Multivariate Time Series Forecasting with Transfer Entropy Graph

Haoyan Xu*
College of Control Science and Engineering
Zhejiang University
haoyanxu@zju.edu.cn

Yida Huang*
College of Computer Science and Engineering
Zhejiang University
stevenhuang@zju.edu.cn

Ziheng Duan*
College of Control Science and Engineering
Zhejiang University
duanziheng@zju.edu.cn

Xiaoqian Wang†
School of Electrical and Computer Engineering
Purdue University
joy.xqwang@gmail.com

Jie Feng
College of Control Science and Engineering
Zhejiang University
zjucse_fj@zju.edu.cn

Pengyu Song†
College of Electrical Engineering
Zhejiang University
3160104168@zju.edu.cn

Abstract

Multivariate time series (MTS) forecasting is an important problem in many fields. Accurate forecasting results can effectively help decision-making. 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, which ignores the causal relationship among variables. To address the above issue, a novel end-to-end deep learning model, termed graph neural network with transfer entropy (TEGNN) is proposed in this paper. To characterize the causal information among variables, the Transfer Entropy (TE) graph is introduced in our model, where each variable is regarded as a graph node and each edge represents the casual relationship between variables. In addition, convolutional neural network (CNN) filters with different perception scales are used for time series feature extraction, which is used to generate the feature of each node. Finally, graph neural network (GNN) is adopted to tackle the forecasting problem of graph structure generated by MTS. Three benchmark datasets from the real world are used to evaluate the proposed TEGNN and the comprehensive experiments show that the proposed method achieves state-of-the-art results in MTS forecasting task.

1 Introduction

In the real world, multivariate time series (MTS) data are common in various fields, such as the sensor data in the Internet of things, the traffic flows on highways, and the prices collected from stock markets. Through the existing MTS data, prediction models can be established to estimate the future trend. In this case, MTS forecasting becomes an important problem in many fields. For example, prediction the stock prices to determine the investment strategy, and prediction the traffic flows to reasonably plan the travel route.

*Contributed equally.
†Correspondence to: Xiaoqian Wang<joy.xqwang@gmail.com>,Pengyu Song<3160104168@zju.edu.cn>

Preprint. Under review.

In recent years, many time series forecasting methods have been widely studied and applied. For univariate situations, autoregressive integrated moving average model (ARIMA) [3] is one of the most classic forecasting methods. This method includes a variety of time series models, including autoregression (AR), moving average (MA), and autoregressive moving average (ARMA), thus has the flexibility and adaptability to various types of time series. However, due to the high computational complexity, ARIMA is not suitable for multivariate situations. VAR [13, 21, 3] method is a multivariate extended version of the AR model. Although VAR is widely used in MTS forecasting tasks due to its simplicity, it cannot handle the nonlinear relationships among variables, which reduce its forecasting accuracy.

In addition to traditional statistical methods, deep learning methods are also applied for the MTS forecasting problem [26]. Due to the flexibility of the neural network structures, deep learning methods can well capture the dynamics and changing trends of the time series by taking temporal sequence into account. The recurrent neural network (RNN) [9] and its two improved versions, namely the long short term memory (LSTM) [15] and the gated recurrent unit (GRU) [7], realize the extraction of time series dynamic information through the memory mechanism. Convolutional neural network (CNN) [20] uses multiple convolution kernels to perform moving convolution operations in the time series, thereby achieving feature extraction according to time order. Besides, the Multi-Head attention mechanism (MHA)[28] which concatenates and projects the input into query, key and value space in the famous Transformer model could also be used in encoding MTS sequence. By specific combination of the above neural network structures can achieve reasonable MTS forecasting results.

Nevertheless, the existing deep learning methods assume that the predicted value of a single variable is affected by all other variables. In fact, for a time series to be predicted, its future value may be only related to a few other variables in the dataset. For example, the future traffic flow of a certain street is easier to be predicted by the traffic information of the neighboring area, while the information of the area farther away is relatively useless. If such priori causal information can be considered, then it is more conducive to the interaction among variables with causality which can be an effective guideline for forecasting. There have been studies on the quantitative characterization of time series causality. Among them, the most famous is Granger causality analysis (G-causality) [12, 18]. This method represents the causality by establishing an AR model and comparing the prediction residuals when selecting different independent variables. However, as a linear model, G-causality cannot well handle nonlinear relationships. In this case, transfer entropy (TE) [2, 10] is also proposed for causal analysis, which is able to deal with the nonlinear relationships. Since TE was proposed, it has been widely used for data analysis in the economic [8], biological [27] and industrial [1] fields.

