A novel weighted approach for time series forecasting based on visibility graph

Tianxiang Zhan\textsuperscript{a}, Fuyuan Xiao\textsuperscript{a,b,c,*}

\textsuperscript{a}School of Computer and Information Science, Southwest University, Chongqing, 400715, China
\textsuperscript{b}School of Big Data and Software Engineering, Chongqing University, Chongqing, 401331, China
\textsuperscript{c}National Engineering Laboratory for Integrated Aero-Space-Ground-Ocean Big Data Application Technology, China

Abstract

Time series has attracted a lot of attention in many fields today. Time series forecasting algorithm based on complex network analysis is a research hotspot. How to use time series information to achieve more accurate forecasting is a problem. To solve this problem, this paper proposes a weighted network forecasting method to improve the forecasting accuracy. Firstly, the time series will be transformed into a complex network, and the similarity between nodes will be found. Then, the similarity will be used as a weight to make weighted forecasting on the predicted values produced by different nodes. Compared with the previous method, the proposed method is more accurate. In order to verify the effect of the proposed method, the experimental part is tested on M1, M3 datasets and Construction Cost Index (CCI) dataset, which shows that the proposed method has more accurate forecasting performance.

Keywords: Time series, complex network, visibility graph, link forecasting, node similarity index

1. Introduction

Time series is a common data format, and there are many typical data, such as wind speed, stock price, seismic waves and so on. Time series forecasting
methods can predict a value at a future node in time. And time series forecasting can also provide reference value for economic evaluation [1, 2, 3, 4], computation optimizer [5, 6, 7], cost forecasting [8, 9] and so on [10, 11, 12].

Complex networks are the hotspot of recent research and have a wide range of influences. Challenging problems in complex network research include, but are not limited to, influence communicator identification [13, 14, 15, 16], fractal analysis [17, 18], decision making [19, 20], network evolution [21, 22, 23, 24], quantum theory [25, 26], network clustering [27] and so on [28, 29]. Information inherited from the network can help us better understand the properties of time series and make better forecasts. Therefore, a lot of research has focused on how to model time series in the form of networks. The visibility graph (VG) algorithm [30], proposed by Lacasa et al., converts time series into complex networks. Visibility graph have a wide range of applications such as image processing, decision-making and so on. In the study of time series forecasting, Zhang et al. proposed the forecasting method [31] using VG and the superposed random walk (SRW) algorithm [32]. Mao and Xiao improved the forecasting method of Zhang et al [33].

The previous method of Zhang et al. [31] and the method of Mao and Xiao [33] only considered the node most similar to the last node of the time series in the complex network when forecasting the time series. Because the structure of time series is variable, considering only one node for the final forecast will reduce the accuracy of the forecast. The previous research [31] uses the similarity of nodes as a reference, so the proposed method uses the similarity as a weight to weight the predicted values of different nodes, and obtains a more accurate forecasting effect than the previous method.

In this paper, there are some fundamental theories in Section 2. In Section 3, the paper provide the explanation of the proposed method. In Section 4, experimental results and analysis are introduced. In Section 5, it draws a conclusion of the paper.
2. Preliminaries

This section will introduce some preliminaries, including VG and link forecasting.

2.1. Visibility graph

A given time series $U$ is as follows:

$$U = \{(t_1, y_1), (t_2, y_2), ..., (t_m, y_m), ..., (t_n, y_n)\}$$  \hspace{1cm} (1)

where $y_m$ is the actual value of the data, and $t_m$ represents the node in time. In the visibility graph, an element $(t_m, y_m)$ is defined as a node.

In a complex network model, the relationship between nodes needs to be considered. So the nodes are analyzed in pairs. If two nodes $(t_a, y_a)$ and $(t_b, y_b)$ are connected with an edge, any node between them $(t_c, y_c)$ is required to satisfy [30]:

$$y_c < y_b + (y_a - y_b) \frac{t_b - t_c}{t_b - t_a}$$  \hspace{1cm} (2)

2.2. Superposed random walk

First, an undirected graph $G(V, E)$ is given. According to previous research [31], $G$ has a matrix $P$ recording transition probabilities and $P_{xy}$ is the probability that random walkers walking from node $x$ to $y$ which define as $P_{xy} = \frac{a_{xy}}{\sum_{k} a_{kx}}$.

Matrix $A$ is the adjacency matrix of $G$ [32].

The probability vector $\vec{\pi}_x(t)$ which records the probability of a walker walks to each node of $G$ in $t$ step from node $x$. When $t = 0$, $\vec{\pi}_x(0)$ is initialized to a vector in the shape of $N \times 1$ which $x$-th element is 1 and others equal to 0, $N = |V|$. And after $t$ steps, $\vec{\pi}_x(t)$ can calculate as follows:

$$\vec{\pi}_x(t) = P^T \vec{\pi}_x(t - 1)$$  \hspace{1cm} (3)

Then the local similarity $S_{xy}^{LRW}$ between node $x$ and $y$ is calculated as follows:

$$S_{xy}^{LRW}(t) = \frac{k_x}{2|E|} \times \vec{\pi}_{xy}(t) + \frac{k_y}{2|E|} \times \vec{\pi}_{yx}(t)$$  \hspace{1cm} (4)
Superposed similarity $S_{SRW}$ is superposed local similarity $S_{LRW}$ until local similarities no longer change which defines as follows:

$$S_{SRW}^{xy} = \sum_{l=1}^{t} S_{LRW}^{xy}(l) \quad (5)$$

3. The proposed method

3.1. Step 1: Transforming time series

Consider a time series $U = \{(t_1, y_1), (t_2, y_2), \ldots, (t_n, y_n)\}$. The first step is to transform the time series $U$ into a undirected graph $G(V, E)$ through the VG definition.

