problem in many fields. Accurate forecasting results can effectively help in making decisions. To date, many MTS forecasting methods have been proposed and widely applied. However, these methods assume that the predicted value of a single variable is affected by all other variables, ignoring the causal relationship among variables. To address the above issue, we propose a novel end-to-end deep learning model, termed graph neural network with neural Granger causality, namely CauGNN, in this paper. To characterize the causal information among variables, we introduce the neural Granger causality graph in our model. Each variable is regarded as a graph node, and each edge represents the casual relationship between variables. In addition, convolutional neural network filters with different perception scales are used for time series feature extraction, to generate the feature of each node. Finally, the graph neural network is adopted to tackle the forecasting problem of the graph structure generated by the MTS. Three benchmark datasets from the real world are used to evaluate the proposed CauGNN, and comprehensive experiments show that the proposed method achieves state-of-the-art results in the MTS forecasting task.

Key words: Multivariate Time Series (MTS) forecasting; neural Granger causality graph; Transfer Entropy (TE)

1 Introduction

In the real world, Multivariate Time Series (MTS) data are common in various fields[1, 2], such as the sensor data in the Internet of things, traffic flows on highways, and the prices collected from stock markets (e.g., metal price)[3, 4]. In recent years, many time series forecasting methods have been widely studied and applied[5]. For univariate situations, the AutoRegressive Integrated Moving Average model (ARIMA)[6] is one of the most classic forecasting methods. However, due to the computational complexity, ARIMA is not suitable for multivariate situations. VAR[6–8] method is an extended multivariate version of the AR model, which is widely used in MTS forecasting tasks due to its simplicity. However, it cannot handle the nonlinear relationships that exist among variables, which consequently reduces its forecasting accuracy.

In addition to traditional statistical methods, deep learning methods are also applied for the MTS forecasting problem[9]. The Recurrent Neural Network (RNN)[10] and its two improved versions, namely the Long Short-Term Memory (LSTM)[11] and the Gated Recurrent Unit (GRU)[12], realize the extraction of time series dynamic information through the memory mechanism. LSTNet[13] encodes short-term local information into low-dimensional vectors using 1D convolutional neural networks, and decodes the vectors through an RNN. However, the existing deep learning methods cannot model the pairwise causal dependencies among MTS variables explicitly. For example, the future traffic flow of a specific street is easier to be influenced and predicted by the traffic information of the neighboring area. In contrast, the knowledge of

• Ziheng Duan and Yueyang Wang are with School of Big Data and Software Engineering, Chongqing University, Chongqing 401331, China. E-mail: duanziheng@zju.edu.cn; yueyangw@cqu.edu.cn.
• Haoyan Xu, Yida Huang, and Jie Feng are with the College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China. E-mail: {haoyanxu, stevenhuang, zjucse_fj}@zju.edu.cn.
∗ To whom correspondence should be addressed.
Manuscript received: 2021-04-24; revised: 2021-09-23; accepted: 2021-10-22

the area farther away is relatively useless[14]. If such a prior causal information can be considered, it is more conducive to the interaction among variables with causality.

Granger causality analysis (G-causality)[15, 16] is one of the most famous studies on the quantitative characterization of time series causality. However, as a linear model, G-causality cannot handle nonlinear relationships well. Thus, Transfer Entropy (TE)[17] is proposed for causal analysis, which can deal with nonlinear cases. TE has been widely used in economic[18], biological[19], and industrial[20] fields.

To further address the above limitation, we propose a novel framework, called Graph Neural Network with Causality, namely CauGNN, for MTS forecasting tasks. After calculating the pairwise TE among variables, the TE matrix can be obtained, which is regarded as the adjacency matrix of the graph structure, and each variable in this matrix corresponds to one node of this graph. In addition, CNN filters with different perception scales are used for time series feature extraction to generate the feature of each node. Finally, the Graph Neural Network (GNN) is adopted to tackle the embedding and forecasting problem of the graph generated by MTS. Our major contributions are as follows:

- To the best of the authors' knowledge, we first propose an end-to-end deep learning framework that considers MTS as a graph structure with causality.
- We use TE to extract the causality among the time series and construct the TE graph as a priori information to guide the forecasting task.
- We conduct extensive experiments on MTS benchmark datasets, and results have proved that CauGNN outperforms the state-of-the-art models.

2 Preliminary

2.1 Neural Granger causality

Neural Granger is an improved Granger causality inference method. It inherits the core idea of Granger causality, i.e., if the addition of historical information of variable i significantly improves the prediction accuracy of another variable j, then variable i is the cause of variable j, and vice versa. The difference is that the traditional linear Granger method uses the AR model for prediction. In contrast, the neural Granger uses deep learning and regularization to account for the nonlinearity and avoid the computational complexity caused by the pairwise calculation.