In this work, a novel framework, termed Graph Neural Network with Transfer Entropy (TEGNN) is proposed and applied for MTS forecasting tasks, which considers the causal relationships among variables. For the introduction of causality, The pairwise TE between variables is calculated, thus obtain the TE matrix, which is regarded as the adjacency matrix of the graph structure and each variable is one node of this graph. In addition, convolutional neural network (CNN) filters with different perception scales are used for time series feature extraction, which is used to generate the feature of each node. What is more, graph neural network is adopted to tackle the embedding and forecasting problem of graph structure generated by MTS. Our major contributions are:

- To the best of our knowledge, we are the first to propose the end-to-end deep learning framework that considers multivariate time series as a graph structure with causality, so that the causality among time series is used as priori information to guide the forecasting task, and graph neural network is utilized to process this graph structure.
- We use transfer entropy to extract the causality among time series and construct the TE graph. A CNN structure with multiple receptive fields is used to comprehensively extract the features of time series, which are used as node features in the TE graph.
- We conduct extensive experiments on MTS benchmark datasets and the results from the experiment have proved that TEGNN out-performs the state-of-the-art models.

The rest of this paper is organized as follows. Section 2 outlines the related preliminary information in detail, including TE and GNN methods. Section 3 describe the proposed TEGNN model. Section 4 reports the evaluation results of the proposed model in comparison with baselines on real-world datasets. Finally, in Section 5, the paper is concluded along with a discussion on the future research.

2 PRELIMINARIES

2.1 Transfer Entropy

Transfer entropy (TE) is a measure of causality based on information theory, which was proposed by Schreiber in 2000 [25]. Before introducing TE, two concepts in information theory should be presented in advance. Given a variable X, its information entropy is defined as:

$$H(X) = -\sum p(x) \log_2 p(x), \tag{1}$$

where x denotes all possible values of variable X. Information entropy is used to measure the amount of information. A larger $H(X)$ indicates that the variable X contains more information. Conditional entropy is another information theory concept. Given two variables X and Y, it is defined as:

$$H(X|Y) = -\sum \sum p(x, y) \log_2 p(x|y), \tag{2}$$

where conditional entropy $H(X|Y)$ represents the information amount of X under the condition that the variable Y is known.

The TE of variables Y to X is defined as:

$$\begin{aligned} T_{Y \to X} &= \sum p\left(x_{t+1}, x_t^{(k)}, y_t^{(l)}\right) \log_2 p\left(x_{t+1}|x_t^{(k)}, y_t^{(l)}\right) - \sum p\left(x_{t+1}, x_t^{(k)}\right) \log_2 p\left(x_{t+1}|x_t^{(k)}\right) \\ &= \sum p\left(x_{t+1}, x_t^{(k)}, y_t^{(l)}\right) \log_2 \frac{p\left(x_{t+1}|x_t^{(k)}, y_t^{(l)}\right)}{p\left(x_{t+1}|x_t^{(k)}\right)} \\ &= H\left(X_{t+1}|X_t\right) - H\left(X_{t+1}|X_t, Y_t\right) \end{aligned} \tag{3}$$

where x_t and y_t represent their values at time t. $x_t^{(k)} = [x_t, x_{t-1}, ..., x_{t-k+1}]$ and $y_t^{(l)} = [y_t, y_{t-1}, ..., y_{t-l+1}]$. It can be found that TE is actually an increase in the information amount of the variable X when Y changes from 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 X and Y is usually further indicated in the following way:

$$T_{X,Y} = T_{X \to Y} - T_{Y \to X} \tag{4}$$

When $T_{X,Y}$ is greater than 0, it means that X is the cause of Y, otherwise X is the consequence of Y.

2.2 Graph Neural Network

The concept of graph neural network (GNN) was first proposed in [24], which extended existing neural networks for processing the data represented in graph domains. A wide variety of graph neural network (GNN) models have been proposed in recent years. Most of these approaches fit within the framework of "neural message passing" proposed by Gilmeret et al.[11]. 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[31].