Following Fig.1 at the is a histogram of a time series $U_1$ which defines as following Eq.6, and edges with visible node pairs have been marked with red segments.

$$U_1 = \{(1, 10), (2, 90), (3, 30), (4, 50), (5, 20), (6, 40), (7, 60), (8, 50), (9, 30), (10, 40)\} \quad (6)$$

By calculating the visual relationship between each pair of nodes, the adjacency matrix $A_1$ which defined in Eq.7 of $G_1$ can be obtained. According to matrix $A_1$, the time series $U_1$ is transformed into $G_1$ as shown in Fig.2.

$$A_1 = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0
\end{bmatrix} \quad (7)$$
3.2. Step 2: Calculate node similarity

In $G$, there are many pairs of nodes, and each pair of nodes has similarity. According to previous studies [31], by finding the node $v'$ with the highest similarity to the last node $v$ of the time series $U$, the forecasting effect of linear fitting is positive.

This paper improves on previous studies [31, 33]. Fig.3 shows the use of node similarity in forecasting by Zhang et al.’s forecasting method [31]. It can be seen that only one node $t_7$ is used. However, as the number of nodes in the time series becomes more, a weighting method is considered for forecasting. Through the SRW algorithm, the similarity between the previous node and the last node $v$ forms a similarity vector $S = [s_1, s_2, ..., s_{N-1}]$, $s_i = S_{iN}^{SRW}$ represents superposed similarity between node $i$ and node $n$, $N$ is the number of nodes.
Figure 3: The Zhang et al.'s method [31] only uses the node $t_7$ which is the most similar to the last node $t_{10}$ for the final forecasting.

Figure 4: The proposed method retains all the similarity information $S$ of the last node $t_{10}$ (the color of nodes $t_1 - t_9$ shows all the similarity of the last node $t_{10}$, and the more red the more similar).

in the time series $U$. As shown in Fig.4, the proposed method retains all the similarity information and uses all the node information in the forecasting.

According to previous studies [31], each node $i$ has a corresponding single step predicted value $\hat{y}_i$. $\hat{y}_i$ is the predicted value by linear fitting node $i$ to node $v$, as shown in Fig.5. $\hat{y}_i$ is calculated in Eq.8. Also, $y_{v'}$ is the predicted value of the previous research [31].

$$\hat{y}_i = y_N + \frac{y_{N} - y_i}{t_{N} - t_{i}} \quad \text{(8)}$$

However, when the number of nodes in the time series increases, with the influence of randomness and uncertainty, the single most similar node will lead to the decline of the forecasting effect. The proposed method first presents a forecasting vector $F$. Eq.10 is the forecasting vector $F_1$ of time series $U_1$ and Fig.6 shows the process of calculating vector $F_1$.

**Definition 1.** Forecasting vector $F$ records the predicted value of all nodes except the
The way previous studies predicted

Figure 5: Schematic diagram of Zhang et al.'s method [31] and node $t_7$ is the node most similar to the last node $t_{10}$.

The process of generating the prediction vector $F$

Figure 6: The process that produces the forecasting vector $F_1$, the nodes' value on $x = 11$ is the value of the vector $F_1$. 
last one as Eq.9.

\[ F = \{ \hat{y}_1, \hat{y}_2, ..., \hat{y}_{N-1} \} \tag{9} \]

\[ F_1 = [43.33, 33.75, 41.42, 38.33, 44.00, 40.00, 33.33, 35.00, 50.00] \tag{10} \]

Given the validity of previous research, here the weight vector \( w \) will come from the vector \( S \).

**Definition 2.** In the proposed method, the weight vector \( w \) is the normalized similarity vector \( S \). For each element \( s_i \) in \( S \), it is standardized according to \( w_i = s_i = \frac{s_i}{\sum_{l=1}^{N-1} s_l} \), so that the two-norm of \( S \) is 1, which meets the condition of being a weight.

