Journal of Universal Computer Science, vol. 30, no. 11, 1569-1594
submitted: 18/10/2023, accepted: 11/4/2024, appeared: 28/10/2024 CC BY-ND 4.0

Multi-Step-Ahead Time Series Forecasting using Deep Learning and Fuzzy Time Series-based Error Correction Method

Samit Bhanja
(Government General Degree College, Singur, West Bengal, India
https://orcid.org/0000-0002-0411-4502, samit.compsc@singurgovtcollege.org)

Banani Ghose
(Haldia Institute of Technology, Haldia, West Bengal, India
https://orcid.org/0000-0001-6558-1334, bananighose@hithaldia.in)

Abhishek Das
(Aliah University, Kolkata, West Bengal, India
https://orcid.org/0000-0002-5546-6188, adas@aliah.ac.in)

Abstract: Recently time series forecasting has become one of the prime application areas of climatology, economics and industries. Many research works are conducted to forecast the time series more accurately. But few of them are concentrated on predicting the time series over an extended future horizon, and there is also a scope to improve their forecasting accuracy. This work proposes a multi-step-ahead foresting method to produce a stable and accurate forecasting result for the extended future horizon. Firstly, a deep learning-based forecasting model is proposed to predict the time series. Secondly, a fuzzy time series-based error correction model is implemented to enhance the prediction performance of the deep learning model. Here to optimize all the fuzzy time series (FTS) parameters in an integrated way, an integrated butterfly optimization (FTS-IBO) algorithm is proposed. In this study, two different types of real-world multivariate time series datasets are used to analyze the forecasting performance of the proposed model. The performance of the proposed FTS-IBO algorithm is compared with the traditional butterfly optimization (FTS-BO) algorithm. The experimental results show that the FTS-IBO technique is superior to the FTS-BO technique. The forecasting performance of the proposed model has also compared the other state-of-the-art models, and the simulation results exhibit that the proposed model produces a more accurate prediction performance for multi-step-ahead time series forecasting problems compared to other models.

Keywords: Time series forecasting, Error correction, Fuzzy time series, Butterfly optimization, Deep learning
Categories: I.2.1, I2.6, I.2.8, I.5.1, I.5.2, I.5.4
DOI: 10.3897/jucs.114357

1 Introduction

Recently, with the rapid advancement of sensor and communication technology, a large volume of data is generated in every timestamp. These massive amounts of data (big data) pose a new challenge for researchers to analyze and extract valuable information from these big data. Since these big data are collected over fixed time intervals and

ordered chronologically, called time series (TS) data. The TS data can be divided into two categories based on the data points collected at each time interval. If a single data point is collected at each time interval, then the time series is called univariate time series (UTS). On the contrary, in the multivariate time series (MTS), multiple data points are collected at each timestamp. Time series forecasting (TSF) has become one of the emerging research areas in extensive data analysis; prediction with high accuracy is essential, which is based on past and current TS data. There are many applications such as air quality, weather, energy consumption, stock market, anomaly detection, medical monitoring, etc. The TSF with two or more steps ahead is known as a multi-step-ahead TSF. Recently the multi-step-ahead TSF in various domains has gained interest not only for the researchers but also for the stakeholders, policymakers, and practitioners of that domain. The multi-step-ahead TSF forecasting is challenging due to accumulated error and lack of information [Weigend, 2018].

The recursive and direct strategies are the two most commonly used modeling strategies for the multi-step-ahead TSF models. The prediction model that adopts the recursive strategy, predicts the one-step-ahead future data at a time, and this predicted data is taken as the input to predict the next-step-ahead data. In this way, the recursive strategy-based forecasting model predicts the multi-step-ahead future data. Since this strategy uses the past predicted value to predict the future, the affected squared error in one step is propagated to the next onward steps. And as a result, it suffers from the error accumulation problem [Bao et al., 2014, Chevillon, 2007]. The direct strategy for multi-step-ahead forecasting is first proposed by Cox in [Cox, 1961]. In contrast to the recursive strategy, it constructs a forecasting model for each future horizon based on past values. So, this type of forecasting model only suffered from the associated multi-step-ahead errors, instead of accumulated squared errors [Cox, 1961, Franses and Legerstee, 2010].

In recent times deep learning (DL) models have become one of the most popular approaches for TSF problems [Reyes and Ventura, 2019, Jiang et al., 2021]. In contrast to the statistical models, DL models have exhibited great potential for extracting complex nonlinear features from the TS data [Reyes and Ventura, 2019]. In the last few years, computing power has increased significantly, and that allows for the construction of deeper DL models, which remarkably enhances the learning capacity of the models. When the DL models are applied to forecast the TS data in the longer future, the prediction errors also increase. So, there is a need to adopt some error correction techniques to enhance the forecasting performance.

The fuzzy time series (FTS) forecasting models are another popular soft computing tool that is widely used in various TSF problems [Singh and Borah, 2014]. In contrast to the traditional TS, the FTS is more descriptive because it can provide semantic meaning for uncertain data [Van Tinh, 2020]. In comparison to DL models, FTS forecasting models can provide better prediction accuracy for a smaller volume of the historical dataset. The success of FTS forecasting models depends on the selection of their hyperparameters. The hyperparameters that mostly affect the performance of these models are the number of fuzzy intervals of the universe of discourse (UOD), interval lengths, and fuzzy order. So, to obtain the best performance from the FTS forecasting models, the adaptation of the hyperparameter optimization technique is essential.

In this work, a multi-step-ahead TS forecasting method is proposed to forecast the TS data for an extended future horizon. The main contributions of the proposed work are summarized as follows:

– A data preprocessing technique is proposed to generate the quality dataset from the raw MTS input dataset to improve the prediction performance.

– A basic deep learning-based prediction model is constructed by combining 1D convolution neural networks (1D-CNNs) and bidirectional gated recurrent units (Bi-GRUs) for multi-step-ahead TS forecasting (here the direct strategy is adopted for multi-step-ahead forecasting).

– Fuzzy time series-based error correction model is proposed to correct the residual errors incurred by the basic deep learning-based prediction model.

– An integrated butterfly optimization (FTS-IBO) algorithm is proposed to optimize all the FTS hyperparameters in an integrated way.

The performance of FTS-IBO is compared with the FTS-BO algorithm, and the experimental results show that the performance of FTS-IBO outperforms the FTS-BO algorithm. The performance of the proposed multi-step-ahead TS forecasting method is compared with the other state-of-the-art methods. The simulation results exhibit that the proposed method outruns other methods.

The rest of the paper is organized as follows:
Literature survey is presented in section 2. A detailed description of the proposed methodology is presented in section 3. In Section 4, the experimental process including dataset description, evaluation metrics, experimental setup, and the result & discussion is presented. The conclusion is drawn in section 5.

2 Literature Review

In recent times, time series forecasting (TSF) has become one of the emerging research fields. It has a wide range of application areas, such as - air quality [Du et al., 2019a, Bhanja and Das, 2021], weather [Fathi et al., 2021], energy consumption [Peng et al., 2022], stock market [Bhanja and Das, 2022a, Sezer et al., 2020], anomaly detection [Pei et al., 2022, Zhang et al., 2021a], medical monitoring [Ali et al., 2021], etc. In the past two decades, a lot of research work has been conducted to successfully forecast the time series. In these works, mainly statistical methods or machine learning methods including ARIMA [Díaz-Robles et al., 2008], SVM new11, HMM [Dong et al., 2009], ANN [Zhou et al., 2014], etc., are used. In the last few years, the DL models have gained the interest of researchers for the TSF problems. However, the DL models generally have low biases and high variances [Ju et al., 2018], but an individual DL model can't maintain its high forecasting accuracy and robustness to forecast the TS data in a dynamic application environment, especially to predict the extended future horizon. Recent research [Du et al., 2019b, Zhang et al., 2021b, Ghose et al., 2022] shows that hybrid DL models can produce better performance compared to shallow DL models. In [Du et al., 2019c], authors proposed a hybrid deep learning model by combining the one-dimensional Convolutional Neural Networks (1D-CNNs) and Bi-directional Long Short-term Memory Networks (Bi-LSTM) to forecast the air quality TS data. They show that their proposed hybrid model produces better prediction results compared to the shalow models. The DL models are also applied to forecast the TS data for the extended future horizons. Recent research shows that prediction errors increase as the forecasting of future horizons increases. Researchers aim to correct prediction errors and proposed various error correction techniques to correct prediction errors [Ding et al., 2019, Kim et al., 2022, Xu et al., 2021].

In recent times, the fuzzy time series (FTS) models have gained interest from researchers to solve TSF problems [Singh and Borah, 2014, Van Tinh, 2020, Chen, 2002].