The neural Granger network structure consists of two parts: (1) the variable selection module and (2) prediction module. The variable selection module is a fully connected layer that directly accepts the historical time series as input. The neural Granger method selects key variables by adding group Lasso regularization constraints to the weight parameters of this layer. Group Lasso is an evolved version of Lasso regularization, which can divide constrained parameters into multiple subgroups. If a specific group is insignificant for prediction, the entire group of parameters will be assigned a zero value. In the variable selection module, the weights connected to an input variable are set at different time points as a group. If the weights of the subgroup are not zero under the regularization constraint, it means that the variable has a significant effect on the prediction and is thus determined as the cause of the variable to be predicted. The second part of the neural Granger is the prediction layer. This part is not significantly different from the general prediction method. Networks, such as the multilayer perceptron or LSTM, can be used. For each variable $\boldsymbol{x}_i$, a neural Granger network is established to find its cause variables. The objective function of the network is as follows:

$$\min_{W} \sum_{t=K}^{T} (x_{it} - g_i(\boldsymbol{x}_{(t-1):(t-K)}))^2 + \lambda \sum_{j=1}^{p} \left\| (\boldsymbol{W}_{:j}^{11}, \boldsymbol{W}_{:j}^{12}, \ldots, \boldsymbol{W}_{:j}^{1K}) \right\|_{\mathrm{F}} \tag{1}$$

where x_{it} is the true value of the variable $\boldsymbol{x}_i$ at time t, $\boldsymbol{x}_{(t-1):(t-K)}$ is the value of all variables in K lags, $g_i()$ is the function that specifies how lags from 1 to K affect the future evolution of the series, T is the observed time points, p is the number of variables, λ is the regularization coefficient, $\boldsymbol{W}_{:j}^{11}, \ldots, \boldsymbol{W}_{:j}^{1K}$ represents all the weight parameters connected with the j-th variable in the variable selection module, and $\| \|_{\mathrm{F}}$ is the F-norm.

2.2 GNN

The concept of GNN was first proposed in Ref. [21], which extended existing neural networks for processing the data represented in graph domains. A wide variety of GNN models have been proposed in recent years[22, 23]. Most of these approaches fit within the framework of "neural message passing" proposed by Gilmer et al.[24]. In the message-passing framework, a GNN is viewed as a message-passing algorithm where node representations are iteratively computed from the features of their neighbor nodes using a differentiable aggregation function[25–27].

A separate line of work focuses on generalizing convolutions to graphs. The Graph Convolutional Networks (GCN)[28] could be regarded as an approximation of the spectral-domain convolution of graph signals. GCN convolutional operation could also be viewed as the sampling and aggregating of the neighborhood information, such as GraphSAGE[29] and FastGCN[30], enabling training in batches while sacrificing some time efficiency. Coming right after GCN, the Graph Isomorphism Network (GIN)[31] and k-GNNs[32] are developed, enabling more complex forms of aggregation. The Graph Attention Network (GAT)[33] is another nontrivial direction to go under the topic of GNN. It incorporates attention into propagation, attending over the neighbors via self-attention. Recently, researchers have also applied GNN to the time series forecasting problem. For example, a Correlational Graph Attention-based Long Short-Term Memory network (CGA-LSTM) proposed in Ref. [34] shows comparable performance. This further reminds us of the superiority of the graph method in MTS forecasting.

3 Methodology

3.1 Problem formulation

Given a matrix consisting of multiple observed time series $\boldsymbol{S}_n = [\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_t]$, where $\boldsymbol{s}_i \in \mathbf{R}^n (i = 1, \ldots, n)$ and n is the number of variables, the goal of MTS forecasting is to predict $\boldsymbol{s}_{t+h}$, where h is the horizon ahead of the current time stamp.

3.2 Causality graph structure with TE

TE is a measure of causality based on information theory, which was proposed by Schreiber in 2000[35]. Given a variable $\boldsymbol{X} \in \mathbf{R}^t$, its information entropy is defined as

$$H(\boldsymbol{X}) = -\sum p(x) \log_2 p(x) \tag{2}$$

where x denotes all possible values of the variable, and $p(x)$ is the corresponding probability. Information entropy is used to measure the amount of information. A larger $H(\boldsymbol{X})$ indicates that the variable $\boldsymbol{X}$ contains more information. Conditional entropy is another information theory concept, given two variables $\boldsymbol{X}$ and $\boldsymbol{Y}$, which is defined as

$$H(\boldsymbol{X}|\boldsymbol{Y}) = -\sum\sum p(x, y) \log_2 p(x|y) \tag{3}$$

where conditional entropy $H(\boldsymbol{X}|\boldsymbol{Y})$ represents the information amount of $\boldsymbol{X}$ under the condition that the variable $\boldsymbol{Y}$ is known. The TE of $\boldsymbol{Y}$ to $\boldsymbol{X}$ is defined as

$$\begin{aligned}T_{Y\to X} &= \sum p(x_{t+1}, \boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t^{(l)}) \log_2 p(x_{t+1}|\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t^{(l)}) - \\ &\sum p(x_{t+1}, \boldsymbol{x}_t^{(k)}) \log_2 p(x_{t+1}|\boldsymbol{x}_t^{(k)}) = \\ &\sum p(x_{t+1}, \boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t^{(l)}) \log_2 \frac{p(x_{t+1}|\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t^{(l)})}{p(x_{t+1}|\boldsymbol{x}_t^{(k)})} = \\ &H(\boldsymbol{X}_{t+1}|\boldsymbol{X}_t) - H(\boldsymbol{X}_{t+1}|\boldsymbol{X}_t, \boldsymbol{Y}_t)\end{aligned} \tag{4}$$

where x_t and y_t represent their values at time t. $\boldsymbol{x}_t^{(k)} = [x_t, x_{t-1}, \ldots, x_{t-k+1}]$ and $\boldsymbol{y}_t^{(l)} = [y_t, y_{t-1}, \ldots, y_{t-l+1}]$. It can be found that TE is an increase in the information amount of the variable $\boldsymbol{X}$ when $\boldsymbol{Y}$ changes from being unknown to known. TE indicates the direction of information flow, thus characterizing causality. It is worth noting that TE is asymmetric, so the causal relationship between $\boldsymbol{X}$ and $\boldsymbol{Y}$ is usually further indicated in the following way:

$$T_{X,Y} = T_{X\to Y} - T_{Y\to X} \tag{5}$$

When $T_{X,Y}$ is greater than 0, it means that $\boldsymbol{X}$ is the cause of $\boldsymbol{Y}$. Otherwise $\boldsymbol{X}$ is the consequence of $\boldsymbol{Y}$. In this paper, we use neural Granger to characterize the causal relationship among variables. The causality matrix $\boldsymbol{A}$ of the MTS $\boldsymbol{S}_n$ can be formulated with the element a_{ij} corresponding to the i-th row and j-th column as

$$a_{ij} = \begin{cases} T_{v_i, v_j}, & T_{v_i, v_j} > c; \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

where $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ are the i-th and j-th rows of $\boldsymbol{S}_n$, i.e., i-th and j-th variables of $\boldsymbol{S}_n$, c is the threshold to determine whether the causality is significant. $\boldsymbol{A}$ can be regarded as the adjacency matrix of the MTS graph structure.

3.3 Feature extraction of multiple receptive fields

Time series is a special kind of data. When analyzing time series, it is necessary to consider not only its numerical value but also its trend over time. In addition, time series from the real world often has multiple meaningful periods. For example, the traffic flow of a certain street shows a similar trend every day and meaningful rules can be observed in the unit of a week. Therefore, it is reasonable to extract the features of time series in units of multiple certain periods. In this paper, we use multiple CNN filters with different receptive fields, namely kernel sizes, to extract features at multiple time scales. Given an input time series $\boldsymbol{v}$ and q CNN filters, denoted as $\boldsymbol{W}_i$, different convolution kernel sizes $(1 \times k_i)(i = 1, 2, \ldots, q)$ are separately generated and the features $\boldsymbol{h}$ are extracted as follows: $\boldsymbol{h}_i = ReLU(\boldsymbol{W}_i * \boldsymbol{v} + \boldsymbol{b}_i), \boldsymbol{h} = [\boldsymbol{h}_1 \oplus \boldsymbol{h}_2 \oplus \cdots \oplus \boldsymbol{h}_q]$. "$*$" denotes the convolution operation, "$\oplus$" represents the concatenate operation, and $ReLU()$ is a nonlinear activation function, $ReLU(\cdot) = \max(0, \cdot)$.

3.4 Node embedding based on causality matrix

After feature extraction, the input MTS is converted into a feature matrix $\boldsymbol{H} \in \mathbf{R}^{n \times d}$, where d is the number of features after the calculation introduced in Section 3.3. $\boldsymbol{H}$ can be regarded as a feature matrix of a graph with n nodes. The adjacency of nodes in the graph structure is determined by the causality matrix $\boldsymbol{A}$. For such a graph structure, GNNs can be directly applied for the embedding of nodes. Inspired by the k-GNNs model[32], we propose the CauGNN model and use the following propagation mechanism for calculating the forward-pass update of a node denoted by $\boldsymbol{v}_i$:

$$\boldsymbol{h}_i^{(l+1)} = \sigma\left(\boldsymbol{h}_i^{(l)}\boldsymbol{W}_1^{(l)} + \sum_{j \in \boldsymbol{N}(i)} \boldsymbol{h}_j^{(l)}\boldsymbol{W}_2^{(l)}\right) \tag{7}$$

where σ denotes the activation function (e.g., sigmoid function), $\boldsymbol{W}_1^{(l)}$ and $\boldsymbol{W}_2^{(l)}$ are parameter matrices, $\boldsymbol{h}_i^{(l)}$ is the hiden state of node i in the l-th layer, and $\boldsymbol{N}(i)$ denotes the neighbors of node i. k-GNNs only perform information fusion between a certain node and its neighbors, ignoring the information of other nonneighbor nodes. This design highlights the relationship among variables, which can effectively avoid the information redundancy brought by high dimensions. By adding a priori causal information obtained by TE, the model does not need to determine the key variables for forecasting by itself. In this paper, the output dimension of the last GNN layer is 1, which is used as the prediction result.

Overall, we use ℓ_1-norm loss to measure the prediction of s_{t+h} and optimize the model via the Adam algorithm[36]. The schematic diagram of CauGNN is shown in Fig. 1.

4 Experiment

In this section, we conduct extensive experiments on three benchmark datasets for MTS forecasting tasks and compare the results of the proposed CauGNN model with the other six baselines. All the data and experiment codes are available online†.

4.1 Data

We use three publicly available benchmark datasets.

- **Exchange-Rate**‡: Exchange rates of eight foreign countries collected from 1990 to 2016, collected per day.
- **Energy**[37]: Measurements of 26 different quantities related to the appliances' energy consumption in a single house for 4.5 months, collected per 10 minutes.
- **Nasdaq**[38]: Stock prices selected as the multivariable time series for 82 corporations, collected per minute.

4.2 Methods for comparison

The methods in the comparative evaluation are as follows:

- **VAR**[6–8] stands for the well-known vector regression model, which has proven to be a useful machine learning method for MTS forecasting.
- **CNN-AR**[39] stands for the classical CNN. We use multilayer CNN with AR components to perform MTS

† https://github.com/RRRussell/CauGNN.
‡ https://github.com/laiguokun/multivariate-time-series-data.