A separate line of work focuses on generalizing convolutions to graphs. The Graph Convolutional Networks(GCN)[17] could be regarded as an approximation of spectral-domain convolution of the graph signals. GCN convolutional operation could also be viewed as sampling and aggregating of the neighborhood information, such as GraphSAGE [14] and FastGCN [5], enabling training in batches while sacrificing some time-efficiency. Coming right after GCN, Graph Isomorphism Network(GIN) [30] and k-GNNs[22] is developed, enabling more complex forms of aggregation. Graph Attention Networks (GAT) [29] is another nontrivial direction to go under the topic of graph neural networks. It incorporates attention into propagation, attending over the neighbors via self-attention.

3 Methodology

This section introduces the proposed TEGNN in detail, which is a graph neural network based approach that takes the causal relationship among variables into account for MTS forecasting. A schematic of TEGNN is illustrated in Figure 1. The details of TEGNN is presented as below.

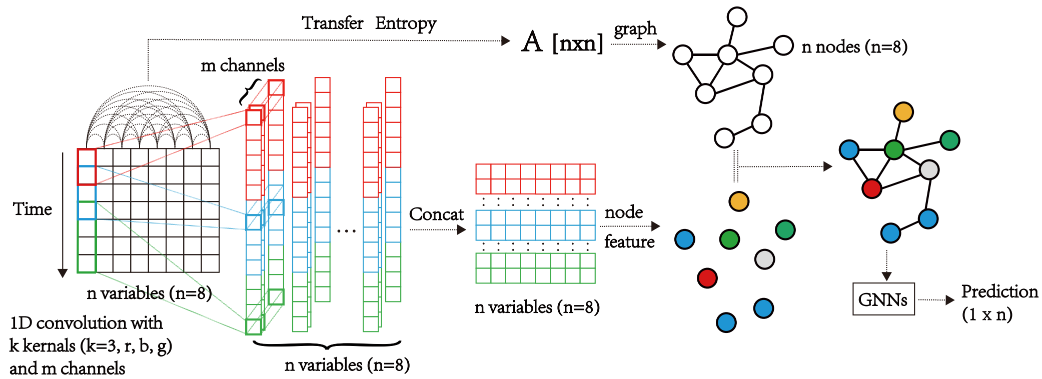

Figure 1: The schematic of TEGNN. A multivariate time series consists of multiple univariate time series. TEGNN maps a multivariate time series to a graph and each univariate time series (variable) is mapped to a node. Transfer entropy matrix is calculated to model the adjacency information of nodes, while convolutional layer is used to catch node features. The node feature matrix and adjacency matrix are then fed into graph neural network to get forecasts.

3.1 Problem Formulation

In this paper, the task of MTS forecasting is focused. Given a matrix consisting of multiple observed time series $X = [x_1, x_2, ..., x_t]$, where $x_i \in R^n (i = 1, ..., n)$ and n is the number of variables, the purpose of MTS forecasting is to predict x_{t+h}, where h is the horizon ahead of the current time stamp, which is usually determined according to the actual application scenario.

3.2 Causality Graph Structure Based on Transfer Entropy

When predicting the future value of a variable x, if we can directly determine which other variables have an effect on predicting x, it will be helpful to reduce the difficulty of model training and prevent incorrect timing relationships from being learned. As mentioned above, transfer entropy can characterize the causal relationship among variables. If the paired transfer entropy between variables is calculated before the prediction model is trained, and input into the model as a priori information, the selection of key variables can be achieved.

According to equations 3-4 in Section 2.1, the transfer entropy matrix T of the multivariate time series X can be obtained, where the element of the i-th row and j-th column of T, denoted t_{ij}, is calculated as:

$$t_{ij} = \begin{cases} T_{x_i,x_j}, & T_{x_i,x_j} > c \\ 0, & otherwise, \end{cases} \tag{5}$$

where x_i is the i-th variable of X, c is the threshold to determine whether the causality is significant. T can be regarded as the adjacency matrix of a graph structure, which is used for subsequent variable selection.