In [Bajestani and Zare, 2011], authors applied a Type-2 fuzzy time series model to forecast the Taiwan stock index (TAIEX). The statistical models are combined with the FTS to predict the AQI in Kuala Lumpur in [Bajestani and Zare, 2011]. This study shows that the combined model produces better forecasting results compared to other shallow models. The success of these FTS forecasting models depends on the values of their hyperparameters. Recently the particle swarm optimization (PSO) algorithm has been widely used to optimize the FTS hyperparameters [Singh and Borah, 2014, Van Tinh, 2020]. Most of these works concentrated on finding the optimal length of fuzzy intervals in UOD. Butterfly optimization (BO) is another popular optimization algorithm proposed by Sankalp et al. in [Arora and Singh, 2019]. In [Bhanja et al., 2022], authors proposed the FTS-BO algorithm to find the optimal value of the fuzzy interval lengths, and they show that the performance of the FTS-BO is better than other FTS optimization algorithms.

In this study, by comparing the related work, we have proposed a deep leaning-based prediction model to the TS data more accurately for the extended future horizon. Here to enhance the forecasting accuracy, a fuzzy time series-based error correction method is proposed. To optimize all the hyperparameters of the FTS forecasting method, an integrated butterfly optimization algorithm is proposed.

3 Methodology

Figure 1 shows the block diagram of the proposed multi-step-ahead prediction method. The central concept of this work is to forecast the time series data for the longer future horizon with a high degree of accuracy. Mathematically the proposed methodology can be expressed as follows:

$$y'_{t+l} = \hat{y}_{t+l} + e_{t+l}, \tag{1}$$

$$\hat{y}_{t+l} = f(y_t), \tag{2}$$

$$e_{t+l} = g(\hat{y}_{t+l}), \tag{3}$$

here, y_t is the processed time series data at the timestamp t, y'_{t+l} is the l-timestamp-ahead predicted value. At first, the function $f()$ is applied to the input time series data y_t to generate the l timestamp-ahead intermediate predicted value $\hat{y}_{t+l}$. Then the error e_{t+l} is predicted by applying the function $g()$ on $\hat{y}_{t+l}$. Finally, the actual forecast value y'_{t+l} is determined using Equation 1. The entire methodology is divided into three processing steps, (1) data preprocessing, (2) basic forecasting model construction, and (3) residual-error correction method.

3.1 Data Preprocessing

This data preprocessing technique is adopted to prepare a quality dataset from the raw MTS dataset that can be directly applied to the basic prediction model for the multi-step-ahead prediction of the time series. In this experiment, to increase the prediction accuracy of the proposed method, the MTS ($TS_1, TS_2, \ldots, TS_M$) is considered. Here the MTS is framed with the M number of times series (TS), and TS_M is considered as the target times series of the proposed work. The data preprocessing technique is as follows:

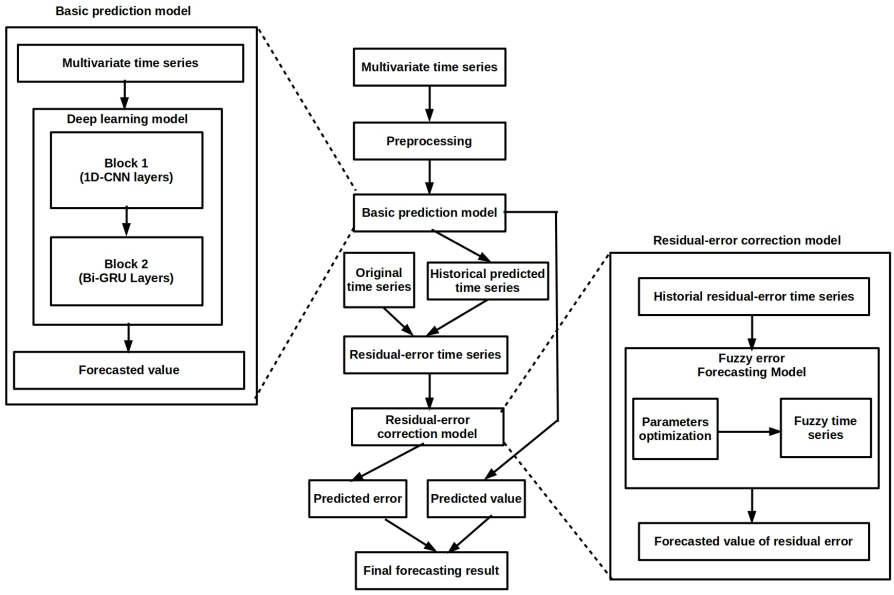

Figure 1: Block diagram of the proposed forecasting framework.

Step 1: To find the impact of the historical data on the current value, the autocorrelation function (ACF) of the target time series TS_M is determined. The ACF for the time series $y_t; (t = 1, 2, \ldots, N)$ is as follows:

$$\rho_a = \frac{\sum_{t=1}^{N-a}(y_t - \bar{y})(y_{t+a} - \bar{y})}{\sum_{t=1}^{N}(y_t - \bar{y})^2}, a = 1, 2, \ldots, N, \tag{4}$$

here ρ_a is the autocorrelation coefficient for lag period a, and N is the number of data samples of the time series TS_M. y_t is the tth observation and $\bar{y}$ is the mean value of the time series TS_M. From these autocorrelation coefficients $\rho_a(a = 1, , 2, \ldots, N)$, we find the lag period ω, where ω is the length of the historical data that has the highest impact on the current data.

Step 2: Now, the MTS is divided into multiple segments $(Seg_i; i = 1, 2, \ldots, n)$ of the dimension $\omega \times M$. These segments are constructed by striding a window of length ω with a striding length of s. The number of segments n is calculated as follows:

$$n = N - (\omega - s). \tag{5}$$

These segments are considered as the input of the forecasting model. Here the segment Seg_i contains the data of the timestamps $\{i, (i + 1), (i + 2), \ldots, (i + \omega - 1)\}$. So for the current timestamp t, the corresponding input segment will be Seg_{cur}, where $cur = t - \omega + 1$.

Step 3: In this step, for each segment $Seg_i(i = 1, 2, \ldots, n)$ the target value $trgt_i$ is determined. This work aims to forecast the multi-step-ahead time series data and to do that, the direct strategy is adopted to predict the l step-ahead time series data. The reason

to select the direct strategy over the recursive strategy is that in the recursive strategy, the error generated in one iteration is propagated and accumulated in the next onwards all iterations. And consequently, the performance of the recursive strategy is degraded with the increase in the value of l [Wang et al., 2016]. The proposed method is applied to forecast the value of the time series TS_M for the timestamp $(t + l)$ when the current timestamp is t. If we consider the input segment at the current timestamp is Seg_{cur}, so the target value y_L is the Lth timestamp's value of TS_M, where, $L = cur + \omega - 1 + l$.
Step 4: Now, all these input segments $Seg_i(i = 1, 2, , \ldots, n)$ are concatenated along the time axis to form the input dataset I. Similarly, the target values $trgt_i(i = 1, 2, \ldots, n)$ are also combined to form the target dataset T.

The dataset I and T generated in this section are used for the prediction purpose of the proposed work.

3.2 Basic Prediction Model

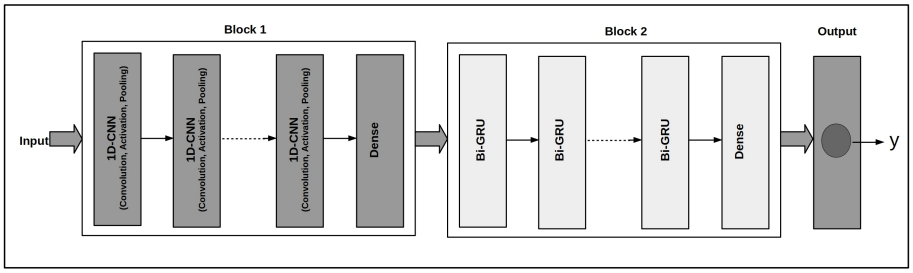

Figure 2: Architecture of the basic prediction model.

Here a basic prediction model is constructed to generate the l timestamp-ahead intermediate forecasting value $\hat{y}_{t+l}$. This basic prediction model is assembled by using the two popular deep-learning architectures such as CNN and GRU. The architecture of the proposed basic prediction model is shown in Figure 2. This deep learning-based prediction model is formulated as per our previously published paper [Bhanja and Das, 2022b]. The model consists of two blocks. The first block consists of a series of 1D convolution neural network (1D-CNN) layers followed by a dense layer. The 1D-CNN is excellent to extract the spatial features from the input matrix [Zhong et al., 2019]. So, here the series of 1D-CNN layers are applied to extract the higher-order spatial features matrix from the input segment. The dense layer is incorporated to convert the features matrix to a features vector. This vector is applied as the input of the next block. The next block is build-up by a series of bidirectional gated recurrent unit (Bi-GRU) layers and a dense layer. The Bi-GRU is used to extract the long-term dependency features from the input vector, and finally, the dense layer yields the output of the model.

