GATE: A Guided Approach for Time Series Ensemble Forecasting

Md Rasel Sarkar[a] (raselbdeee@gmail.com), Sreenatha G. Anavatti[a] (agsrenat@adfa.edu.au), Tanmoy Dam[b] (tanmoydam@yahoo.com), Md Meftahul Ferdaus[c] (mferdaus@uno.edu), Murat Tahtali[a] (m.tahtali@adfa.edu.au), Savitha Ramasamy[d] (ramasamy.savitha@gmail.com), Mahardhika Pratama[e] (Dhika.Pratama@unisa.edu.au)

[a] School of Engineering & Information Technology, The University of New South Wales, Canberra, ACT 2610, Australia

[b] Saab-NTU Joint Lab, Nanyang Technological University, 637460, Singapore

[c] Department of computer science, University of New Orleans, 70148, USA

[d] I2R, A*STAR, 1 Fusionopolis Way, 21-01 Connexis, 138632, Singapore

[e] STEM, University of South Australia, Adelaide 5095, Australia

Corresponding Author:

Sarkar

School of Engineering Information Technology, The University of New South Wales, Canberra, ACT 2610, Australia

raselbdeee@gmail.com

GATE: A Guided Approach for Time Series Ensemble Forecasting

Md Rasel Sarkar[a,*], Sreenatha G. Anavatti[a], Tanmoy Dam[b], Md Meftahul Ferdaus[c], Murat Tahtali[a], Savitha Ramasamy[d], Mahardhika Pratama[e]

[a]*School of Engineering & Information Technology, The University of New South Wales, Canberra, ACT 2610, Australia*
[b]*Saab-NTU Joint Lab, Nanyang Technological University, 637460, Singapore*
[c]*Department of Computer Science, University of New Orleans, 70148, USA*
[d]*I2R, A*STAR, 1 Fusionopolis Way, 21-01 Connexis, 138632, Singapore*
[e]*STEM, University of South Australia, Adelaide 5095, Australia*

Abstract

In this article, a new ensemble learning model called GATE is proposed to improve the accuracy and stability of time-series forecasting, which is a crucial aspect of modern engineering practices. Despite the promise of deep learning (DL) models in this area, their performance can be volatile due to the diversity of time series data. To address this, the GATE model combines the strengths of recurrent neural networks (RNN), long short-term memory network (LSTM), and convolution-LSTM (Conv-LSTM) structures and utilizes an unsupervised learning strategy to steer the ensemble output using a guided network. To prevent overfitting in DL models, GATE optimizes the sample loss function and the weight updating function for each individual model within the ensemble structure. A comprehensive evaluation of the proposed GATE method on four real-world datasets is presented. The experimental results unequivocally demonstrate that GATE surpasses state-of-the-art ensemble methods and individual models, exhibiting the best performance in terms of testing errors. Notably, GATE outperforms existing single models in addressing long-term prediction tasks. To validate the effectiveness of GATE, ablation studies are carried out, comparing different ensemble combinations involving two distinct

*Corresponding author.
Email address: raselbdeee@gmail.com (Md Rasel Sarkar)

models. Through systematic analysis, these studies provided valuable insights into the performance of various ensemble configurations, further confirming the effectiveness and superiority of the proposed GATE method.
Keywords: Time Series Forecasting, RNN, LSTM, Conv-LSTM, Guided Network and, Ensemble Model.

---

1. Introduction

Time series forecasting (TSF) is a method of making predictions about future events using a collection of past observational data that has been chronologically organized. Time series forecasting has become an increasingly important topic of interest due to its potential to provide valuable insight into future trends and patterns in various fields, including finance, economics, marketing, and meteorology Gajamannage et al. (2023); Sarkar et al. (2023). With the help of advanced machine learning algorithms Mancuso et al. (2021) and statistical techniques, time series forecasting can analyze and interpret large amounts of historical data to make accurate predictions about future trends Zhang et al. (2021). A deterministic model that accurately predicts future patterns can be extremely valuable Fang & Yuan (2019). Having a stable and accurate TSF is crucial for many applications Panigrahi & Behera (2017). For instance, wind speed forecasting holds the promise of enhancing the efficiency of wind farms, while air pollution forecasting aids in mitigating health risks associated with polluted air. Predicting power demand facilitates the coordination between electricity supply and demand, leading to economic optimization Zhang et al. (2021). With accurate temperature predictions, individuals, businesses, and governments can make informed decisions and take appropriate measures to manage risks and optimize their operations de Mattos Neto et al. (2022).