3.3 Time Series 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 this paper, multiple CNN filters with different receptive fields are used to extract individual features for each input time series. Time series from the real world often have multiple meaningful periods. For example, the traffic flow of a certain street not only shows a similar trend every day, but meaningful rules can also 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. However, before determining the network structure of the model, the effective period is often unknown. 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 x and p CNN filters, denoted as W_i, with different convolution kernel sizes $(1 \times k_i)(i = 1, 2...p)$ are separately generated and the features h are extracted as follows: $h_i = ReLU(W_i * x + b_i)$, $h = [h_1, h_2, ..., h_p]$. $*$ denotes the convolution operation, $[\cdot]$ represents the concatenate operation, and $ReLU$ is a nonlinear activation function $ReLU(x) = max(0, x)$. In this way, features under different periods are extracted, which provides effective information for time series prediction. It is worth noting that the feature extraction of each time series is separated from each other here, because the subsequent steps need to merge the information of different time series according to the transfer entropy matrix T.

3.4 Node Embedding Based on Transfer Entropy Matrix

After feature extraction, the input MTS is converted into a feature matrix $H \in \mathbb{R}^{n \times d}$, where d is the number of features after the calculation introduced in Section 3.3. 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 transfer entropy matrix T. For such graph structure, graph neural networks can be directly applied for the embedding of nodes. Inspired by k-GNNs[22] model, we propose TEGNN model and use the following propagation model for calculating the forward-pass update of a node denoted by v_i:

$$h_i^{(l+1)} = \sigma\Big(h_i^{(l)} W_1^{(l)} + \sum_{j \in N(i)} h_j^{(l)} W_2^{(l)}\Big), \tag{6}$$

where $W_1^{(l)}$ and $W_2^{(l)}$ are parameter matrices, $h_i^{(l)}$ is the hidden state of node v_i in the l^{th} layer and $N(i)$ denotes the neighbors of node i. k-GNNs only perform information fusion between a certain node and its neighbors, ignoring the information of other non-neighbor nodes. In this way, for the prediction of a time series, only other series with significant causality are considered. This design plays a role in highlighting the relationship among variables, which can effectively avoid the information redundancy brought by high dimensions. By adding the priori causal information obtained by TE, the model does not need to find out the key variables for forecasting by itself. In this paper, the output dimension of the last graph neural network layer is 1, which is used as the prediction result. We also conduct experiments using GIN[30] model and our corresponding model is called TEGIN. GIN can efficiently gather information of neighboring nodes, and learn accurate structural information through summation aggregation:

$$h_v^{(k)} = MLP^{(k)}\big((1 + \epsilon^{(k)}) \cdot h_v^{(k-1)} + \sum_{u \in N(v)} h_u^{(k-1)}\big), \tag{7}$$

where $h_v^{(k)}$ is the k-th layer node embedding for the node v, ϵ is a trainable parameter, MLP represents the nonlinear mapping composed of multi-layer fully connected neural networks and $N(v)$ represents the neighbor nodes of node v.

3.5 Objective Function

In the task of MTS forecasting, the following absolute loss (L_1-loss) function is often used:

$$\min_{\Theta} \sum_{t \in \Omega_{train}} \sum_{i=1}^{n} \left\| \hat{Y}_{t+h} - Y_{t+h} \right\|_1, \tag{8}$$

where $\hat{Y}_{t+h}$ is the prediction result of Y_{t+h} output by the model, n is the number of variables, Ω_{train} is the set of time stamps used for training and Θ denotes all trainable parameters in the model. This optimization function is also used in this paper and the optimization problem can be solved by stochastic gradient decent (SGD) or its improved versions such as Adam[16].

In this way, our model is completed: constructing the TE graph, extracting node features by CNN, and using graph neural networks to process the graph structure obtained from multivariate time series. The experiments in Section 4 prove the effectiveness of our framework.

4 Experiments

In this section, we conduct extensive experiments on three benchmark datasets for multivariate time series forecasting tasks, and compare the results of proposed TEGNN model with other 6 baselines. All the data and experiment codes are available online[3].

[3] Our codes will be released as well upon the acceptance of this paper.

4.1 Data

We use three benchmark datasets which are publicly available.

Exchange-Rate[4]: the exchange rates of eight foreign countries collected from 1990 to 2016, collected per day.

Energy[4]: measurements of 26 different quantities related to appliances energy consumption in a single house for 4.5 months, collected per 10 minutes.