The basic prediction model is tuned during the simulation process to ascertain the best forecasting result from the model. And by tuning the model, we find the optimal number of layers for Block 1 and Block 2 respectively for which the model produces the best forecasting results. The output of the prediction model is the l step-ahead intermediate predicted value.

3.3 Residual-Error Correction Model

The forecasting accuracy of the basic prediction model is needed to be improved. So, in this section, a residual error correction model is proposed to correct the forecasting errors. The main theme of this residual error correction model is to predict the error e_{t+l} incurred in the intermediate forecast value $\hat{y}_{t+l}$ of the basic prediction model. Then the predicted error is added to the intermediate forecast value $\hat{y}_{t+l}$ to obtain the final forecast value y'_{t+l} as per the Equation 1.

Here a fuzzy time series (FTS) forecasting method [Zadeh, 1973, Song and Chissom, 1993b, Song and Chissom, 1993a] is used to predict the residual errors. An error dataset is constructed for the FTS forecasting method, and to optimize the FTS parameters, an integrated butterfly optimization (FTS-IBO) algorithm is proposed.

3.3.1 Error Dataset Generation

The historical dataset is essential for any forecasting method. So, to forecast the error incurred in the intermediate predicted value ($\hat{y}$) of the basic prediction model, an error dataset is generated. After the successful completion of the training process of the basic prediction model, the residual error is calculated. Let t be the current timestamp, and the basic prediction model is applied to forecast the l step-ahead value. So, the residual error is computed as per the following formulae:

$$e_{t'} = y_{t'} - \hat{y}_{t'}, \tag{6}$$

where $y_{t'}$ and $\hat{y}_{t'}$ are the actual and the intermediate forecast value for the timestamp t' respectively, where $t' = t + l$.

Now a residual error time series is generated by combining all such errors along with the corresponding timestamps. In Table 1, we represent the snapshot (01/01/2021) of the residual error time series dataset for the original dataset DS1.

3.3.2 FTS Forecasting Method

Here, to forecast the residual error, FTS forecasting is proposed. The reason for selecting the FTS forecasting method over other forecasting methods is that the FTS forecasting method can handle uncertain and vague data. It can also provide more accurate forecasting results for a smaller volume of the historical dataset [Van Tinh, 2020, Bose and Mali, 2019]. The input of this method is the error time series dataset which is generated in section 3.3.1. The complete process of the FTS forecasting method is described in the following steps:

Step 1: Define the universe of discourse (UOD).

Let us consider that $E(t)$ is the error time series dataset, e_{min} and e_{max} be the minimum and maximum values of the dataset $E(t)$ respectively. Then the UOD is defined as $[e_{min} - \phi, e_{max} + \phi]$. Here ϕ is the standard deviation of the time series.

Step 2: Partition the UOD into several overlapping intervals.

Here UOD is divided into k number of equal length adjoining intervals, where δ is the interval length. The interval length δ, and the mid-value c_τ of each intervals $u_\tau (\tau = 1, 2, \ldots, k)$ are determined as per the following formula:

$$\delta = 2 \times \frac{(e_{max} + \phi) - (e_{min} - \phi)}{k} + \eta, \tag{7}$$

Date	Time	Error($\mu g/m^3$)
01/01/2021	01:00	0.83
01/01/2021	02:00	-0.88
01/01/2021	03:00	-1.99
01/01/2021	04:00	-0.88
01/01/2021	05:00	-0.97
01/01/2021	06:00	-1.06
01/01/2021	07:00	-1.11
01/01/2021	08:00	-0.23
01/01/2021	09:00	0.66
01/01/2021	10:00	-2.18
01/01/2021	11:00	2.03
01/01/2021	12:00	-0.98
01/01/2021	13:00	-0.82
01/01/2021	14:00	-1.83
01/01/2021	15:00	-0.47
01/01/2021	16:00	0.67
01/01/2021	17:00	2.15
01/01/2021	18:00	-0.97
01/01/2021	19:00	-0.69
01/01/2021	20:00	-0.51
01/01/2021	21:00	-0.96
01/01/2021	22:00	-0.61
01/01/2021	23:00	0.98
01/01/2021	24:00	-1.12
⋮	⋮	⋮

Table 1: Residual-error time series of DS1(01/01/2021).

$$u_1 = [(e_{min} - \phi), (e_{min} - \phi + \delta)], \tag{8}$$

$$u_\tau = [u_\tau^{low}, u_\tau^{high}], \tau = 2, 3, \ldots, k, \tag{9}$$

$$u_\tau^{low} = (e_{min} - \phi) + i \times (\delta - 1) - 1, \tau = 2, 3, \ldots, k, \tag{10}$$

$$u_\tau^{high} = (e_{min} - \phi) + (i + 1) \times (\delta - 1), \tau = 2, 3, \ldots, k, \tag{11}$$

$$c_\tau = \frac{1}{2} \times (u_\tau^{low} + u_\tau^{high}), \tau = 1, 2, \ldots, k, \tag{12}$$

here η is a randomly generated small positive number. u_τ is the τth interval and u_τ^{low},

u_τ^{high} are the lower bound and the upper bound of that interval.

Step 3: Assign linguistic variables for each interval and define a fuzzy set for each interval.

Here we define linguistic variable for each interval generated in step 2. So there will be the k number of linguistic variable, since the number intervals are k. Therefore the fuzzy set $\tilde{A_\tau}$ is generated for each linguistic variable as follows:

$$\tilde{A_\tau} = \frac{f_{\tilde{A_\tau}}(u_1)}{u_1} + \frac{f_{\tilde{A_\tau}}(u_2)}{u_2} + \cdots + \frac{f_{\tilde{A_\tau}}(u_k)}{u_k}, \tag{13}$$

here $f_{\tilde{A_\tau}}$ represents the degree of the membership of u_τ in the fuzzy set $\tilde{A_\tau}$, and it belongs to $[0, 1]$. Here to determine the membership value, the triangular membership function is applied. Let x be an observation that belongs to the interval u_τ. Then the fuzzy membership value of x in fuzzy set $\tilde{A_\tau}$ is calculated as follows:

$$f_{\tilde{A_\tau}}(u_\tau) = \begin{cases} 0, & x < u_\tau^{low} \\ \frac{x - c_\tau}{c_\tau - u_\tau^{low}}, & u_\tau^{low} \leq x \leq c_\tau \\ \frac{u_\tau^{high} - x}{u_\tau^{high} - c_\tau}, & c_\tau < x \leq u_\tau^{high} \\ 0, & x > u_\tau^{high} \end{cases} \tag{14}$$

Step 4: Fuzzify each observation of the error time series.

In this step, each observation of the error time series is mapped to a linguistic value. In this fuzzification process, at first, the degree of membership value of each observation is calculated belonging to each of the fuzzy set $A_\tau (\tau = 1, 2, \ldots, k)$. Then we identify the fuzzy set A_τ where the maximum membership value occurs for an observation x, and that fuzzy set A_τ is considered as the fuzzified value of x.

Step 5: Create m-order fuzzy logic relationships (M-FLRs).

The m-order fuzzy logic relationship established for the timestamp t is as follows:

$$F(t-m), F(t-m+1), \ldots, F(t-2), F(t-1) \rightarrow F(t), \tag{15}$$

here $F(t-m), F(t-m+1), \ldots, F(t-2), F(t-1)$ is called the previous state and $F(t)$ is called the current state. Then the M-FLRs are generated by replacing each $F(t-i); (i = m, m-1, \ldots, 2, 1, 0)$ by its corresponding linguistic value (fuzzy set). In the M-FLRs, the right-hand side must be unique, which means no fuzzy set cannot appear on the right-hand side of an M-FLR more than once. In this error forecasting method, we have tried to predict the l timestamps ahead error. So the Eq. 15 is modified to generate the M-FLR for the timestamp t and the modified equation is as follows:

$$F(t-m+1-l), F(t-m+2-l), \ldots, F(t-1-l), F(t-l) \rightarrow F(t). \tag{16}$$

For example, considering $m = 2$ and $l = 2$, the Eq. 16 can be represented as follows:

$$F(t-3), F(t-2) \rightarrow F(t). \tag{17}$$

Let $\tilde{A_r}$ be the fuzzy set corresponding to the timestamp t. And $\tilde{A_p}$ and $\tilde{A_q}$ are the fuzzy sets analogous to the timestamps $(t-3)$ and $(t-2)$ respectively. Therefore the M-FLR

is formulated as follows:

$$\text{M-FLR: } \tilde{A}_p, \tilde{A}_q \rightarrow \tilde{A}_r, \tag{18}$$

here the M-FLR is called the 2nd order fuzzy logic relationship.