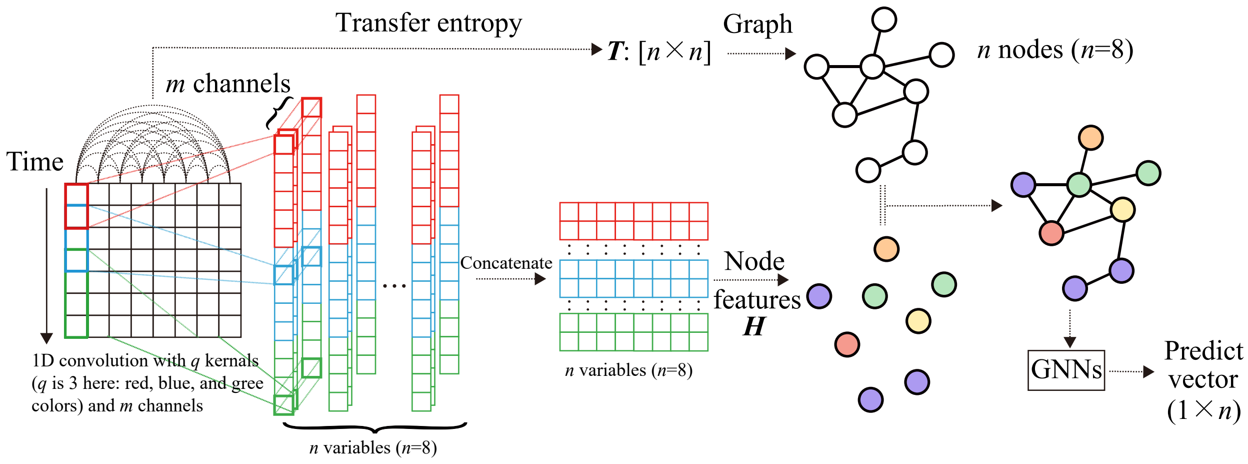

Fig. 1 Schematic diagram of CauGNN. A multivariate time series consists of multiple univariate time series. CauGNN maps a multivariate time series to a graph, and each univariate time series (variable) is mapped to a node. The causality matrix is calculated to model the adjacency information of nodes, while the convolutional layer is used to catch node features. The node feature matrix and adjacency matrix are then fed into the GNN to get forecasts.

forecasting tasks.

- **RNN-GRU**[12] is the RNN using the GRU cell with AR components.
- **MultiHead Attention**[40] stands for multihead attention components in the famous Transformer model, where the multihead mechanism runs through the scaled dot-product attention multiple times in parallel.
- **LSTNet**[13] is a famous MTS forecasting framework that shows great performance by modeling long- and short-term temporal patterns of MTS data.
- **MLCNN**[41] is a novel multi-task deep learning framework that adopts the idea of fusing forecasting information of different future times.
- **CauGNN** stands for our proposed GNN with TE. We apply the multilayer CNN and k-GNNs to perform MTS forecasting tasks.
- **CauGIN** stands for the proposed graph isomorphism network with TE, where k-GNNs layers are replaced by GIN layers.
- **CauGNN-nCau** removes the TE matrix and uses an all-one adjacency matrix instead.
- **CauGNN-nCNN** removes the CNN component and uses the input time series data as node features.

4.3 Metrics

We apply three conventional evaluation metrics to evaluate the performance of different models for MTS prediction. The Mean Absolute Error (MAE), Relative Absolute Error (RAE), and empirical correlation coefficient (CORR) are calculated as follows:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|p_i - a_i|,$$

$$\text{RAE} = \frac{\sum\limits_{i=1}^{n}|p_i - a_i|}{\sum\limits_{i=1}^{n}|\bar{a} - a_i|}s,$$

$$\text{CORR} = \frac{\sum\limits_{i=1}^{n}(p_i - \bar{p})(a_i - \bar{a})}{\sqrt{\sum\limits_{i=1}^{n}(p_i - \bar{p})^2}\sqrt{\sum\limits_{i=1}^{n}(a_i - \bar{a})^2}} \tag{8}$$

where a is the actual target and p is the predict target.

For MAE and RAE metrics, a lower value is better; while a higher value is better for CORR metric.

4.4 Experiment details

We conduct a grid search for the tunable hyper-parameters of each method in all datasets. Specifically, we set the same grid search range of input window size for each method from $\{2^0, 2^1, \ldots, 2^9\}$ if applied. We vary hyper-parameters for each baseline method to achieve their best performance on this task. For RNN-GRU and LSTNet, the hidden dimension of the recurrent and convolutional layer is chosen from $\{10, 20, \ldots, 100\}$. For LSTNet, the skip-length is chosen from $\{0, 12, \ldots, 48\}$. For MLCNN, the hidden dimension of the recurrent and convolutional layer is chosen from $\{10, 25, 50, 100\}$. We adopt the dropout layer after each layer, and set the dropout rate from $\{0.1, 0.2\}$. We calculate the TE matrix based on the training and validation data. For CauGNN, CauGIN, CauGNN-nCau, and CauGNN-nCNN, we all set the size of the three convolutional kernels to be $\{3, 5, 7\}$, and the number of channels of each kernel is 12 in all our models. The hidden dimension of the k-GNNs layer is chosen from $\{10, 20, \ldots, 100\}$. For CauGIN, the hidden size is chosen from $\{10, 20, \ldots, 100\}$. For the hyperparameter c, which is the threshold to determine the significance of the causality, we search it in the range of [0, 0.1], and choose it to be 0.005. The Adam algorithm is used to optimize the parameters of the proposed model. For more details, please refer to the attached code.