Nasdaq[23]: the stock prices are selected as the multivariable time series for 82 corporations, collected per minute.

4.2 Methods for Comparison

The methods in our comparative evaluation are as follows:

- **VAR**[13, 21, 3] stands for the well-known vector regression model, which has proven to be a useful machine learning method for multivariate time series forecasting.
- **CNN-AR**[20] stands for classical convolution neural network. We use multi-layer CNN with AR components to perform MTS forecasting tasks.
- **RNN-GRU**[7] is the Recurrent Neural Network using GRU cell with AR components.
- **MultiHead Attention**[28] stands for multihead attention components in the famous Transformer model, where multi-head mechanism runs through the scaled dot-product attention multiple times in parallel.
- **LSTNet**[19] is a famous MTS forecasting framework which shows great performance by modeling long- and short-term temporal patterns of MTS data.
- **MLCNN**[6] is a novel multi-task deep learning framework which adopts the idea of fusing foreacasting information of different future time.
- **TEGNN** stands for our proposed Graph Neural Network with Transfer Entropy. We apply multi-layer CNN and k-GNNs to perform MTS forecasting tasks.
- **TEGIN** stands for our proposed Graph Isomorphism Network with Transfer Entropy, where k-GNNs layers are replaced by GIN layers.
- **TEGNN-nTE** We remove the Transfer entropy matrix use all-one adjacency matrix instead.
- **TEGNN-nCNN** We remove the CNN component and use input time series data as node features.

4.3 Metrics

We apply three conventional evaluation metrics to evaluate the performance of different models for multivariate time series prediction: Mean Absolute Error (**MAE**), Relative Absolute Error (**RAE**), Empirical Correlation Coefficient (**CORR**):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|p_i - a_i|, \quad RAE = \frac{\sum_{i=1}^{n}|p_i-a_i|}{\sum_{i=1}^{n}|\bar{a}-a_i|}, \quad CORR = \frac{\sum_{i=1}^{n}(p_i-\bar{p})(a_i-\bar{a})}{\sqrt{\sum_{i=1}^{n}(p_i-\bar{p})^2}\sqrt{\sum_{i=1}^{n}(a_i-\bar{a})^2}} \quad (9)$$

a = actual target
p = predict target

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

[4]https://github.com/laiguokun/multivariate-time-series-data

4.4 Experiment Details

We conduct grid search on tunable hyper-parameters on each method over all datasets. Specifically, we set the same grid search range of input window size for each method from $\{2^0,2^1,...,2^9\}$ 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 Recurrent and Convolutional layer is chosen from $\{10, 20, ..., 100\}$. For LSTNet, the skip-length p is chosen from $\{0, 12, ..., 48\}$. For MLCNN, the hidden dimension of Recurrent and Convolutional layer is chosen from $\{10, 25, 50, 100\}$. We adopt dropout layer after each layer, and the dropout rate is set from $\{0.1, 0.2\}$. We calculate transfer entropy matrix based on train and validation data. For TEGNN, TEGIN, TEGNN-nTE, TEGNN-nCNN, we set the size of the three convolutional kernels to be $\{3, 5, 7\}$ respectively and the number of channels of each kernel is 12 in all our models. The hidden dimension of k-GNNs layer is chosen from $\{10, 20, ..., 100\}$. For TEGIN, the hidden size is chosen from $\{10, 20, ..., 100\}$. The Adam algorithm is used to optimize the parameters of our model.

4.5 Main Results

Table 1 summarizes the evaluation results of all the methods on 3 benchmark datasets with 3 metrics. Following the test settings of [19], we use each model for time series predicting on 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 metrics on each dataset is set bold in the Table 1.