Step 6: Form m-order fuzzy logic relationship groups (M-FLRGs).

M-FLRGs are constructed by combining all the M-FLRs whose previous states are the same. Let us consider two M-FLRs are $\tilde{A}_p, \tilde{A}_q \rightarrow \tilde{A_{r1}}$ and $\tilde{A}_p, \tilde{A}_q \rightarrow \tilde{A_{r2}}$. Therefore the M-FLRG is as follows:

$$\text{M-FLRG: } \tilde{A}_p, \tilde{A}_q \rightarrow \tilde{A_{r1}}, \tilde{A_{r2}}. \tag{19}$$

Then all M-FLRGs are mapped to their corresponding timestamps.

Step 7: Defuzzify each M-FLRG [Abhishekh et al., 2018].

In this step, to predict the residual error, we defuzzify each M-FLRG. In the defuzzification process, two different rules are used. Here Rule 1 is applied to the training process of the forecasting method. Whereas Rule 2 is used for the testing phase.

Rule 1: Let us consider at the timestamp t, the M-FLRG is $\tilde{A}_{\alpha_1}, \tilde{A}_{\alpha_2}, \ldots, \tilde{A}_{\alpha_g} \rightarrow \tilde{A}_{\beta_1}, \tilde{A}_{\beta_2}, \ldots, \tilde{A}_{\beta_h}$. Then the defuzzified value for the previous state is calculated as follows:

$$e_{prv} = \frac{c_{\alpha_1} \times w_{\alpha_1} + c_{\alpha_2} \times w_{\alpha_2} + \cdots + c_{\alpha_g} \times w_{\alpha_g}}{w_{\alpha_1} + w_{\alpha_2} + \cdots + w_{\alpha_g}}, \tag{20}$$

here $c_{\alpha_1}, c_{\alpha_2}, \ldots, c_{\alpha_g}$ are the mid-values of the intervals $u_{\alpha_1}, u_{\alpha_2}, \ldots, u_{\alpha_g}$ respectively and $w_{\alpha_1}, w_{\alpha_2}, \ldots, w_{\alpha_g}$ are the maximum membership value corresponding to the fuzzy sets $\tilde{A}_{\alpha_1}, \tilde{A}_{\alpha_2}, \ldots, \tilde{A}_{\alpha_g}$ respectively.

Similarly the defuzzified value for the current state is calculated as follows:

$$e_{curr} = \frac{c_{\beta_1} \times w_{\beta_1} + c_{\beta_2} \times w_{\beta_2} + \cdots + c_{\beta_g} \times w_{\beta_g}}{w_{\beta_1} + w_{\beta_2} + \cdots + w_{\beta_g}}. \tag{21}$$

Now the predicted residual error for the timestamp t is calculated as follows:

$$e_t = \frac{e_{prv} + e_{curr}}{2}. \tag{22}$$

Rule 2: This rule is applied for the testing phase of the residual error correction method. During the testing process, only the input data is available, so M-FLRG contains only the previous state. Therefore, the error is predicted by applying the defuzzification process to the previous state of the M-FLRG only. Assuming that at the timestamp t, the M-FLRG is $\tilde{A}_{\gamma_1}, \tilde{A}_{\gamma_2}, \ldots, \tilde{A}_{\gamma_f} \rightarrow \#$. Here # is represented as Empty current state. Now the defuzzified value of the previous state e_{prv} is determined as per the Eq. 20. Now the predicted residual error e_t at the timestamp t is determined as follows:

$$e_t = e_{prv}. \tag{23}$$

3.3.3 FTS Optimization

The success of any fuzzy time series (FTS) forecasting method largely depends on the number of fuzzy intervals, interval lengths, and the fuzzy order [Singh and Borah, 2014, Van Tinh, 2020]. Here the Butterfly Optimization Algorithm (BOA) is used to optimize the FTS hyperparameters. In recent times, various researchers deployed the

BO algorithm for different types of optimization problems and got satisfactory results in terms of performance and speed of convergence [Bhanja et al., 2022, Abdulhussein et al., 2021, Li et al., 2022, Arora and Singh, 2017]. Inspired by their research works, in this proposed work, BOA is applied to optimize the FTS hyperparameters.

Butterfly Optimization Algorithm (BOA): The BOA was proposed by Arora and Singh [Arora and Singh, 2019]. It is based on the biological phenomenon of butterflies where the foraging and mating are simulated to determine the locations of food or mating partner. Butterflies emit a fragrance that proliferates over a certain distance. The search-agents (butterflies) use this fragrance as a guide for movement in the BOA. Each butterfly moves towards the best butterfly that produces more fragrance than other butterflies, called the global search. If a butterfly fails to detect the fragrance of the best butterfly's fragrance within the search space, this butterfly will move randomly within the search space, called the local search. The magnitude of fragrance (mf) is calculated as per the following equation:

$$mf = \mu \times I^e, \tag{24}$$

where μ and I are the sensory modality and stimulus intensity. e is the power exponent of fragrance and it lies between [0, 1]. In the global search phase, the butterfly (bf_i) updates the position to the fittest butterfly at the $(t+1)$th iteration, which is given by

$$bf_i(t+1) = bf_i(t) + (r^2 \times G^* - bf_i(t)) \times mf_i, \tag{25}$$

where $bf_i(t)$ represents the solution vector of the ith butterfly for tth iteration and G^* serves as the best solution at the tth iteration. r is a random number and $r \in [0, 1]$. In the local search phase, the update position of the butterfly (bf_i) at the $(t+1)$th iteration, which is determined by

$$bf_i(t+1) = bf_i(t) + (r^2 \times bf_j(t) - bf_k(t)) \times mf_i, \tag{26}$$

where $bf_j(t)$ is the solution vector of jth butterfly for the tth iteration and $bf_k(t)$ represents the solution vector of the kth butterfly for the tth iteration. Each butterfly searches for food or its mating partner based on the global or local search. So there is a switch probability p defined in the BOA to switch between the global and the local search.

Integrated BOA: In this study, we introduce an integrated FTS-BO (FTS-IBO) algorithm to optimize the interval lengths along with the number of intervals and the fuzzy order. The proposed FTS-IBO algorithm is presented in algorithm 1.

Algorithm 1 FTS-IBO algorithm

1: Initialization search space of number of intervals $[k_{min}, k_{min+1}, \ldots, k_{max}]$
2: Choose the number of butterflies ν
3: Initialize the sensory modality μ, the power exponent e, and the probability of switching p
4: **for** the number of intervals k $(k = k_{min}, k_{min+1}, \ldots, k_{max})$ **do**
5: **for** each butterfly $bf_i (i = 1, 2, \ldots, \nu)$ **do**
6: Initialize the interval length δ_i as per Eq. 7
7: Calculate the mid-value $c_i^j (j = 1, 2, \ldots, k)$ of each of the interval as per Eq. 12
8: **end for**
9: **while** the sopping condition is not encountered **do**
10: **for** each butterfly $bf_i (i = 1, 2, \ldots, \nu)$ **do**
11: Divide the UOD into k numbers of intervals based on the current mid-values

$c_i^j (j = 1, 2, \ldots, k)$
Define linguistic value for each of the intervals
Generate a fuzzy set F by fuzzifying each of the observations of the input time series data
for each fuzzy order $m (m = 1, 2, \ldots, maxOrder)$ **do**
Generate m-order fuzzy relationships $M - FLRs$ as per step 5 of section 3.3.2
Generate the fuzzy relationship groups $M - FLRGs$ by combining $M - FLRs$ as per the step 6 of Section 3.3.2
Mapping out the $M - FLRGs$ to their corresponding timestamps
Determine the forecast value by using the Eq. 22
Compute the mean absolute error (MAE) (which is considered as the stimulus intensity I)
end for
Update the personal best butterfly bf_{i_best} and the fuzzy order m_{i_best} for the butterfly bf_i whose MAE is lowest (MAE_{i_best})
end for
Update the global best butterfly bf_{gbest}^k and the fuzzy order m_{gbest}^k among all the butterflies $bf_{i\_best} (i = 1, 2, \ldots, \nu)$ for the fuzzy intervals k having lowest MAE_{best}.
if $MAE_{best} \leq p$ **then**
for each butterfly $bf_i (i = 1, 2, \ldots, \nu)$ **do**
Adjust mid-values $c_i^j (j = 1, 2, \ldots, k)$ as per Eq. 25
end for
else
for each butterfly $bf_i (i = 1, 2, \ldots, \nu)$ **do**
Adjust the mid-values $c_i^j (j = 1, 2, \ldots, k)$ as per Eq. 26
end for
end if
end while
Update the global best number of intervals k_{gbest}
end for
Return global best combinations $(k_{gbest}, bf_{gbest}, m_{gbest})$

In the FTS-IBO algorithm, two loops are used. The inner loop is used to optimize the fuzzy order along with the FTS interval length for a given number of intervals mentioned in the outer loop. Whereas the outer loop repeats within a search space of a given number of intervals and determine the optimum values of the FTS parameters among all the iterations. Step 1 of the algorithm initialize the search space of the number of intervals. Step 4 iterates over the search space to find the optimal value of the number of intervals. Step 9 of the algorithm is used to find the optimal value of the interval lengths for the number of intervals k. And these optimum interval lengths are determined by moving the position of each butterfly as per the Eq. 25 and 26. Whereas step 14 finds the optimal value of the fuzzy order for each butterfly. The final result of this algorithm is the optimum value of the number of intervals, interval lengths, and the fuzzy order.