### 3.3. Step 3: Weighted forecasting vector

The previous study is equivalent to assigning all the weights to the nodes with the largest similarity, that is, all the weights in \( w \) are 0 except the position of \( \text{argmax}(S) \) is 1. However, there is often a situation that the number of nodes with the largest similarity is not unique, and in this case, multiple predicted values will be generated. Therefore, the proposed method is to take the similarity as the weight vector \( w \), and the weighted average of the forecasting vector \( F \) as the forecasting value \( \hat{y} \). The predicted value \( \hat{y} \) is the dot product of the weight vector \( w \) and the predicted vector \( F \), as shown in Eq.11.

\[ \hat{y} = w \cdot F \tag{11} \]

In the time series \( U_{t1} \), the weight vector of Zhang et al.’s method is as in Eq.12, and only position of node \( t_7 \) is 1. The proposed similarity vector \( S_{\text{Proposed}} \) and weight vector \( w_{\text{Proposed}} \) are shown in Eq.13, Eq.14. The difference with Zhang et al.’s method [31] is that the weight of the proposed method retains the similarity information of nodes \( t_1 - t_9 \).

\[ w_{\text{Zhang}} = [0, 0, 0, 0, 0, 0, 1, 0, 0] \tag{12} \]
Thus, the predicted values of the proposed method and Zhang et al.’s method are the same as shown in Eq.15, Eq.16. Of course, the time series \( U_1 \) is only an example, it is a simulated series, and the exact value at node \( t_{11} \) cannot be determined. In the experiment section, the effectiveness of the proposed method will be compared. Algorithm 1 below is the pseudocode of the proposed method.

\[
\hat{y}_{Zhang} = w_{Zhang} \ast F_1 = \hat{y}_7 = 33.33 \tag{15}
\]

\[
\hat{y}_{Proposed} = w_{Proposed} \ast F_1 = \sum_{l=1}^{9} (w_l \times \hat{y}_l) = 38.10 \tag{16}
\]

4. Experiments

4.1. Construction Cost Index

The Engineering Cost Record (ENR) publishes the Construction Cost Index (CCI) once a month. The CCI data of the construction industry is worthy of reference data, and many scholars in the construction industry have conducted research. A total of 295 CCI data values (CCI data sets from January 1990 to July 2014) are used for forecasting time series.

Firstly, a fragment of length 11 of CCI is selected and named as time series \( U_2 \) as shown in Eq.17. 10 of these nodes are used for forecastings, and the last node in time is used as a target (underlined item in Eq.17). Visibility Graph \( G \) of time series \( U_2 \) is shown in Fig. 7.
Algorithm 1 The proposed forecasting method

1: Load time series $U$
2: for Each node pairs $(i, j)$ in $U$ do
3:  Calculate $A_{ij}$
4: Calculate the transition matrix $P$
5: Initialize forecasting vector $F$
6: for Each node $i$ in $U$ without last node do
7:  Calculate $F_i$
8: Initialize superposed similarity vector $S_{SRW}$
9: while True do
10:  for Each node $i$ in $U$ do
11:     if $t = 0$ then
12:        Initialize $\pi_i(0)$
13:     else
14:        $\pi_i(t) \leftarrow \pi_i(t + 1)$
15:     Calculate local similarity vector $S_{LRW}$
16:     $S_{SRW} = S_{SRW} + S_{LRW}$
17:     Backup current $S_{LRW}$ as $S'_{LRW}$
18:     if $t \neq 1$ and $S_{LRW} = S'_{LRW}$ then
19:         Break
20: Similarity vector $S = S_{SRW}$
21: Normalize $S$ as weight vector $w$
22: Forecasting Value $\hat{y} = w \cdot F$ return $\hat{y}$
$U_2 = \{(1, 5071), (2, 5070), (3, 5106), (4, 5167), (5, 5262), (6, 5260),
(7, 5252), (8, 5230), (9, 5255), (10, 5264), (11, 5278)\}$ (17)

Figure 7: Histograms and visibility graph $G$ of time series $U_2$ (the color of nodes $t_1 - t_9$ shows all the similarity of the last node $t_{10}$, and the more red the more similar).

The similarity vector $S_2$ corresponding to Fig. 7 is shown in Eq. 18. The similarity vector $S_2$ becomes weight vector $w_2$ after normalization, as shown in Eq. 19. Fig. 8 shows the generation process of the forecasting vector $F_2$.

$$S_2 = [0.8290, 0.8480, 0.8480, 0.8480, 1.4142, 0.7522, 0.7373, 0.3667, 0.9345]$$ (18)

$$w_2 = [0.1094, 0.1119, 0.1119, 0.1119, 0.1866, 0.0993, 0.0973, 0.0484, 0.1233] \left( \| w_2 \|^2 = 1 \right)$$ (19)
Through the forecasting vector $F_2$ and weight vector $w_2$, the predicted value of the proposed method $\hat{y}_{Proposed_2}$ and the predicted value of the previous work $\hat{y}_{Zhang_2}$ can be calculated, as shown in Tab.1. It can be seen that at node $t_{11}$, the proposed method is closer to the prediction target, and the error of previous research is larger. Of course, it is necessary to predict the whole time series and evaluate the prediction effect of the proposed method by using other comparison methods and error functions.