4.5 Main results

Table 1 summarizes the evaluation results of all methods on the three benchmark datasets with three metrics. Following the test settings of Ref. [13], we use each model for the time series predicting on the future moment $\{t+5, t+10, t+15\}$; thus, we set horizon= $\{5, 10, 15\}$, which means the horizon is set from 5 to 15 days for forecasting over the exchange-rate data, from 50 to 150 minutes over the energy data, and from 5 to 15 minutes over the Nasdaq data. The best results for each metric on each dataset are set as bold in Table 1. We save the model that has the best performance on the validation set based on the RAE or MAE metric after training 1000 epochs for each method. Then we use the model to test and record the results. The results show that the proposed CauGNN model outperforms most of the baselines in most cases, indicating the effectiveness of the proposed model on MTS predicting tasks adopting the idea of using causality as the guideline for forecasting. On the other side, we observe the results of the VAR model on the Nasdaq dataset are far worse than those of other methods in some cases, partly because VAR is insensitive to the scale of input data, lowering its performance.

MLCNN shows impressive results because it can fuse near and distant future visions, and LSTNet shows

Table 1 MTS forecasting results measured by MAE/RAE/CORR score over three datasets.

Dataset	Horizon	Method								
		VAR			CNN-AR			RNN-GRU		
		MAE	RAE	CORR	MAE	RAE	CORR	MAE	RAE	CORR
Exchange-rate	5 days	0.0065	0.0188	0.9619	0.0063	0.0182	0.9638	0.0066	0.0192	0.9630
	10 days	0.0093	0.0270	0.9470	0.0085	0.0249	0.9490	0.0092	0.0268	0.9491
	15 days	0.0116	0.0339	0.9318	0.0106	0.0303	0.9372	0.0122	0.0355	0.9323
Energy	50 min	3.1628	0.0545	0.9106	2.4286	0.0419	0.9159	2.7306	0.0471	0.9167
	100 min	4.2154	0.0727	0.8482	2.9499	0.0509	0.8618	3.0590	0.0528	0.8624
	150 min	5.1539	0.0889	0.7919	3.5719	0.0616	0.8150	3.7150	0.0641	0.8106
Nasdaq	5 min	0.1706	0.0011	0.9911	0.2110	0.0014	0.9920	0.2245	0.0015	0.9930
	10 min	0.2667	0.0018	0.9273	0.2650	0.0017	0.9919	0.2313	0.0015	0.9901
	15 min	0.3909	0.0026	0.5528	0.2663	0.0017	0.9860	0.2700	0.0018	0.9877

Dataset	Horizon	Method								
		MultiHead Attention			LSTNet			MLCNN		
		MAE	RAE	CORR	MAE	RAE	CORR	MAE	RAE	CORR
Exchange-rate	5 days	0.0078	0.0227	0.9630	0.0063	0.0184	0.9639	0.0065	0.0189	0.9693
	10 days	0.0101	0.0294	0.9500	0.0085	0.0247	0.9490	0.0094	0.0274	**0.9559**
	15 days	0.0119	0.0347	0.9376	0.0107	0.0311	0.9373	0.0107	0.0312	**0.9511**
Energy	50 min	2.6155	0.0451	0.9178	2.2813	0.0393	0.9190	2.4529	0.0423	0.9212
	100 min	3.2763	0.0565	0.8574	3.0951	0.0534	0.8640	3.4381	0.0593	0.8603
	150 min	3.8457	0.0663	0.8106	3.4979	0.0603	0.8216	3.7557	0.0648	0.8121
Nasdaq	5 min	0.2218	0.0014	0.9945	0.1708	0.0011	0.9940	0.1301	0.0009	0.9965
	10 min	0.2446	0.0017	0.9915	0.2511	0.0016	0.9902	0.2054	0.0013	0.9931
	15 min	0.3177	0.0027	0.9857	0.2603	0.0017	0.9872	0.2375	0.0016	0.9898

Dataset	Horizon	Method					
		CauGNN-nCau			CauGNN-nCNN		
		MAE	RAE	CORR	MAE	RAE	CORR
Exchange-rate	5 days	0.0076	0.0221	0.966	0.0074	0.0240	0.9634
	10 days	0.0093	0.0290	0.9531	0.0096	0.0350	0.9518
	15 days	0.0113	0.0315	0.9425	0.0118	0.0325	0.9398
Energy	50 min	2.1753	0.0369	0.9210	2.2346	0.0575	0.9196
	100 min	2.8731	0.0475	0.8587	2.7488	0.0574	0.8608
	150 min	3.4122	0.0588	0.8167	3.5229	0.0673	0.8121
Nasdaq	5 min	0.1601	0.0010	0.9942	0.1884	0.0012	0.9937
	10 min	0.2174	0.0014	0.9907	0.4454	0.0029	0.9909
	15 min	0.2490	0.0016	0.9879	0.3342	0.0022	0.9856

Dataset	Horizon	Method					
		CauGNN			CauGIN		
		MAE	RAE	CORR	MAE	RAE	CORR
Exchange-rate	5 days	**0.0060**	**0.0176**	**0.9694**	0.0065	0.0188	0.9690
	10 days	**0.0083**	**0.0243**	0.9548	0.0089	0.0259	0.9551
	15 days	**0.0104**	**0.0302**	0.9438	0.0108	0.0315	0.9441
Energy	50 min	**2.0454**	**0.0358**	**0.9267**	2.1768	0.0375	0.9204
	100 min	**2.7242**	**0.0470**	**0.8673**	2.8097	0.0485	0.8615
	150 min	**3.3232**	**0.0573**	**0.8221**	3.3572	0.0579	0.8131
Nasdaq	5 min	0.1549	0.0010	0.9951	**0.1174**	**0.0008**	**0.9968**
	10 min	0.1897	0.0012	0.9922	**0.1664**	**0.0011**	**0.9937**
	15 min	0.2358	0.0015	0.9887	**0.2043**	**0.0013**	**0.9907**

comparable results by modeling periodic dependency patterns. The proposed CauGNN uses the TE matrix to collect the internal relationship between variables and analyze the topology composed of variables and relationships through the graph network. Thus, it can break through these restrictions and perform well on general datasets.