4 Experimental Process

Here all the simulation works are conducted using the Python programming language with Keras deep learning library and pyFTS fuzzy time series library. All the experiments are executed in a computer with Intel Core i3-7100U processor with a clock speed of 2.40 GHz, and 8 GB RAM.

4.1 Experimental Data

Dataset	Parameters	Duration
Air Quality (DS1)	PM2.5($\mu g/m^3$), Humidity(%), Temperature(^oC) Wind direction(^{o}), Wind Speed(m/s)	01-01-2020 01:00 - 08-05-2021 24:00
Smart Home (DS2)	Power consumption(kW), Temperature(^oC) Humidity(%), Pressure(Pa), Wind Speed(m/s)	01-01-2016 01:00 -15-12-2016 21:00

Table 2: Dataset description.

In this experiment, two different MTS datasets are used for the case study of the proposed work. The first dataset (DS1) is an air quality dataset, which is collected from the data collection station Liverpool in Sydney, Australia. Liverpool is located in the urban area of Sydney. Sydney is the capital of the state of New South Wales (NSW), and the data collection stations are controlled and operated by the NSW Department of Planning, Industry, and Environment (DPIE) [NSW, 2021]. The second dataset (DS2) is the "Smart Home Dataset with Weather Information" dataset, which is publicly available on the Kaggle Web site. The detailed description of these two datasets is presented in Table 2.

In this study, DS1 is used to forecast the multi-step-ahead PM2.5($\mu g/m^3$) concentration using meteorological factors. Whereas the DS2 is applied to forecast the electrical power consumption (kW) of a building based on the weather information.

4.2 Performance Evaluation Indicators

In this experiment, two statistical measures are used to evaluate the performance of the proposed work; Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). The MAE is used during the training phase of the work. And during the testing phase of the work, both the MAE and RMSE are used for the performance evaluation. These errors are defined as follows:

$$MAE = \frac{1}{n}\sum_{i=1}^{n} \|y_i - y_i'\|, \tag{27}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (y_i - y_i')^2}, \tag{28}$$

here y_i and y_i' are the actual and the predicted value of the ith timestamp respectively and n represents the number of data samples (timestamp) present in the dataset.

Here to analyze the effectiveness of the proposed work, it is compared with four baseline models. These models are summarized as follows:

MIMO-SVR [Bao et al., 2014]: Support vector regression (SVR) is a popular machine learning model which mostly applied for the time series forecasting problem. In MIMO-SVR the multiple-input multiple-output (MIMO) prediction strategy is applied for multi-step-ahead time series prediction.

DIR-SVR [Bao et al., 2014]: Here the SVR model is applied to forecast the multi-step-ahead time series data by adopting the direct strategy.

DIR-LSTM: The long short-term memory (LSTM) recurrent neural network is a deep learning model that is mostly used for the time series prediction problem. Here the direct strategy is adopted to predict the multi-step-ahead time series data.

DNFPS [Bhanja et al., 2022]: Deep neuro-fuzzy prediction system (DNFPS) is a hybrid deep model that uses both the deep learning model and the FTS model. Here the recursive strategy is applied for the multi-step-ahead time series forecasting.

4.3 Experimental Setup

FTS Method					
Partitions	41	42	43	44	45
MAE	3.796	3.823	3.728	2.821	3.895
Partitions	46	47	48	49	50
MAE	3.755	2.917	3.803	3.826	3.734

Table 3: Prediction errors of FTS method with the increase of number of partition for the dataset DS1.

FTS	**Fuzzy order**	1	2	3	4	5	6	7
	MAE	3.734	2.242	1.409	1.117	0.891	0.786	0.713

Table 4: Prediction error of FTS for the varying number of fuzzy orders for the dataset DS1.

The success of any FTS forecasting method largely depends on the value of its hyperparameters, such as - the number of fuzzy intervals (k), interval length (δ), and fuzzy order. So, to obtain the best performance, the value of these hyperparameters is optimized. In this experiment, we have simulated the FTS forecasting method with various numbers of intervals, different interval lengths, and various fuzzy orders. In Table 3, we have represents the performance of the FTS forecasting method for the dataset DS1 with the number of intervals in UOD ranging from 41 to 50 for the first order fuzzy. The table shows that the performance of the FTS forecasting method fluctuates

with an increase in the number of intervals. So to obtain the best performance, the optimal number of interval determination is highly important. Here the BO algorithm is applied to determine the optimal value of the number of intervals in the UOD.

In the next phase of the experiment, the importance of the fuzzy order for the FTS forecasting method is determined. Here the FTS forecasting method is simulated with the fuzzy order ranging from 1 to 7 for the different number of FTS intervals. The range of the fuzzy orders is obtained as per the experiment conducted in [Chen, 2002]. And the prediction error (Mean Absolute Error (MAE)) for the dataset DS1 is show in Table 4 for the number of FTS intervals 44. The table shows that the prediction error decreases with an increase in the fuzzy order. This prediction result is quite identical to the findings of the work [Kumar and Susan, 2021].

Method	**Parameter**	**Value**
FTS	DS1: No. of intervals	{25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80}
	DS2: No. of intervals	{15, 20, 25, 30, 35, 40, 45, 50, 55, 60}
	Fuzzy order	{1, 2, 3, 4, 5, 6, 7}
BO	No. of Butterflies	20
	Maximum no. of iteration	50
	μ, e, p	0.1, 0.7, Lowest MAE of the 1st iteration

Table 5: The value of the parameters used in this experiment.

The realization of the optimum value of all the hyperparameters of the FTS forecasting method is a challenging and also time-consuming task. Here an integrated BO (FTS-IBO) algorithm is proposed to find all the hyperparameters of the FTS forecasting method. Here we also simulate the FTS-BO algorithm [Bhanja et al., 2022] by using a nested loop to optimize the hyperparameters. In this work, the experiment is conducted with a varying number of partitions of UOD ranging from 25 to 80 for the dataset DS1 and 15 to 60 for the dataset DS2. This work is also simulated with a varying number of fuzzy orders ranging from 1 to 7 for both datasets. The value of the FTS parameters is presented in Table 5. Here in FTS, the number of intervals is selected in such a way that the highest number of intervals will be less than half of the data samples present in UOD. The BO parameters (μ, e, and p) used in this experiment are also presented in Table 5. μ is called the sensory modality, and it controls the convergence speed of the algorithm. e is the power exponent, it determines the degree of absorption. p is the switching probability that controls the switching between the global and local search. The optimal values of these parameters are realized from the literature survey and as per the experiment conducted in [Bhanja et al., 2022]. Table 6 represents the performance of both the FTS-IBO and FTS-BO algorithms.

Here, the performance of both the FTS-BO and FTS-IBO algorithms are analyzed as per the specifications mentioned in Table 5, and the comparison results are presented in Table 6. From this table, it can be observed that the execution time of the FTS-IBO algorithm is very low compared to the FTS-BO algorithm for both datasets. Although

both the algorithms take the same execution time for a single iteration, the execution time of the FTS-IBO algorithm is low due to the lesser number of iterations in the inner loop. In the next section, we analyzed the performance of the proposed multi-step-ahead foresting method.

Dataset	**Method**	**Outer loop**	**Inner loop**	**Execution Time**
DS1	Nested FTS-BO	12	37	$12 \times 37 \times 8.16ms = 3.62s$
	FTS-IBO	12	19	$12 \times 19 \times 8.16ms = 1.86s$
DS2	Nested FTS-BO	10	31	$10 \times 31 \times 8.73ms = 2.71s$
	FTS-IBO	10	14	$10 \times 14 \times 8.73ms = 1.22s$

Table 6: Run time analysis of the Nested FTS-BO and FTS-IBO algorithms.