Table 1: Forecasting results of previous research and proposed methods on time series $U_2$
(numbers in parentheses indicate difference from target)

|          | Target | Zhang et al. [31] | Proposed |
|----------|--------|-------------------|----------|
|          | 5278.00| 5260.40(-17.6)    | 5275.90(-2.1) |

In order to better complete the prediction, the CCI data were analyzed. Fig.9 is the original data of CCI, and the specific nature of CCI cannot be analyzed. The stationarity analysis of CCI dataset is carried out. Here, `plot_acf` and `plot_pacf` in Python package `statsmodels.graphics.tsaplots` [35] are used to draw
autocorrelation and partial autocorrelation plots of CCI as shown in Fig.10. From Fig.10, the autocorrelation coefficient of CCI decreases slowly, so CCI is a non-stationary series. In addition, it can be seen from the partial autocorrelation coefficient that CCI data is only related to the former item.

Therefore, the stationarity of CCI data is determined again by difference. The autocorrelation coefficient of the first difference soon decays to near 0, which
is obviously a stationary series.

At the same time, the stationarity hypothesis test method is used to verify again. The current mainstream of hypothesis testing methods for stationarity is the unit root test, which tests whether there is a unit root in the series. If there is a unit root, it is a non-stationary series, and if there is no unit root, it is a stationary series. Dickey-fuller Test with GLS Detredding (DFGLS) is a unit root Test method proposed by Elliott, Rothenberg, and Stock [36]. The
implementation of DFGLS is used here to verify CCI stationarity using DFGLS from the Python arch.unitroot package [37].

The test result is shown in Fig.13, the original data p-value > 0.05, so the null hypothesis is not rejected, and CCI data is not stationary. According to CCI of the first difference’s p-value < 0.05, so the null hypothesis is rejected and the CCI data is stationary.

| Dickey-Fuller GLS Results of CCI | Dickey-Fuller GLS Results of CCI_diff |
|---------------------------------|--------------------------------------|
| Test Statistic: -0.500           | Test Statistic: -8.723               |
| P-value: 0.962                   | P-value: 0.000                       |
| Lags: 1                         | Lags: 4                              |
|---------------------------------|--------------------------------------|
| Trend: Constant and Linear Time Trend | Trend: Constant and Linear Time Trend |
| Critical Values: -3.48 (1%), -2.91 (5%), -2.62 (10%) | Critical Values: -3.48 (1%), -2.91 (5%), -2.62 (10%) |
| Null Hypothesis: The process contains a unit root. | Null Hypothesis: The process contains a unit root. |
| Alternative Hypothesis: The process is weakly stationary. | Alternative Hypothesis: The process is weakly stationary. |

CCI dataset is differenced

Figure 13: Hypothesis test result

After data analysis, CCI will be predicted. Here, the first six data are taken as the retained data, and the prediction starts from the seventh data of CCI. Fig.14 is the prediction effect diagram of the proposed method and the comparison methods. The prediction results of the proposed method and the comparison method are both ideal. ARIMA in the comparison method is an implementation of auto_arima in the Python package pmdarmia.arima [38] [39]. In order to more accurately compare the difference between the prediction results, mean absolute error (MAE), mean absolute percentage error (MAPE), Symmetric Mean Absolute Percentage Error (SMAPE), root mean square error (RMSE), and normalized root mean squared error (NRMSE) error indexes [33] are used to measure, as shown in Eq.21 - Eq.25.
\[ MAE = \frac{1}{N} \sum_{t=1}^{N} |\hat{y}(t) - y(t)| \]  
(21)

\[ MAPE = \frac{1}{N} \sum_{t=1}^{N} \frac{|\hat{y}(t) - y(t)|}{y(t)} \]  
(22)

\[ SMAPE = \frac{2}{N} \sum_{t=1}^{N} \frac{|\hat{y}(t) - y(t)|}{\hat{y}(t) + y(t)} \]  
(23)

\[ RMSE = \sqrt{\frac{1}{N} \sum_{t=1}^{N} |\hat{y}(t) - y(t)|^2} \]  
(24)

\[ NRMSE = \sqrt{\frac{\frac{1}{N} \sum_{t=1}^{N} |\hat{y}(t) - y(t)|^2}{y_{max} - y_{min}}} \]  
(25)

Figure 14: The result of the proposed method and the comparison methods on CCI

Tab.2 shows the prediction errors of the proposed method and the comparison methods. According to Tab.2, the prediction error of the proposed method is the smallest, which is a great progress compared with previous studies in MAE index. At the same time, the statistical method ARIMA has an ideal prediction effect, but the error is larger than the proposed method. The proposed
method will continue to validate the prediction effect on several competition datasets.

Table 2: The error of the proposed method and the comparison methods on CCI

| Method          | MAE  | MAPE | SMAPE | RMSE | NRMSE |
|-----------------|------|------|-------|------|-------|
| Zhang et al.    | 21.10| 0.3025| 0.3031| 30.74| 43.12 |
| Mao and Xiao    | 20.42| 0.2936| 0.2943| 30.02| 42.10 |
| ARIMA           | 20.50| 0.2971| 0.2974| 28.80| 40.40 |
| Proposed        | 19.67| 0.2847| 0.2850| 27.75| 38.92 |

Figure 15: Robustness verification: A record of each prediction

Meanwhile, in order to verify the robustness of the proposed method, the prediction errors of each time series are calculated. Fig.15 is the comparison of the prediction errors of the proposed method and the comparison methods at each time nodes. The error curve in red is the error curve of the proposed method. It can be seen that the error curve of the proposed method is below the curve of the comparison methods at most time points, indicating that the proposed method has small prediction error and strong robustness, and can
maintain an ideal prediction effect in the continuous prediction.