Traditional methodologies such as the often deployed autoregressive moving average (ARIMA) framework, endorsed by Whittle's research Whittle (1952), alongside its numerous adaptations, have indeed showcased their prowess in modeling protracted linear correlations Makridakis & Hibon (1997). Alas, with

the cascade of voluminous data in recent times, our predictive models crave refinements in their structures to effectively grapple with the extensive non-linear intricacies woven into time series. To address this issue, several nonlinear auto-regressive models have been introduced. However, these models suffer from the limitation of using predefined non-linear mapping functions, which may not accurately capture the true underlying non-linear relationships Yan et al. (2013).

Artificial neural networks (ANNs) have the ability to numerically approximate non-linear, continuous relationships. An embryonic blueprint to encapsulate the nonlinear intricacies inherent in time series data emerged courtesy of Faruk (2010), utilizing the firepower of DL architectures such as the multi-layer perceptron (MLP) neural network (NN), and the feed-forward neural network (FFNN). This strategy presupposed the existence of both linear and nonlinear associations within the data and thus meticulously tailored the nonlinear associations based on the residuals emanating from the chronological data. Despite the initial strides, more refined methodologies leverage RNNs, designed to mold sequential data without bifurcating the duo of dependencies Rumelhart et al. (1986) Elman (1991) Khaldi et al. (2023). The process of enhancing RNNs via gradient descent algorithms typically warrants meticulously curated learning paradigms to circumvent the hurdles of the disappearance of gradients Bengio et al. (1994). Illustrative triumphs have used an advanced variant of RNN, referred to as LSTM networks, for the prognostication of single-step time series Marino et al. (2016); Cirstea et al. (2018). Nevertheless, high variances and low biases are both typical characteristics of DL models. It can be challenging for a single DL model to maintain high levels of accurate forecasting and robustness in an application setting that is both complex and dynamic application environment. The solution to this issue, however, is ensemble learning—an amalgamation of a variety of discrete singular models fused with specialized ensemble stratagems to bolster the overarching model's aptitude for generalization.

With the growing popularity of DL, there has been significant interest in developing new models in the forecasting of time series. However, despite the apparent differences between these models, a closer examination reveals that

they often rely on the same underlying network units, differing primarily in their hyper-parameters. This scenario owes its existence to the restricted assortment of DL models at our disposal. The intention of the study revolves around scrutinizing the efficacy of various DL models engaged in tasks related to time series forecasting and presenting innovative techniques to increase their performance. This paper presents two contributions to enhance the performance of forecasting models. The first contribution is an ensemble model that combines various DL models, including RNN, LSTM, and Conv-LSTM, to improve forecasting tasks. The second contribution is a guided network that incorporates a learning strategy based on the one-hot-code approach. This guided network is used to steer the ensemble output and identify the optimal models during the training process, enhancing the stability and performance of all forecasting models. However, the selected model output, determined during the inference stage, originates from the guided model. This selection process facilitated by the guided network assists in mitigating the problem of overfitting associated with individual model performances. By leveraging the guided network, GATE ensures that the most suitable model output is chosen, enhancing the overall performance and reducing potential overfitting issues. The important contributions of this paper are listed below.

- Presenting the GATE ensemble time series prediction model - a sophisticated combination of RNN, LSTM, and Conv-LSTM structures that harness the strengths of DL and ensemble learning. The model adeptly extracts implicit features from time series data, enhancing its ability to generalize and remain robust while minimizing the risk of overfitting.
- We propose a new unsupervised learning strategy to prevent overfitting when selecting the best models from ensemble structures. Our approach involves using a guided network to steer the ensemble output and identify the optimal models during the training process. This results in an end-to-end learning approach that effectively selects the best models.
- By utilizing the error metric (either MSE or MAE), GATE optimizes

the sample loss function and weight updating function for each individual model within the ensemble structure. This approach enhances the learning of feature information from diverse sample distributions, even when faced with limited sample sizes. As a result, each DL model in the ensemble can effectively capture the intrinsic characteristics of the time series data, leading to improved performance.