4.4 Results & Discussion

The performance comparison results for the dataset DS1 are presented in Table 7 and 8. Table 7 exhibits the MAE for 1 to 24 hours-ahead predictions, whereas the Table 8 show the RMSE for the same forecasting horizon. From these tables, it can be seen that the proposed method produces the lowest MAEs and RMSEs for all the forecasting horizons compared to the other models. The DNFPS produces the 2nd best result for the 1 to 2 hours-ahead forecasting, but when the forecasting horizon is greater than 2 hours its performance is worst. In Figure 3, we represent the average forecasting errors (MAE and RMSE) for the consecutive 6 hours ahead predictions for the dataset DS1. This figure also confirms that the proposed method yields the best result compared to others.

Forecasting horizon (h)	Method				
	DNFPS	Proposed	DIR-SVR	MIMO-SVR	DIR-LSTM
	MAE				
1	0.493	0.364	0.613	0.681	0.525
2	0.563	0.499	0.791	0.893	0.689
3	0.721	0.577	0.685	0.967	0.747
4	1.275	0.716	1.334	1.890	0.908
5	2.178	0.793	1.432	1.924	0.992
6	2.569	0.922	1.747	1.995	1.442
7	2.673	0.933	1.802	2.017	1.658
8	2.698	1.061	2.073	2.151	1.833
9	3.027	1.121	2.091	2.219	2.214
10	4.339	1.243	3.684	2.230	3.123
11	5.781	1.285	3.767	2.520	3.297
12	6.707	1.267	4.259	2.590	3.846
13	7.538	1.266	4.281	2.610	3.916
14	8.703	1.292	4.567	2.690	4.035
15	8.873	1.356	4.812	2.740	4.157
16	8.876	1.487	4.933	2.990	4.211
17	8.919	1.564	5.106	3.130	4.235
18	9.168	1.813	5.332	3.370	4.549
19	9.823	1.874	5.481	3.340	4.582
20	10.374	2.069	5.952	3.390	4.656
21	10.203	2.493	5.989	3.410	4.785
22	10.612	2.531	6.079	3.460	4.838
23	11.213	2.669	6.343	3.530	5.019
24	11.018	2.789	6.645	3.560	5.085

Table 7: MAE of various prediction models with varying number of forecasting horizons for the dataset DS1.

Forecasting horizon (h)	Method				
	DNFPS	Proposed	DIR-SVR	MIMO-SVR	DIR-LSTM
	RMSE				
1	0.519	0.416	1.059	1.209	0.807
2	0.683	0.621	1.308	1.417	0.915
3	1.305	0.893	1.119	1.598	1.116
4	1.974	1.261	2.293	2.713	1.588
5	3.076	1.423	2.375	2.903	1.653
6	3.519	1.706	2.938	3.175	2.506
7	3.751	1.837	3.154	3.366	2.813
8	3.942	1.805	3.473	3.519	3.119
9	4.701	1.964	3.518	3.821	3.717
10	6.538	2.007	5.068	3.893	4.557
11	7.915	2.041	4.937	4.017	4.652
12	8.939	2.145	5.809	4.136	5.113
13	9.861	2.291	5.881	4.355	5.407
14	11.807	2.219	6.008	4.507	5.792
15	11.951	2.309	6.254	4.299	5.822
16	12.153	2.397	6.467	4.716	5.907
17	12.973	2.604	7.112	4.918	5.925
18	13.109	2.713	7.409	5.009	6.318
19	12.227	2.990	7.551	5.012	6.461
20	14.318	2.975	8.105	5.117	6.588
21	14.273	3.021	8.116	5.283	6.673
22	15.934	3.442	8.529	5.406	6.880
23	16.883	3.725	8.907	5.611	7.011
24	16.509	3.841	9.411	5.588	7.192

Table 8: RMSE of various prediction models with varying number of forecasting horizons for the dataset DS1.

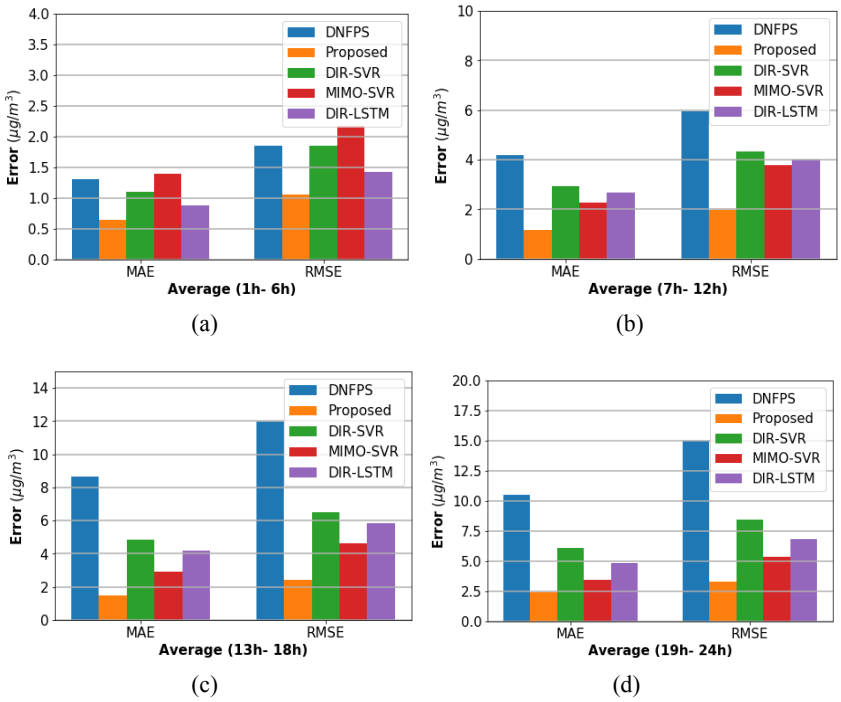

Figure 3: Average forecasting errors of the dataset DS1 of the different models for different number of forecasting horizons (a) 1h - 6h, (b) 7h - 12h, (c) 13h - 18h, and (d) 19h - 24h.

Table 9 and 10 illustrates the performance comparison in terms of MAE and RMSE for the dataset DS2 respectively. These two tables also exhibit that the performance of the proposed method is better than the other models, and the model DNFPS produces the worst result when the forecasting horizon is greater than 1 hour. The average forecasting errors (MAE and RMSE) for forecasting horizon 1-6 hours, 7-12 hours, 13-18 hours, and 19-24 hours is presented in Figure 4 for the dataset DS2. Here also it can be seen that for all the average forecasting horizons the proposed method produces the lowest errors.

Figure 5 represents 1 to 24 hours (04-05-2021 01:00 - 24:00 hours) ahead forecasting results for for all the models for the dataset DS1. Whereas Figure 6 shows the forecasting results for 1 to 24 hours (15-12-2016 01:00 - 24:00 hours) for the dataset DS2. From these two figures, we can see that all the forecasting horizons, proposed model produce the best forecasting results.

Forecasting horizon (h)	Method				
	DNFPS	Proposed	DIR-SVR	MIMO-SVR	DIR-LSTM
	MAE				
1	0.710	0.495	1.106	1.234	0.953
2	1.085	0.525	1.137	1.420	1.052
3	0.587	0.471	0.863	1.123	0.832
4	1.622	0.797	1.046	1.462	0.949
5	3.725	0.854	2.073	1.629	1.415
6	5.031	0.972	2.004	2.019	1.969
7	5.475	1.023	2.560	2.646	2.492
8	7.112	1.208	3.699	2.959	3.099
9	8.718	0.978	3.094	2.168	2.594
10	6.353	0.716	3.024	1.759	2.403
11	8.462	1.547	4.069	2.068	3.558
12	7.899	1.583	4.528	2.300	3.828
13	7.096	1.245	3.048	1.793	2.944
14	8.436	1.717	3.831	1.835	3.347
15	5.659	1.315	3.607	1.551	2.713
16	7.646	1.497	4.208	2.434	3.737
17	10.894	3.240	5.428	3.766	4.842
18	10.995	3.506	5.507	3.691	4.859
19	9.188	2.773	5.379	3.504	4.812
20	11.442	3.969	6.144	4.534	5.412
21	11.502	3.499	6.039	4.517	5.355
22	10.912	3.407	5.898	4.301	5.090
23	12.304	4.348	8.569	5.086	7.969
24	11.919	4.155	7.847	5.076	7.463

Table 9: MAE of various prediction models with varying number of forecasting horizons for the dataset DS2.