4.2. Competition datasets

In order to further verify the superiority of the proposed method, the proposed method will be validated in the competition dataset [40]. The Makridakis Competition (also known as the M Competition) is a series of open competitions to evaluate and compare the accuracy of different time series forecasting methods [41]. They were organized by a team led by forecasting researcher Spyros Makridakis and were first held in 1982. M1 [41, 42, 43, 44] and M3 [45, 46, 47, 48, 49] data sets are used this time which shown in Fig.16, and the descriptive characteristics of the data sets [40] are shown in Tab.3.

| Dataset | Number of Series | Minimum Length | Maximum Length |
|---------|-----------------|----------------|----------------|
| M1      | 1001            | 15             | 150            |
| M3      | 3003            | 20             | 144            |

To illustrate the proposed method accuracy, the proposed method compared with methods including Simple Exponential Smoothing (SES) [50], Theta [51], Trigonometric Box-Cox ARMA Trend Seasonal Model (TBATS) [52], Exponential Smoothing (ETS) [53], ARIMA [54], Pooled Regression Model (PR) [55], CatBoost [56], Feed-Forward Neural Network (FFNN) [57], DeepAR [58], N-BEATS [59], WaveNet [60], Transformer [61]. The proposed method is compared with the comparison methods in MAE and SMAPE. There are more than one time series in M1 and M3 data sets, so the average error of data sets is compared in the experiment. The experimental error in MAE is shown in Tab.4, and the experimental error in SMAPE is shown in Tab.5.

According to Tab.4 and Tab.5, the prediction accuracy of the proposed method is higher than that of the comparison method. Among the MAE errors, the proposed method is M3-yearly 514.40, and the comparison methods are all higher than 900, and the error of the proposed method is only about half of the comparison methods. However, the proposed method does not achieve the optimal performance in M1-monthly, because the short length of time series in
Figure 16: Partial time series in M1, M3 data sets
Table 4: Mean MAE error of proposed method and comparison methods

| Dataset | SES | Theta | TBATS | ETS | ARIMA | PR | CatBoost |
|---------|-----|-------|-------|-----|-------|----|---------|
| M1-Yearly | 171353.41 | 152999.26 | 103006.90 | 146110.11 | 145608.87 | 134246.38 | 215904.20 |
| M1-Quarterly | 2209.27 | 1981.86 | 2326.46 | 2088.15 | 2191.10 | 1630.38 | 1802.18 |
| M1-Monthly | 2259.04 | 2166.18 | 237.30 | 1905.28 | 2080.13 | 2085.25 | 2052.32 |
| M3-Yearly | 1022.27 | 957.40 | 1192.85 | 1031.40 | 1416.31 | 1018.48 | 1163.36 |
| M3-Quarterly | 751.96 | 486.31 | 561.77 | 513.06 | 559.40 | 519.30 | 593.29 |
| M3-Monthly | 743.41 | 623.71 | 630.59 | 626.46 | 654.80 | 692.97 | 732.00 |
| M3-Other | 277.83 | 215.35 | 189.42 | 194.98 | 193.02 | 234.43 | 318.13 |

| Dataset | FFNN | DeepAR | N-BEATS | WaveNet | Transformer | Proposed |
|---------|------|--------|---------|---------|-------------|-----------|
| M1-Yearly | 136238.80 | 152084.40 | 173200.20 | 284953.90 | 164637.90 | 68196.39 |
| M1-Quarterly | 1617.39 | 1951.14 | 1820.25 | 1855.89 | 1864.08 | 1383.17 |
| M1-Monthly | 2162.58 | 1860.81 | 1820.37 | 2184.42 | 2723.98 | 2007.33 |
| M3-Yearly | 1082.03 | 994.72 | 962.33 | 987.28 | 924.47 | 514.40 |
| M3-Quarterly | 528.47 | 519.35 | 494.85 | 523.04 | 719.62 | 406.19 |
| M3-Monthly | 692.48 | 728.81 | 648.60 | 699.30 | 798.38 | 580.32 |
| M3-Other | 240.17 | 247.56 | 221.85 | 245.29 | 239.24 | 91.80 |