We also fine-tune other deep learning baseline models and choose suitable hyperparameters to achieve their best performance. For the MulitiHead Attention, LSTNet, and MLCNN, we set hidCNN, hidRNN, hidSkip, and window size as 50, 50, 5, and 128, respectively. For RNN-GRU, we set hidRNN as 50 and highway window as 24. As for our model CauGNN, we set hidCNN, hidGNN1, hidGNN2, and window size as 12, 30, 10, and 32, respectively. Compared with these baseline models, our proposed CauGNN model can share the same hyperparameters among various datasets and situations with robust performance, as revealed from the results.

4.6 Variant comparison

Our proposed framework has strong universality and compatibility. We replace the k-GNNs layer with the GIN layer, which also well preserves the distinctness of inputs. As shown in Table 1, the GIN layer fits into our model well and CauGIN has a similar performance with CauGNN.

For the ablation study, we also replace the TE matrix with an all-one matrix in CauGNN-nCau, assuming the value to be predicted of a single variable is related to all other variables. Thus, a completed graph is fed into the GNN layers. The experiment results show that CauGNN outperforms CauGNN-nCau, indicating the significant role that the TE matrix plays in the CauGNN model. On the other hand, we conduct experiments using the CauGNN-nCNN model, in which the CNN component is removed. The input time series data without feature extraction are fed into the GNN layer instead of the node features extracted from the CNN layer. Experiment results show that CauGNN outperforms CauGNN-nCNN, suggesting the significant role that the CNN component plays in the CauGNN model.

To test the parameter sensitivity of our model, we evaluate how the hidden size of the GNN component can affect the results. We report the MAE, RAE, and CORR metrics on the exchange-rate dataset. As seen in Fig. 2 , while ranging the hidden size of GNN layers from $\{10, 20, \ldots, 100\}$, the model performance is steady, being relatively insensitive to the hidden dimension parameter.

To prove the superiority of multiple CNN filters, we also did an ablation study on the exchange-rate dataset when the horizon is 5 days. As shown in Fig. 3, CauGNN-RF, CauGNN-1CNN, and CauGNN represent the direct use of the raw feature (original data) as the input of the node embedding model, using one CNN filter (here, we set kernel size to 3), and the complete model CauGNN is constructed with three CNN filters. We can find that CauGNN-RF has the worst performance, indicating that the direct use of raw features will introduce too much noise, which is not conducive to the subsequent learning of the model. Meanwhile, CauGNN has the best performance, indicating that stacking multiple CNN filters can better capture multiple inherent time series period characteristics and make more accurate predictions.

5 Conclusion

In this paper, we propose a novel deep learning framework (namely CauGNN) for MTS forecasting. Using CNN with multiple receiving fields, our model introduces a priori causal information with TE features and uses a GNN for feature extraction, which effectively improves the results in the MTS forecasting. With in-depth theoretical analysis and experimental verification, we confirm that CauGNN successfully captures the causal relationship among variables and selects key variables for accurate forecasting using a GNN.

Acknowledgment

This work was supported in part by the National

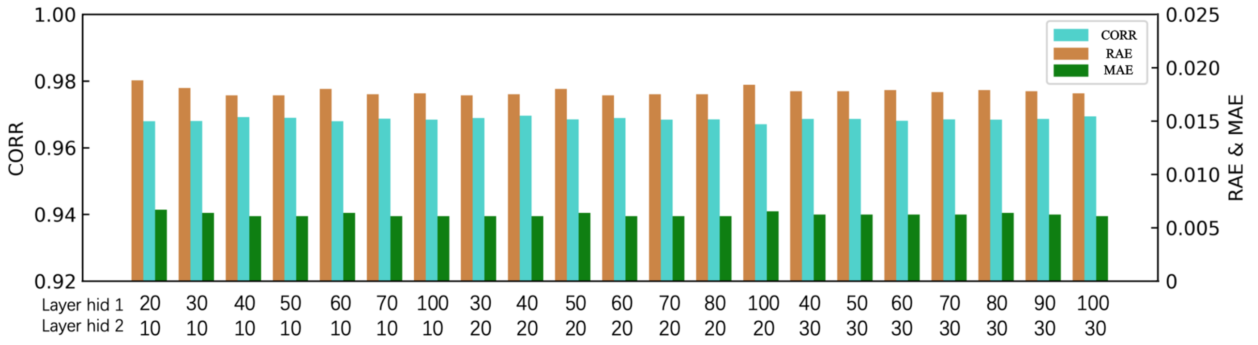

Fig. 2 Parameter sensitivity test results. Here "Layer hid 1" and "Layer hid 2" represent the embedding size of the first GNN layer and the second GNN layer, respectively. CauGNN shows steady performance under different settings of hidden sizes in the GNN layer.