Forecasting **horizon** (h)	**Method**				
	DNFPS	**Proposed**	**DIR-SVR**	**MIMO-SVR**	**DIR-LSTM**
	RMSE				
1	1.382	0.835	2.351	2.563	1.807
2	2.109	0.938	2.364	2.609	2.007
3	0/.973	0.822	1.539	2.314	1.507
4	2.975	1.357	2.138	2.709	1.871
5	5.403	1.516	3.825	2.953	2.708
6	7.304	1.861	3.251	3.296	3.107
7	7.806	2.244	3.901	4.115	3.813
8	10.005	2.517	5.223	4.531	4.629
9	10.413	1.907	5.002	3.791	4.106
10	8.915	1.352	4.910	2.911	4.027
11	10.831	2.805	5.881	3.796	5.563
12	10.214	2.913	6.018	3.991	5.694
13	9.716	2.628	5.219	2.976	4.863
14	10.791	3.108	5.714	3.035	5.107
15	7.813	2.517	5.439	2.811	4.532
16	9.817	2.682	5.991	3.417	5.883
17	16.112	5.094	7.115	5.019	6.981
18	16.308	5.417	7.218	5.012	6.995
19	14.743	4.576	6.868	4.892	6.874
20	17.105	5.117	7.852	6.513	7.351
21	17.311	4.956	7.715	6.488	7.219
22	16.882	4.877	7.331	6.284	6.917
23	18.105	6.117	10.882	7.359	10.358
24	17.627	5.891	9.974	7.108	9.443

Table 10: RMSE of various prediction models with varying number of forecasting horizons for the dataset DS2.

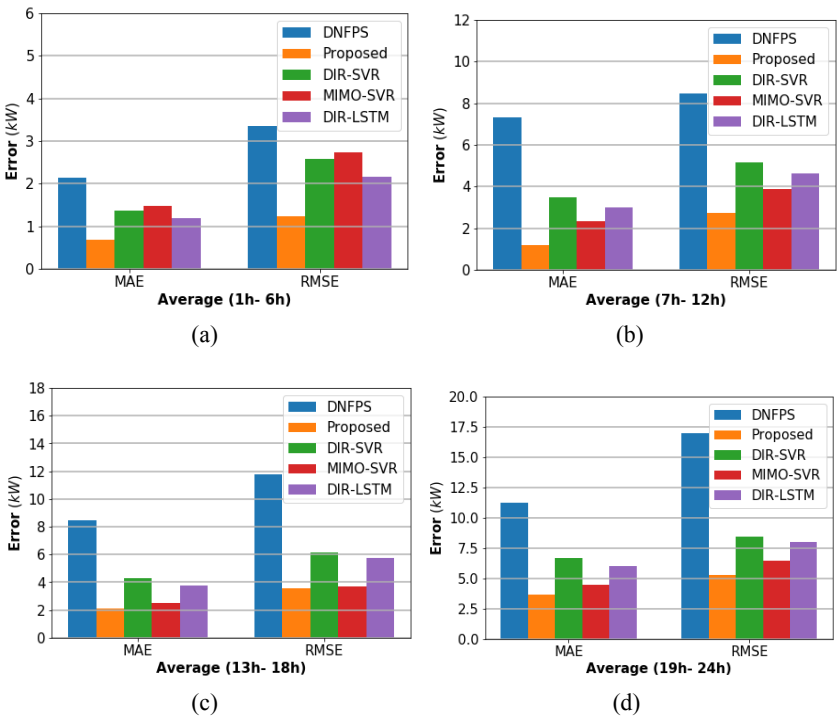

Figure 4: Average forecasting errors of the dataset DS2 of the different models for different number of forecasting horizons (a) 1h - 6h, (b) 7h - 12h, (c) 13h - 18h, and (d) 19h - 24h.

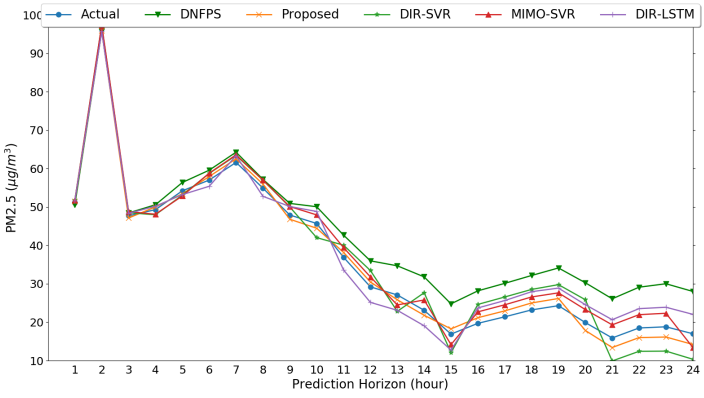

Figure 5: 1 to 24 hours ahead forecasting results of different models of the dataset DS1.

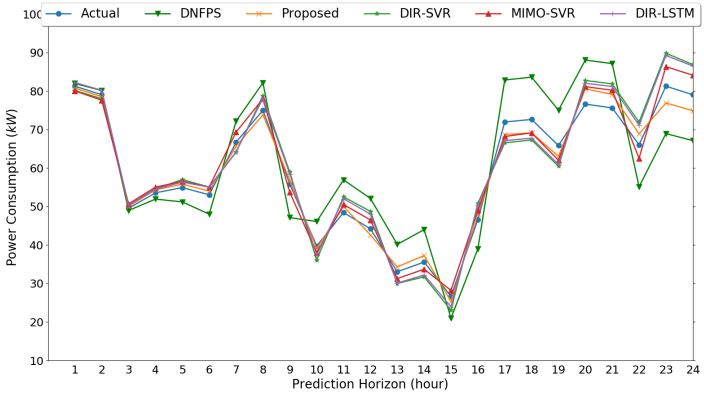

Figure 6: 1 to 24 hours ahead forecasting results of different models of the dataset DS2.

From all the experimental results the following observations are drawn:

- Our proposed method produces better results across two datasets.
- DNFPS is good enough for single or two-step-ahead predictions, but worst for multi-step-ahead (> 2) predictions.
- Among the two SVR models (MIMO-SVR and DIR-SVR), the MIMO-SVR model produces a better result.
- The performance of the DIR-LSTM model is superior compared to both the SVR models.

5 Conclusion

In this study, a multi-step-ahead TS forecasting method is proposed to forecast TS data more accurately for extended future horizons. As part of the proposed forecasting method, we build a deep learning-based prediction model to forecast the TS data. Then a residual error correction model is proposed to correct the forecasting error. This error correction model is created based on the FTS forecasting method. Here we also proposed an FTS-IBO algorithm to optimize all the hyperparameters of FTS. To analyze the effectiveness of the proposed forecasting method, its performance is compared with the other benchmark methods. From the simulation results, the following conclusions are drawn:

- The proposed FTS-IBO algorithm is able to optimize all the FTS hyperparameters in an integrated way, and it is computationally faster than the FTS-BO algorithm.
- Inclusion of an optimization algorithm with FTS has significantly improved the forecasting accuracy.
- The performance of the proposed forecasting method is superior compared to the other benchmark methods.

– The proposed method can be applied to forecast both the single-step and multi-step-ahead TS forecasting problems.

In this experiment, the proposed method is tested for normal circumstances, where there is a steady movement of the time series data. However, there are a variety of reasons for which the time series data can suddenly fall or rise. In the future, we like to extend this work to deal with this issue.

References

[Abdulhussein et al., 2021] Abdulhussein, K. G., Yasin, N. M., and Hasan, I. J. (2021). Comparison between butterfly optimization algorithm and particle swarm optimization for tuning cascade pid control system of pmdc motor. *Int J Pow Elec & Dri Syst ISSN*, 2088(8694):8694.

[Abhishekh et al., 2018] Abhishekh, Gautam, S. S., and Singh, S. (2018). A refined weighted method for forecasting based on type 2 fuzzy time series. *International Journal of Modelling and Simulation*, 38(3):180–188.

[Ali et al., 2021] Ali, M. M., Paul, B. K., Ahmed, K., Bui, F. M., Quinn, J. M., and Moni, M. A. (2021). Heart disease prediction using supervised machine learning algorithms: Performance analysis and comparison. *Computers in Biology and Medicine*, 136:104672.

[Arora and Singh, 2017] Arora, S. and Singh, S. (2017). Node localization in wireless sensor networks using butterfly optimization algorithm. *Arabian Journal for Science and Engineering*, 42:3325–3335.

[Arora and Singh, 2019] Arora, S. and Singh, S. (2019). Butterfly optimization algorithm: a novel approach for global optimization. *Soft Computing*, 23(3):715–734.

[Bajestani and Zare, 2011] Bajestani, N. S. and Zare, A. (2011). Forecasting taiex using improved type 2 fuzzy time series. *Expert Systems with Applications*, 38(5):5816–5821.

[Bao et al., 2014] Bao, Y., Xiong, T., and Hu, Z. (2014). Multi-step-ahead time series prediction using multiple-output support vector regression. *Neurocomputing*, 129:482–493.