Table 5: Mean SMAPE error of proposed method and comparison methods

| Dataset | SES | Theta | TBATS | ETS | ARIMA | PR | CatBoost |
|---------|-----|-------|-------|-----|-------|----|---------|
| M1-Yearly | 23.10 | 20.17 | 17.42 | 18.61 | 19.47 | 18.79 | 20.25 |
| M1-Quarterly | 18.10 | 16.35 | 16.65 | 17.47 | 16.62 | 16.67 | 17.60 |
| M1-Monthly | 17.43 | 16.53 | 15.15 | 15.05 | 15.65 | 15.20 | 16.51 |
| M3-Yearly | 17.76 | 16.76 | 17.37 | 17.00 | 18.84 | 17.13 | 20.07 |
| M3-Quarterly | 10.90 | 9.20 | 10.22 | 9.68 | 10.24 | 9.77 | 11.18 |
| M3-Monthly | 16.22 | 13.86 | 13.85 | 14.14 | 14.24 | 15.17 | 16.41 |
| M3-Other | 6.28 | 4.92 | 4.35 | 4.37 | 4.35 | 5.32 | 6.74 |

| Dataset | FFNN | DeepAR | N-BEATS | WaveNet | Transformer | Proposed |
|---------|------|--------|---------|---------|-------------|-----------|
| M1-Yearly | 18.20 | 18.72 | 20.52 | 21.25 | 18.96 | 8.73 |
| M1-Quarterly | 16.45 | 16.17 | 16.76 | 15.76 | 19.37 | 11.51 |
| M1-Monthly | 15.71 | 17.16 | 16.77 | 16.59 | 22.17 | 14.65 |
| M3-Yearly | 17.59 | 17.24 | 17.03 | 16.98 | 15.82 | 9.63 |
| M3-Quarterly | 9.90 | 9.93 | 9.47 | 9.72 | 13.17 | 8.10 |
| M3-Monthly | 15.33 | 15.74 | 14.76 | 15.43 | 17.13 | 13.98 |
| M3-Other | 5.41 | 5.57 | 4.97 | 5.09 | 5.78 | 2.21 |
M1-monthly does not reflect the advantages of the proposed method. Among the MAE errors, the accuracy of the proposed method exceeds that of all comparison methods on all data sets.

In order to compare the proposed method with the comparison method more intuitively, Friedman test and Nemenyi test [62] are used to evaluate the model performance and the resulting Friedman test figure is shown in Fig.17. It can be seen that the proposed method outperforms the comparison method on M1 and M3 datasets.
5. Conclusion

Recently, time series forecasting in the framework under complex network is paid great attention. In this paper, a new time series forecasting method is presented. The similarity generated after random walk is a key factor for forecasting in the network. A new weighted node similarity is constructed. The results show the efficiency of the proposed method. In future work, the proposed method will continue to be improved.

CRediT authorship contribution statement

Tianxiang Zhan: Conceptualization, Methodology, Software, Writing – original draft. Fuyuan Xiao: Writing – review & editing, Project administration, Funding acquisition, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This research is supported by the National Natural Science Foundation of China (No. 62003280) and Chongqing Talents: Exceptional Young Talents Project (CQYC202105031).
References

[1] Y. Huang, X. Mao, Y. Deng, Natural visibility encoding for time series and its application in stock trend prediction, Knowledge-Based Systems 232 (2021) 107478.

[2] H. Hewamalage, C. Bergmeir, K. Bandara, Global models for time series forecasting: A simulation study, Pattern Recognition 124 (2022) 108441.

[3] Y. Huang, Y. Gao, Y. Gan, M. Ye, A new financial data forecasting model using genetic algorithm and long short-term memory network, Neurocomputing 425 (2021) 207–218.

[4] D. Cheng, F. Yang, S. Xiang, J. Liu, Financial time series forecasting with multi-modality graph neural network, Pattern Recognition 121 (2022) 108218.

[5] W. Cheng, Y. Wang, Z. Peng, X. Ren, Y. Shuai, S. Zang, H. Liu, H. Cheng, J. Wu, High-efficiency chaotic time series prediction based on time convolution neural network, Chaos, Solitons & Fractals 152 (2021) 111304.

[6] K. Bandara, H. Hewamalage, Y.-H. Liu, Y. Kang, C. Bergmeir, Improving the accuracy of global forecasting models using time series data augmentation, Pattern Recognition 120 (2021) 108148.

[7] V. Le Guen, N. Thome, Deep time series forecasting with shape and temporal criteria, IEEE Transactions on Pattern Analysis and Machine Intelligence (2022).

[8] Y. Huang, Y. Deng, A new crude oil price forecasting model based on variational mode decomposition, Knowledge-Based Systems 213 (2021) 106669.

[9] T. Zhan, F. Xiao, A fast evidential approach for stock forecasting, Int. J. Intell. Syst. 36 (12) (2021) 7544–7562. doi:10.1002/int.22598
[10] X. Song, F. Xiao, Combining time-series evidence: A complex network model based on a visibility graph and belief entropy, Appl. Intell. 52 (9) (2022) 10706–10715. doi:10.1007/s10489-021-02956-5

[11] I. Ilic, B. Görgülü, M. Cevik, M. G. Baydoğan, Explainable boosted linear regression for time series forecasting, Pattern Recognition 120 (2021) 108144.

[12] X. Chen, L. Sun, Bayesian temporal factorization for multidimensional time series prediction, IEEE Transactions on Pattern Analysis and Machine Intelligence (2021).