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

Dataset		Exchange-Rate			Energy			Nasdaq		
Methods	Metrics	horizon 5	horizon 10	horizon 15	horizon 5	horizon 10	horizon 15	horizon 5	horizon 10	horizon 15
VAR	MAE	0.0065	0.0093	0.0116	3.1628	4.2154	5.1539	0.1706	0.2667	0.3909
	RAE	0.0188	0.0270	0.0339	0.0545	0.0727	0.0889	0.0011	0.0018	0.0026
	CORR	0.9619	0.9470	0.9318	0.9106	0.8482	0.7919	0.9911	0.9273	0.5528
CNN-AR	MAE	0.0063	0.0085	0.0106	2.4286	2.9499	3.5719	0.2110	0.2650	0.2663
	RAE	0.0182	0.0249	0.0303	0.0419	0.0509	0.0616	0.0014	0.0017	0.0017
	CORR	0.9638	0.9490	0.9372	0.9159	0.8618	0.8150	0.9920	0.9919	0.9860
RNN-GRU	MAE	0.0066	0.0092	0.0122	2.7306	3.0590	3.7150	0.2245	0.2313	0.2700
	RAE	0.0192	0.0268	0.0355	0.0471	0.0528	0.0641	0.0015	0.0015	0.0018
	CORR	0.9630	0.9491	0.9323	0.9167	0.8624	0.8106	0.9930	0.9901	0.9877
MultiHead Att	MAE	0.0078	0.0101	0.0119	2.6155	3.2763	3.8457	0.2218	0.2446	0.3177
	RAE	0.0227	0.0294	0.0347	0.0451	0.0565	0.0663	0.0014	0.0017	0.0027
	CORR	0.9630	0.9500	0.9376	0.9178	0.8574	0.8106	0.9945	0.9915	0.9857
LSTNet	MAE	0.0063	0.0085	0.0107	2.2813	3.0951	3.4979	0.1708	0.2511	0.2603
	RAE	0.0184	0.0247	0.0311	0.0393	0.0534	0.0603	0.0011	0.0016	0.0017
	CORR	0.9639	0.9490	0.9373	0.9190	0.8640	0.8216	0.9940	0.9902	0.9872
MLCNN	MAE	0.0065	0.0094	0.0107	2.4529	3.4381	3.7557	0.1301	0.2054	0.2375
	RAE	0.0189	0.0274	0.0312	0.0423	0.0593	0.0648	0.0009	0.0013	0.0016
	CORR	0.9693	**0.9559**	**0.9511**	0.9212	0.8603	0.8121	0.9965	0.9931	0.9898
TEGNN-nTE	MAE	0.0076	0.0093	0.0113	2.1753	2.8731	3.4122	0.1601	0.2174	0.2490
	RAE	0.0221	0.0290	0.0315	0.0369	0.0475	0.0588	0.0010	0.0014	0.0016
	CORR	0.9660	0.9531	0.9425	0.9210	0.8587	0.8167	0.9942	0.9907	0.9879
TEGNN-nCNN	MAE	0.0074	0.0096	0.0118	2.2346	2.7488	3.5229	0.1884	0.4454	0.3342
	RAE	0.0240	0.0350	0.0325	0.0575	0.0574	0.0673	0.0012	0.0029	0.0022
	CORR	0.9634	0.9518	0.9398	0.9196	0.8608	0.8121	0.9937	0.9909	0.9856
TEGNN	MAE	**0.0060**	**0.0083**	**0.0104**	**2.0454**	**2.7242**	**3.3232**	0.1549	0.1897	0.2358
	RAE	**0.0176**	**0.0243**	**0.0302**	**0.0358**	**0.0470**	**0.0573**	0.0010	0.0012	0.0015
	CORR	**0.9694**	0.9548	0.9438	**0.9267**	**0.8673**	**0.8221**	0.9951	0.9922	0.9887
TEGIN	MAE	0.0065	0.0089	0.0108	2.1768	2.8097	3.3572	**0.1174**	**0.1664**	**0.2043**
	RAE	0.0188	0.0259	0.0315	0.0375	0.0485	0.0579	**0.0008**	**0.0011**	**0.0013**
	CORR	0.9690	0.9551	0.9441	0.9204	0.8615	0.8131	**0.9968**	**0.9937**	**0.9907**

We save the model that has the best performance on validation set based on RAE or MAE metric after training 1000 epochs for each method. Then we use the model to test and record the results. The results shows the proposed TEGNN model outperforms most of the baselines in most cases, indicating the effectiveness of our proposed model on multivariate time series predicting tasks adopting the idea of using causality as guideline for forecasting. On the other side, we observe the result of VAR model on Nasdaq dataset is far worse than other methods in some cases, partly because VAR is not sensitive to the scale of input data which lower its performance.