- We validate our proposed GATE method on four distinctive real-world datasets, including wind speed, temperature, and power demand dataset. The empirical findings underscore the supremacy of GATE, outclassing the state-of-the-art ensemble methodologies as well as single models, reigning supreme in mitigating testing errors.

The remaining segments of this manuscript unfold in the following sequence. Section 2 dives into a review of previous research that shares a close affinity. The conundrum described in the proposed technique is elaborated in Section 3. Section 4 provides details of the methodologies used. Section 5 deals with the experiments and the ensuing analysis. The concluding Section 6 imparts the inferences drawn from the study.

2. Literature Review

TSF in real-world scenarios is often a complex and demanding task, particularly when it requires prognostication over several temporal steps. The forecast spanning multiple steps forward is universally acclaimed as a prominent unresolved conundrum in the realm of time-series dissection, a complexity magnified by the intrinsic structure of time-series data, characterized by its fickleness and escalated oscillations, primarily fueled by a multitude of influential factors. This makes it difficult to create reliable forecasts. There has been a renewed interest within the community in utilizing DL techniques to create more effective and resilient time-series models. This article will focus on highlighting several noteworthy studies in the field of time series forecasting.

Chen et al. (2018) unveiled a non-linear ensemble deep learning blueprint for time series forecasting, thereby augmenting the diversity of foundational predictors through the fabrication of a cohort of LSTM networks, each embellished with variegated hidden layers and neurons. The crux of such explorations is primarily rooted in cultivating unique network structures or network parameters to amplify the diversity and precision of core predictors. An ensemble model for wind power forecasting was conceived by Wang et al. (2019), amalgamating Wavelet with Echo State Neural Networks (WESNN) and applying the model to authentic data from a Belgian wind farm. The consequent mean absolute percentage error (MAPE) wavered between 1.2% and 2.1%, with the ensemble learning model outstripping both support vector regression (SVR) and a multilayer perceptron neural network, trained with error backpropagation (BPNN) in terms of accuracy—evaluated using mean absolute error (MAE), root mean square error (RMSE), and MAPE. The architects suggested that their methodology harbors significant prospects for practical application in actual electrical power and energy systems, including preventative maintenance.

The landscape of wind power forecasting for future steps 5, 10, and 15 was extended by Ding et al. (2021), developing an ensemble learning model. Their proposition surpassed empirical mode decomposition (EMD) and wavelet transform (WT)-based models employing direct-recursive forecasting strategies in terms of RMSE and MAE. However, the methodology has a downside: it requires additional information for medium and long-term forecasts. Livieris & Pintelas (2022) introduced a unique technique for forecasting residential power consumption, deploying a hybrid univariate deep learning neural network framework, which merges convolutional layers with LSTM layers to exploit the merits of the 1D convolution operation and the recurrent neural network philosophy Goodfellow et al. (2016). Experiments carried out on high-frequency power consumption data from five London households indicated the superiority of their proposed model over traditional time series models, including the support vector machine (SVM), ARIMA, LSTM, and a convolution-based model.

An in-depth study by Pintelas et al. (2020) was carried out to assess the per-

formance of several robust DL technique for time-series forecasting. They also delineated the constraints of classic regression metrics to gauge the reliability of such models. Although these metrics might hint at the proximity between predicted and actual values, they do not ensure the model's capacity to generate reliable predictions. The authors further cautioned against exclusive reliance on these metrics, given their potential to misguide. In response to these issues, Livieris et al. (2020) conducted a comprehensive analysis and identified that deep neural networks could sometimes produce unreliable forecasts, mainly due to nonstationary data. To remedy this, they proposed an intelligent time-series preprocessing framework to grapple with non-stationary characteristics and to cultivate a suitable training time-series dataset for deep learning models. Their experimental analysis evaluated the prediction prowess, the prediction of directional movements, and the prediction reliability of their proposed framework, with the results providing substantial evidence of the potency and efficiency of their approach.