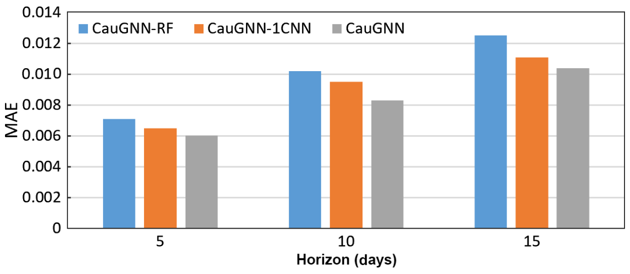

Fig. 3 Ablation study without multiple CNN filters on the exchange-rate dataset when the horizon is 5 days.

Natural Science Foundation of China (No. 62002035) and the Natural Science Foundation of Chongqing (No. cstc2020jcyj-bshX0034).

References

[1] Z. H. Duan, H. Y. Xu, Y. Y. Wang, Y. D. Huang, A. N. Ren, Z. B. Xu, Y. Z. Sun, and W. Wang, Multivariate time series classification with hierarchical variational graph pooling, arXiv preprint arXiv: 2010.05649, 2020.

[2] H. Chen, S. Feng, X. Pei, Z. Zhang, and D. Yao, Dangerous driving behavior recognition and prevention using an autoregressive time-series model, *Tsinghua Science and Technology*, vol. 22, no. 6, pp. 682–690, 2017.

[3] Y. S. Liu, C. H. Yang, K. K. Huang, and W. H. Gui, Nonferrous metals price forecasting based on variational mode decomposition and LSTM network, *Knowl. Based Syst.*, vol. 188, p. 105006, 2020.

[4] L. Yu, S. Dong, M. Lu, and J. Wang, LSTM based reserve prediction for bank outlets, *Tsinghua Science and Technology*, 2018, vol. 24, no. 1, pp. 77–85, 2018.

[5] Y. Y. Wang, Z. H. Duan, Y. D. Huang, H. Y. Xu, J. Feng, and A. N. Ren, MTHetGNN: A heterogeneous graph embedding framework for multivariate time series forecasting, *Pattern Recognit. Lett.*, vol. 153, pp. 151–158, 2022.

[6] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*, 4th Edition, Hoboken, NJ, USA: John Wiley & Sons, 2016.

[7] J. D. Hamilton, *Time Series Analysis*. Princeton, NJ, USA: Princeton University Press, 1994.

[8] H. Lütkepohl, *New Introduction to Multiple Time Series Analysis*. Berlin, Germany: Springer, 2005.

[9] A. Tokgöz and G. Ünal, A RNN based time series approach for forecasting Turkish electricity load, in *Proc. 2018 26th Signal Processing and Communications Applications Conf.* (*SIU*), Izmir, Turkey, 2018, pp. 1–4.

[10] J. L. Elman, Finding structure in time, *Cogn. Sci.*, vol. 14, no. 2, pp. 179–211, 1990.

[11] S. Hochreiter and J. Schmidhuber, Long short-term memory, *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling, arXiv preprint arXiv: 1412.3555, 2014.

[13] G. K. Lai, W. C. Chang, Y. M. Yang, and H. X. Liu, Modeling long- and short-term temporal patterns with deep neural networks, in *Proc. the 41st Int. ACM SIGIR Conf. on Research & Development in Information Retrieval*, Ann Arbor, MI, USA, 2017, pp. 95–104.

[14] Y. F. Zhou, Z. H. Duan, H. Y. Xu, J. Feng, A. N. Ren, Y. Y. Wang, and X. Q. Wang, Parallel extraction of long-term trends and short-term fluctuation framework for multivariate time series forecasting, arXiv preprint arXiv: 2008.07730, 2020.

[15] C. W. J. Granger, Investigating causal relations by econometric models and cross-spectral methods, *Econometrica*, vol. 37, no. 3, pp. 424–438, 1969.

[16] G. Kirchgässner, J. Wolters, and U. Hassler, Granger causality, in *Introduction to Modern Time Series Analysis*, G. Kirchgässner, J. Wolters, and U. Hassler, eds. Berlin, Germany: Springer, 2013, pp. 95–125.

[17] T. Bossomaier, L. Barnett, M. Harré, and J. T. Lizier, Transfer entropy, in *An Introduction to Transfer Entropy*, T. Bossomaier, L. Barnett, M. Harré, and J. T. Lizier, eds. Cham, Switzerland: Springer, 2016, pp. 65–95.

[18] T. Dimpfl and F. J. Peter, Using transfer entropy to measure information flows between financial markets, *Stud. Nonlinear Dyn. Econom.*, vol. 17, no. 1, pp. 85–102, 2013.

[19] T. Q. Tung, T. Ryu, K. H. Lee, and D. Lee, Inferring gene regulatory networks from microarray time series data using transfer entropy, in *Proc. 20th IEEE Int. Symp. on Computer-Based Medical Systems* (*CBMS'07*), Maribor, Slovenia, 2007, pp. 383–388.

[20] M. Bauer, J. W. Cox, M. H. Caveness, J. J. Downs, and N. F. Thornhill, Finding the direction of disturbance propagation in a chemical process using transfer entropy, *IEEE Trans. Control Syst. Technol.*, vol. 15, no. 1, pp. 12–21, 2007.

[21] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, The graph neural network model, *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, 2009.