[Bhanja and Das, 2021] Bhanja, S. and Das, A. (2021). A hybrid deep learning model for air quality time series prediction. *Indonesian Journal of Electrical Engineering and Computer Science*, 22(3):1611–1618.

[Bhanja and Das, 2022a] Bhanja, S. and Das, A. (2022a). A black swan event-based hybrid model for indian stock markets' trends prediction. *Innovations in Systems and Software Engineering*, pages 1–15.

[Bhanja and Das, 2022b] Bhanja, S. and Das, A. (2022b). A black swan event-based hybrid model for indian stock markets' trends prediction. *Innovations in Systems and Software Engineering*, pages 1–15.

[Bhanja et al., 2022] Bhanja, S., Metia, S., and Das, A. (2022). A hybrid neuro-fuzzy prediction system with butterfly optimization algorithm for pm2. 5 forecasting. *Microsystem Technologies*, pages 1–16.

[Bose and Mali, 2019] Bose, M. and Mali, K. (2019). Designing fuzzy time series forecasting models: A survey. *International Journal of Approximate Reasoning*, 111:78–99.

[Chen, 2002] Chen, S.-M. (2002). Forecasting enrollments based on high-order fuzzy time series. *Cybernetics and Systems*, 33(1):1–16.

[Chevillon, 2007] Chevillon, G. (2007). Direct multi-step estimation and forecasting. *Journal of Economic Surveys*, 21(4):746–785.

[Cox, 1961] Cox, D. R. (1961). Prediction by exponentially weighted moving averages and related methods. *Journal of the Royal Statistical Society: Series B (Methodological)*, 23(2):414–422.

[Díaz-Robles et al., 2008] Díaz-Robles, L. A., Ortega, J. C., Fu, J. S., Reed, G. D., Chow, J. C., Watson, J. G., and Moncada-Herrera, J. A. (2008). A hybrid arima and artificial neural networks model to forecast particulate matter in urban areas: The case of temuco, chile. *Atmospheric Environment*, 42(35):8331–8340.

[Ding et al., 2019] Ding, M., Zhou, H., Xie, H., Wu, M., Nakanishi, Y., and Yokoyama, R. (2019). A gated recurrent unit neural networks based wind speed error correction model for short-term wind power forecasting. *Neurocomputing*, 365:54–61.

[Dong et al., 2009] Dong, M., Yang, D., Kuang, Y., He, D., Erdal, S., and Kenski, D. (2009). Pm2. 5 concentration prediction using hidden semi-markov model-based times series data mining. *Expert Systems with Applications*, 36(5):9046–9055.

[Du et al., 2019a] Du, S., Li, T., Yang, Y., and Horng, S.-J. (2019a). Deep air quality forecasting using hybrid deep learning framework. *IEEE Transactions on Knowledge and Data Engineering*, 33(6):2412–2424.

[Du et al., 2019b] Du, S., Li, T., Yang, Y., and Horng, S.-J. (2019b). Deep air quality forecasting using hybrid deep learning framework. *IEEE Transactions on Knowledge and Data Engineering*, 33(6):2412–2424.

[Du et al., 2019c] Du, S., Li, T., Yang, Y., and Horng, S.-J. (2019c). Deep air quality forecasting using hybrid deep learning framework. *IEEE Transactions on Knowledge and Data Engineering*, 33(6):2412–2424.

[Fathi et al., 2021] Fathi, M., Haghi Kashani, M., Jameii, S. M., and Mahdipour, E. (2021). Big data analytics in weather forecasting: A systematic review. *Archives of Computational Methods in Engineering*, pages 1–29.

[Franses and Legerstee, 2010] Franses, P. H. and Legerstee, R. (2010). A unifying view on multi-step forecasting using an autoregression. *Journal of Economic Surveys*, 24(3):389–401.

[Ghose et al., 2022] Ghose, B., Rehena, Z., and Anthopoulos, L. (2022). A deep learning based air quality prediction technique using influencing pollutants of neighboring locations in smart city. *JUCS: Journal of Universal Computer Science*, 28(8).

[Jiang et al., 2021] Jiang, F., Zhang, C., Sun, S., and Sun, J. (2021). Forecasting hourly pm2.5 based on deep temporal convolutional neural network and decomposition method. *Applied Soft Computing*, 113:107988.

[Ju et al., 2018] Ju, C., Bibaut, A., and van der Laan, M. (2018). The relative performance of ensemble methods with deep convolutional neural networks for image classification. *Journal of Applied Statistics*, 45(15):2800–2818.

[Kim et al., 2022] Kim, J.-M., Cho, C., and Jun, C. (2022). Forecasting the price of the cryptocurrency using linear and nonlinear error correction model. *Journal of Risk and Financial Management*, 15(2):74.

[Kumar and Susan, 2021] Kumar, N. and Susan, S. (2021). Particle swarm optimization of partitions and fuzzy order for fuzzy time series forecasting of covid-19. *Applied Soft Computing*, 110:107611.

[Li et al., 2022] Li, Y., Yu, X., and Liu, J. (2022). Enhanced butterfly optimization algorithm for large-scale optimization problems. *Journal of Bionic Engineering*, 19(2):554–570.

[NSW, 2021] NSW (2021). Dataset: New south wales. https://www.dpie.nsw.gov.au/air-quality/air-quality-data-services/data-download-facility.

[Pei et al., 2022] Pei, J., Zhong, K., Jan, M. A., and Li, J. (2022). Personalized federated learning framework for network traffic anomaly detection. *Computer Networks*, 209:108906.

[Peng et al., 2022] Peng, L., Wang, L., Xia, D., and Gao, Q. (2022). Effective energy consumption forecasting using empirical wavelet transform and long short-term memory. *Energy*, 238:121756.

[Reyes and Ventura, 2019] Reyes, O. and Ventura, S. (2019). Performing multi-target regression via a parameter sharing-based deep network. *International journal of neural systems*, 29(09):1950014.

[Sezer et al., 2020] Sezer, O. B., Gudelek, M. U., and Ozbayoglu, A. M. (2020). Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. *Applied soft computing*, 90:106181.

[Singh and Borah, 2014] Singh, P. and Borah, B. (2014). Forecasting stock index price based on m-factors fuzzy time series and particle swarm optimization. *International Journal of Approximate Reasoning*, 55(3):812–833.

[Song and Chissom, 1993a] Song, Q. and Chissom, B. S. (1993a). Forecasting enrollments with fuzzy time series—part i. *Fuzzy sets and systems*, 54(1):1–9.

[Song and Chissom, 1993b] Song, Q. and Chissom, B. S. (1993b). Fuzzy time series and its models. *Fuzzy sets and systems*, 54(3):269–277.

[Van Tinh, 2020] Van Tinh, N. (2020). Forecasting of covid-19 confirmed cases in vietnam using fuzzy time series model combined with particle swarm optimization. *Comput Res Prog Appl Sci Eng*, 6(2):114–120.

[Wang et al., 2016] Wang, J., Song, Y., Liu, F., and Hou, R. (2016). Analysis and application of forecasting models in wind power integration: A review of multi-step-ahead wind speed forecasting models. *Renewable and Sustainable Energy Reviews*, 60:960–981.

[Weigend, 2018] Weigend, A. S. (2018). Time series prediction: forecasting the future and understanding the past.

[Xu et al., 2021] Xu, W., Liu, P., Cheng, L., Zhou, Y., Xia, Q., Gong, Y., and Liu, Y. (2021). Multi-step wind speed prediction by combining a wrf simulation and an error correction strategy. *Renewable Energy*, 163:772–782.

[Zadeh, 1973] Zadeh, L. A. (1973). Outline of a new approach to the analysis of complex systems and decision processes. *IEEE Transactions on systems, Man, and Cybernetics*, pages 28–44.

[Zhang et al., 2021a] Zhang, M., Li, X., and Wang, R. (2021a). Incipient fault diagnosis of batch process based on deep time series feature extraction. *Arabian Journal for Science and Engineering*, 46(10):10125–10136.

[Zhang et al., 2021b] Zhang, Z., Zeng, Y., and Yan, K. (2021b). A hybrid deep learning technology for pm2. 5 air quality forecasting. *Environmental Science and Pollution Research*, 28(29):39409–39422.

[Zhong et al., 2019] Zhong, L., Hu, L., and Zhou, H. (2019). Deep learning based multi-temporal crop classification. *Remote sensing of environment*, 221:430–443.

[Zhou et al., 2014] Zhou, Q., Jiang, H., Wang, J., and Zhou, J. (2014). A hybrid model for pm2. 5 forecasting based on ensemble empirical mode decomposition and a general regression neural network. *Science of the Total Environment*, 496:264–274.