[13] T. Wen, Y. Deng, Identification of influencers in complex networks by local information dimensionality, Inf. Sci. 512 (2020) 549–562. doi:10.1016/j.ins.2019.10.003

[14] S. Li, F. Xiao, The identification of crucial spreaders in complex networks by effective gravity model, Inf. Sci. 578 (2021) 725–749. doi:10.1016/j.ins.2021.08.026

[15] Q. Shang, Y. Deng, K. H. Cheong, Identifying influential nodes in complex networks: Effective distance gravity model, Inf. Sci. 577 (2021) 162–179. doi:10.1016/j.ins.2021.01.053

[16] H. Zhang, S. Zhong, Y. Deng, K. H. Cheong, LFIC: identifying influential nodes in complex networks by local fuzzy information centrality, IEEE Trans. Fuzzy Syst. 30 (8) (2022) 3284–3296. doi:10.1109/TFUZZ.2021.3112226

[17] T. Wen, K. H. Cheong, The fractal dimension of complex networks: A review, Inf. Fusion 73 (2021) 87–102. doi:10.1016/j.infus.2021.02.001

[18] Q. Zhang, M. Li, A betweenness structural entropy of complex networks, Chaos, Solitons & Fractals 161 (2022) 112264.
[19] L. C. Ribas, J. J. d. M. S. Junior, L. F. Scabini, O. M. Bruno, Fusion of complex networks and randomized neural networks for texture analysis, Pattern Recognition 103 (2020) 107189.

[20] Z. Liu, X. He, Y. Deng, Network-based evidential three-way theoretic model for large-scale group decision analysis, Inf. Sci. 547 (2021) 689–709. doi:10.1016/j.ins.2020.08.042

[21] L. Wang, L. Ma, C. Wang, N. Xie, J. M. Koh, K. H. Cheong, Identifying influential spreaders in social networks through discrete moth-flame optimization, IEEE Trans. Evol. Comput. 25 (6) (2021) 1091–1102. doi:10.1109/TEVC.2021.3081478

[22] D. Li, W. Song, J. Liu, Complex network evolution model based on turing pattern dynamics, IEEE Transactions on Pattern Analysis and Machine Intelligence (2022) 1–18 doi:10.1109/TPAMI.2022.3197276

[23] X. Yang, F. Xiao, An improved gravity model to identify influential nodes in complex networks based on k-shell method, Knowl. Based Syst. 227 (2021) 107198. doi:10.1016/j.knosys.2021.107198

[24] L. Chen, Y. Deng, K. H. Cheong, Probability transformation of mass function: A weighted network method based on the ordered visibility graph, Eng. Appl. Artif. Intell. 105 (2021) 104438. doi:10.1016/j.engappai.2021.104438

[25] F. Xiao, CED: A distance for complex mass functions, IEEE Trans. Neural Networks Learn. Syst. 32 (4) (2021) 1525–1535. doi:10.1109/TNNLS.2020.2984918

[26] F. Xiao, CEQD: A complex mass function to predict interference effects, IEEE Trans. Cybern. 52 (8) (2022) 7402–7414. doi:10.1109/TCYB.2020.3040770

[27] H. Li, Z. Liu, Multivariate time series clustering based on complex network, Pattern Recognition 115 (2021) 107919.
[28] K. Nguyen, C. Fookes, S. Sridharan, A. Ross, Complex-valued iris recognition network, IEEE Transactions on Pattern Analysis and Machine Intelligence (2022) 1–1. doi:10.1109/TPAMI.2022.3152857.

[29] B. Shang, P. Shang, Effective instability quantification for multivariate complex time series using reverse shannon-fisher index, Chaos, Solitons & Fractals 160 (2022) 112295.

[30] L. Lacasa, B. Luque, F. Ballesteros, J. Luque, J. C. Nuno, From time series to complex networks: The visibility graph, Proceedings of the National Academy of Sciences 105 (13) (2008) 4972–4975.

[31] R. Zhang, B. Ashuri, Y. Deng, A novel method for forecasting time series based on fuzzy logic and visibility graph, Advances in Data Analysis and Classification 11 (4) (2017) 759–783.

[32] W. Liu, L. Lü, Link prediction based on local random walk, EPL (Europhysics Letters) 89 (5) (2010) 58007.

[33] S. Mao, F. Xiao, Time series forecasting based on complex network analysis, IEEE Access 7 (2019) 40220–40229. doi:10.1109/ACCESS.2019.2906268.

[34] B. Ashuri, J. Lu, Time series analysis of enr construction cost index, Journal of Construction Engineering and Management 136 (11) (2010) 1227–1237.

[35] S. Seabold, J. Perktold, Statsmodels: Econometric and statistical modeling with python, in: Proceedings of the 9th Python in Science Conference, Vol. 57, Austin, TX, 2010, pp. 10–25080.

[36] C. Baum, R. Sperling, Dfgls: Stata module to compute dickey-fuller/gls unit root test (2001).

[37] K. Sheppard, S. Khrapov, G. Lipták, mikedeltalima, R. Capellini, alejandro cermeno, Hugle, esvhd, S. bot, A. Fortin, JPN, M. Judell, W. Li, A. Adams,
jbrockmendel, M. Rabba, M. E. Rose, N. Tretyak, T. Rochette, U. Leo, X. RENE-CORAIL, X. Du, B. Çelik, bashtage/arch: Release 5.3.1 (Jun. 2022). doi:10.5281/zenodo.6684078

[38] T. C. Nokeri, Forecasting using arima, sarima, and the additive model, in: Implementing Machine Learning for Finance, Springer, 2021, pp. 21–50.