MLCNN shows impressing results because it can fuse near and distant future visions, while LSTNet model shows impressing results when modeling periodic dependency patterns occurred in data. Our proposed TEGNN uses transfer entropy matrix to collect the internal relationship between variables and analyze the topology composed of variables and relationships through graph network, thus it can break through these restrictions and perform well on general datasets.

Other deep learning baseline models show similar performance. This results from the fine-tuned work on general deep learning methods and the suitable hyper-parameters we used. We use the following sets of hyperparameters for RNN-GRU, MultiHead Attention, LSTNet and MLCNN: 50 (hidCNN), 50 (hidRNN), 5 (hidSkip), 128 (windowsize); RNN-GRU: 50 (hidRNN), 24 (highway window) on Exchange-Rate dataset, and fine-tuned adjustment over other datasets. TEGNN model sets 12 (hidCNN), 30 (hidGNN1), 10 (hidGNN2), 32 (window size) applying to all datasets and horizons. Compared with these baseline models, our proposed TEGNN model can share the same hyper-parameters among varies datasets and situations with robust performance as the results show.

4.6 Variant Comparison

Our proposed framework has strong universality and compatibility. We replace the k-GNNs layer with GIN layer, which also well preserves the distinctness of inputs. As showned in Table1, GIN layer fits into our model well and TEGIN has similar performance with TEGNN.
For ablation study, we also replace transfer entropy matrix with all-one matrix in TEGNN-nTE, assuming the value to be predicted of a single variable is related to all other variables, thus a completed graph is fed into GNN layers. The experiment results show that TEGNN outperforms TEGNN-nTE, which indicates the significant role TE matrix plays in TEGNN model. On the other hand, we conduct experiments by using TEGNN-nCNN model, in which CNN component is removed. The input time series data without feature extraction are fed into GNN layer instead of node features extracted from CNN layer. The experiment results show that TEGNN outperforms TEGNN-nCNN, which suggest the significant role CNN component plays in TEGNN model.

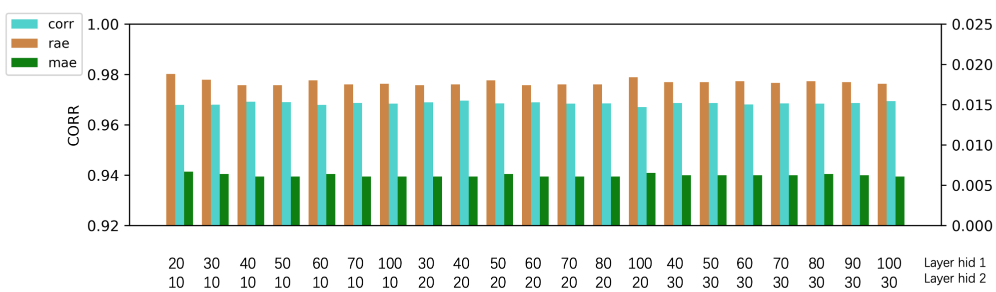

Figure 2: Parameter sensitivity test results. TEGNN shows steady performance under different settings of hidden sizes in GNN layer.

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**, **CORR** metrics on Exchange-Rate dataset. As can be seen in figure 2, while ranging the hidden size of GNN layers from $\{10, 20, ..., 100\}$, the model performance is steady, being relatively insensitive to the hidden dimension parameter.

5 Conclusion

In this paper, we propose a novel deep learning framework (TEGNN) for the task of multivariate time series forecasting. By using CNN with multiple receptive fields, introducing causal prior information characterized by transfer entropy, and adopting graph neural network for feature extraction, the proposed method effectively improved the state-of-the-art results in MTS forecasting on multiple datasets. With in-depth theoretical analysis and experimental verification, we confirm that TEGNN successfully captures the causal relationship among variables and uses graph neural network to select key variables for accurate forecasting.

In the future, there are several promising research directions that deserve more attention and efforts. Firstly, we use transfer entropy to represent causality. In fact, other causal calculation methods can also be tried to make more accurate selection of key variables. Secondly, other time series forecasting methods can be incorporated into the graph neural network to further improve prediction performance.

Broader Impact

Multivariate time series (MTS) forecasting is actually a regression prediction method, which belongs to quantitative prediction. Its basic principle is: it recognizes the continuity of the development of things, uses the past multivariate time series data for statistical analysis, and infers the development trend of things; on the other hand, it fully considers the randomness due to the impact of accidental factors. In order to eliminate the impact of random fluctuations, historical data is used for statistical analysis, and the data is appropriately processed for trend prediction.