[22] Y. Y Wang, Z. H. Duan, B. B. Liao, F. Wu, and Y. T. Zhuang, Heterogeneous attributed network embedding with graph convolutional networks, in *Proc. the AAAI Conf. on Artificial Intelligence*, Palo Alto, CA, USA, 2019, pp. 10061&10062.

[23] Z. H. Duan, Y. Y. Wang, W. H. Ye, Z. X. Feng, Q. L. Fan, and X. H. Li, Connecting latent relationships over heterogeneous attributed network for recommendation, arXiv preprint arXiv: 2103.05749, 2021.

[24] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, Neural message passing for quantum chemistry, in *Proc. the 34th Int. Conf. on Machine Learning*, Sydney, Australia, 2017, pp. 1263–1272.

[25] Z. T. Ying, J. X. You, C. Morris, X. Ren, W. Hamilton, and J. Leskovec, Hierarchical graph representation learning with differentiable pooling, in *Proc. Advances in Neural Information Processing Systems*, Montréal, Canada, 2018, pp. 4800–4810.

[26] H. Y. Xu, R. J. Chen, Y. S. Bai, Z. H. Duan, J. Feng, Y. Z. Sun, and W. Wang, CoSimGNN: Towards large-scale graph similarity computation, arXiv preprint arXiv: 2005.07115, 2020.

[27] H. Y. Xu, Z. H. Duan, Y. Y. Wang, J. Feng, R. J. Chen, Q. R. Zhang, and Z. B. Xu, Graph partitioning and graph neural network based hierarchical graph matching for graph similarity computation, *Neurocomputing*, vol. 439, pp. 348–362, 2021.

[28] T. N. Kipf and M. Welling, Semi-supervised classification with graph convolutional networks, arXiv preprint arXiv: 1609.02907, 2016.

[29] W. L. Hamilton, R. Ying, and J. Leskovec, Inductive representation learning on large graphs, in *Proc. the 31st Int. Conf. on Neural Information Processing Systems*, Long Beach, CA, USA, 2017, pp. 1025–1035.

[30] J. Chen, T. F. Ma, and C. Xiao, FastGCN: Fast learning with graph convolutional networks via importance sampling, arXiv preprint arXiv: 1801.10247, 2018.

[31] K. Y. L. Xu, W. H. Hu, J. Leskovec, and S. Jegelka, How powerful are graph neural networks? arXiv preprint arXiv: 1810.00826, 2018.

[32] C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan, and M. Grohe, Weisfeiler and leman go neural: Higher-order graph neural networks, in *Proc. the AAAI Conf. on Artificial Intelligence*, Palo Alto, CA, USA, 2019, pp. 4602–4609.

[33] P. Velikovi, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, Graph attention networks, arXiv preprint arXiv: 1710.10903, 2017.

[34] S. Han, H. B. Dong, X. Y. Teng, X. H. Li, and X. W. Wang, Correlational graph attention-based long short-term memory network for multivariate time series prediction, *Appl. Soft Comput.*, vol. 106, p. 107377, 2021.

[35] T. Schreiber, Measuring information transfer, *Phys. Rev. Lett.*, vol. 85, no. 2, pp. 461–464, 2000.

[36] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv: 1412.6980, 2014.

[37] L. M. Candanedo, V. Feldheim, and D. Deramaix, Data driven prediction models of energy use of appliances in a low-energy house, *Energy Build.*, vol. 140, pp. 81–97, 2017.

[38] Y. Qin, D. J. Song, H. F. Cheng, W. Cheng, G. F. Jiang, and G. W. Cottrell, A dual-stage attention-based recurrent neural network for time series prediction, in *Proc. the 26th Int. Joint Conf. on Artificial Intelligence*, Melbourne, Australia, 2017, pp. 2627–2633.

[39] Y. LeCun and Y. Bengio, Convolutional networks for images, speech, and time series, in *The Handbook of Brain Theory and Neural Networks*, M. A. Arbib, ed. Cambridge, MA, USA: MIT Press, 1998, pp. 255–258.

[40] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, Attention is all you need, in *Proc. the 31st Int. Conf. on Neural Information Processing Systems*, Long Beach, CA, USA, 2017, pp. 6000–6010.

[41] J. Z. Cheng, K. Z. Huang, and Z. B. Zheng, Towards better forecasting by fusing near and distant future visions, in *Proc. the AAAI Conf. on Artificial Intelligence*, Palo Alto, CA, USA, 2020, pp. 3593–3600.

Ziheng Duan received the BEng degree from Zhejiang University, China in 2020. Now he is a research intern at Chongqing University. His research interests lie in the area of machine learning, computational biology, graph representation learning, time series analysis, espeically the interaction of them.

Haoyan Xu is an undergraduate student at the College of Control Science and Engineering, Zhejiang University. His research interests lie in the area of graph representation learning, time series analysis, robot learning, and microfluidics. He is particular interested in graph neural networks, with their applications in language processing, graph mining, etc.

Yida Huang is an undergraduate student at College of Computer Science and Engineering, Zhejiang University. His research interests lie in the area of time series analysis, graph representation learning, natural language processing, and their applications in cyber security.

Jie Feng is an undergraducate student at the College of Control Science and Engineering, Zhejiang University. His research interests include artificial intelligence and robotics.

Yueyang Wang received the BEng degree from Nanjing University, Nanjing, China in 2013, and the PhD degree from Zhejiang University, Hangzhou, China in 2019. She is currently a lecturer at the School of Big Data and Software Engineering, Chongqing University. Her research interests include social network analysis and data mining.