[39] pmdarima: Arima estimators for python — pmdarima 1.8.5 documentation, https://alkaline-ml.com/pmdarima/ (Accessed on 08/18/2022).

[40] R. Godahewa, C. Bergmeir, G. I. Webb, R. J. Hyndman, P. Montero-Manso, Monash time series forecasting archive, in: Neural Information Processing Systems Track on Datasets and Benchmarks, 2021.

[41] S. Makridakis, M. Hibon, Accuracy of forecasting: An empirical investigation, Journal of the Royal Statistical Society: Series A (General) 142 (2) (1979) 97–125.

[42] R. Godahewa, C. Bergmeir, G. Webb, R. Hyndman, P. Montero-Manso, M1 yearly dataset (Jun. 2020). doi:10.5281/zenodo.4656193 URL https://doi.org/10.5281/zenodo.4656193

[43] R. Godahewa, C. Bergmeir, G. Webb, R. Hyndman, P. Montero-Manso, M1 quarterly dataset (Jun. 2020). doi:10.5281/zenodo.4656154 URL https://doi.org/10.5281/zenodo.4656154

[44] R. Godahewa, C. Bergmeir, G. Webb, R. Hyndman, P. Montero-Manso, M1 monthly dataset (Jun. 2020). doi:10.5281/zenodo.4656159 URL https://doi.org/10.5281/zenodo.4656159

[45] S. Makridakis, M. Hibon, The m3-competition: results, conclusions and implications, International journal of forecasting 16 (4) (2000) 451–476.

[46] R. Godahewa, C. Bergmeir, G. Webb, R. Hyndman, P. Montero-Manso, M3 yearly dataset (Jun. 2020). doi:10.5281/zenodo.4656222 URL https://doi.org/10.5281/zenodo.4656222
[47] R. Godahewa, C. Bergmeir, G. Webb, R. Hyndman, P. Montero-Manso, M3 quarterly dataset (Jun. 2020). doi:10.5281/zenodo.4656262
URL https://doi.org/10.5281/zenodo.4656262

[48] R. Godahewa, C. Bergmeir, G. Webb, R. Hyndman, P. Montero-Manso, M3 monthly dataset (Jun. 2020). doi:10.5281/zenodo.4656298
URL https://doi.org/10.5281/zenodo.4656298

[49] R. Godahewa, C. Bergmeir, G. Webb, R. Hyndman, P. Montero-Manso, M3 other dataset (Jun. 2020). doi:10.5281/zenodo.4656335
URL https://doi.org/10.5281/zenodo.4656335

[50] E. Ostertagová, O. Ostertag, The simple exponential smoothing model, in: The 4th International Conference on modelling of mechanical and mechatronic systems, Technical University of Košice, Slovak Republic, Proceedings of Conference, 2011, pp. 380–384.

[51] V. Assimakopoulos, K. Nikolopoulos, The theta model: a decomposition approach to forecasting, International journal of forecasting 16 (4) (2000) 521–530.

[52] A. M. De Livera, R. J. Hyndman, R. D. Snyder, Forecasting time series with complex seasonal patterns using exponential smoothing, Journal of the American statistical association 106 (496) (2011) 1513–1527.

[53] R. Hyndman, A. B. Koehler, J. K. Ord, R. D. Snyder, Forecasting with exponential smoothing: the state space approach, Springer Science & Business Media, 2008.

[54] G. E. Box, G. M. Jenkins, G. C. Reinsel, G. M. Ljung, Time series analysis: forecasting and control, John Wiley & Sons, 2015.

[55] J. R. Trapero, N. Kourentzes, R. Fildes, On the identification of sales forecasting models in the presence of promotions, Journal of the operational Research Society 66 (2) (2015) 299–307.
[56] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, A. Gulin, Catboost: unbiased boosting with categorical features, Advances in neural information processing systems 31 (2018).

[57] I. Goodfellow, Y. Bengio, A. Courville, Deep learning, MIT press, 2016.

[58] D. Salinas, V. Flunkert, J. Gasthaus, T. Januschowski, Deepar: Probabilistic forecasting with autoregressive recurrent networks, International Journal of Forecasting 36 (3) (2020) 1181–1191.

[59] B. N. Oreshkin, D. Carpov, N. Chapados, Y. Bengio, N-beats: Neural basis expansion analysis for interpretable time series forecasting, arXiv preprint arXiv:1905.10437 (2019).

[60] A. Borovykh, S. Bohte, C. W. Oosterlee, Conditional time series forecasting with convolutional neural networks, arXiv preprint arXiv:1703.04691 (2017).

[61] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[62] D. G. Pereira, A. Afonso, F. M. Medeiros, Overview of friedman’s test and post-hoc analysis, Communications in Statistics-Simulation and Computation 44 (10) (2015) 2636–2653.