MTS forecasting has a wide range of applications in real life. For example, in order to ensure the robustness of some key indicators in the company, they need to be detected in real time. Multivariate time series are formed when these key indicators have evolved over time. How to perform anomaly detection on these time series has become a key issue. If the anomaly is not found in time, it will affect the product and even bring a bad experience to the users. At the same time, in daily life, we can also use MTS forecasting to monitor human health: to summarize the health status from the changes in blood sugar and blood pressure in a day, which is undoubtedly beneficial to many elderly people living alone.

We are also fully aware that this technology may become a tool for some conspirators to induce others to do things. Driven by interests, people's privacy has been further exacerbated. Mastering the behavior data of a person's various indicators in the past, they can predict and even guide his future behavior. In the era of big data, how to better protect personal privacy, and how to prevent merchants from consuming traps tailored to others, this requires us to restrict the technology to some extent and cannot be abused by others.

References

[1] Margret Bauer, John W Cox, Michelle H Caveness, James J Downs, and Nina F Thornhill. Finding the direction of disturbance propagation in a chemical process using transfer entropy. *IEEE transactions on control systems technology*, 15(1):12–21, 2006.

[2] Terry Bossomaier, Lionel Barnett, Michael Harré, and Joseph T Lizier. Transfer entropy. In *An introduction to transfer entropy*, pages 65–95. Springer, 2016.

[3] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[4] Luis M Candanedo, Véronique Feldheim, and Dominique Deramaix. Data driven prediction models of energy use of appliances in a low-energy house. *Energy and buildings*, 140:81–97, 2017.

[5] Jie Chen, Tengfei Ma, and Cao Xiao. Fastgcn: fast learning with graph convolutional networks via importance sampling. *arXiv preprint arXiv:1801.10247*, 2018.

[6] Jiezhu Cheng, Kaizhu Huang, and Zibin Zheng. Towards better forecasting by fusing near and distant future visions. *arXiv preprint arXiv:1912.05122*, 2019.

[7] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[8] Thomas Dimpfl and Franziska Julia Peter. Using transfer entropy to measure information flows between financial markets. *Studies in Nonlinear Dynamics & Econometrics*, 17(1):85–102, 2013.

[9] Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

[10] Luca Faes, Giandomenico Nollo, and Alberto Porta. Compensated transfer entropy as a tool for reliably estimating information transfer in physiological time series. *Entropy*, 15(1):198–219, 2013.

[11] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1263–1272. JMLR. org, 2017.

[12] Clive WJ Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: journal of the Econometric Society*, pages 424–438, 1969.

[13] James D Hamilton. *Time series analysis*, volume 2. Princeton New Jersey, 1994.

[14] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. *CoRR*, abs/1706.02216, 2017.

[15] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

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

[17] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907, 2016.

[18] Gebhard Kirchgässner, Jürgen Wolters, and Uwe Hassler. Granger causality. In *Introduction to Modern Time Series Analysis*, pages 95–125. Springer, 2013.

[19] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long- and short-term temporal patterns with deep neural networks. *CoRR*, abs/1703.07015, 2017.

[20] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[21] Helmut Lütkepohl. *New introduction to multiple time series analysis*. Springer Science & Business Media, 2005.

[22] Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4602–4609, 2019.

[23] Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. *arXiv preprint arXiv:1704.02971*, 2017.

[24] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008.

[25] Thomas Schreiber. Measuring information transfer. *Physical review letters*, 85(2):461, 2000.

[26] Alper Tokgöz and Gözde Ünal. A rnn based time series approach for forecasting turkish electricity load. In *2018 26th Signal Processing and Communications Applications Conference (SIU)*, pages 1–4. IEEE, 2018.

[27] Thai Quang Tung, Taewoo Ryu, Kwang H Lee, and Doheon Lee. Inferring gene regulatory networks from microarray time series data using transfer entropy. In *Twentieth IEEE International Symposium on Computer-Based Medical Systems (CBMS'07)*, pages 383–388. IEEE, 2007.

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017.

[29] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[30] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

[31] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. In *Advances in neural information processing systems*, pages 4800–4810, 2018.