In the field of load prediction, the efficacy of LSTM-based models was championed by Kong et al. (2017), positing that their performance could be enhanced by incorporating additional energy sequences from other measurable appliances. Marino et al. (2016) unveiled that the LSTM-based sequence to sequence (Seq2Seq) architecture trumps the standard LSTM model when predicting residential loads of one minute ahead. For load prediction concerning combined heat and power, a concatenated LSTM, comprising two NN, was developed by Kuan et al. (2017), yielding considerably superior results than using a single LSTM. Convolutional Neural Network (CNN) and RNN were integrated by He (2017) to extract implicit features from historical load series, followed by the use of a dense layer to transform other features. However, Fan et al. (2019) observed that the fusion of one-dimensional convolutional and recurrent operations may not necessarily improve the performance of the energy predictor.

3. Problem Formulation

Improving the precision of multi-step prediction requires addressing overfitting problems through the use of suitable modeling strategies. Let us consider a time series, x, spanning length T, and can be depicted as $X = [x_1, x_2, ..., x_t]$, where x_t signifies the fragment of the time series at the $t - th$ instant in time. The crux of the problem lies in the deployment of a model to implement step-ahead ensemble predictions, $F_{concat}(.)$, thereby allowing the transformation of the input domain into the target domain, $F_{concat}(.)$: $X \rightarrow Y$. Each input, x_t, and target, y_t, come from $\mathbb{R}^{\mathbb{D}+1}$, where $\mathbb{D}$ symbolizes the dimension of the feature, with the feature representing a univariate at the t-th time index. The target attribute, y_t, is defined as $x_{t+s_{step}}$, with s_{step} denoting the number of time steps. Time-series forecasting is considered here where a well-ahead prediction is made or s_{step} is large. The selection of a forecasting methodology is shaped by both the requirements of the practical application and the rhythm of time-series data.

4. Methods

The proposed ensemble-guided model has been developed based on RNN, LSTM, and Conv-LSTM networks. The ensemble-guided network is illustrated as follows.

4.1. RNN

RNNs were developed to overcome the limitations of regular feedforward neural networks, which lack the ability to model data sequencing and time dependencies in historical data Goodfellow et al. (2016). While regular neural networks process data in one direction through hidden layers, RNNs incorporate past information to generate outputs. RNN-based architectures for temporal forecasting applications have been created because of the natural interpretations of time series data as sequences of inputs and targets Salinas et al. (2020); Rangapuram et al. (2018).

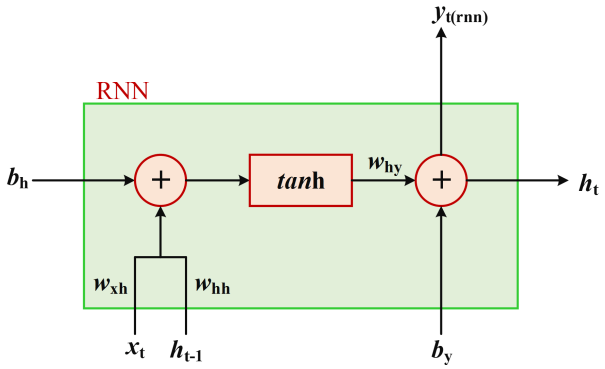


Figure 1: The inner structure of RNN.

. This is accomplished through cells with gates that influence output using historical observations. Fig. 1 provides a inner structure of RNN. RNNs are particularly effective for learning temporal information, as they allow for the computation of a hidden state, h_t, for a given input sequence $X = [x_1, x_2, ..., x_t]$.

$$h_t = tanh(W_{xh} * x_t + W_{hh} * h_{t-1} + b_h) \tag{1}$$

$$y_{t(rnn)} = W_{hy} * h_t + b_y \tag{2}$$

where h_t is the hidden state at time step t, x_t is the input at time step t, W_{xh},W_{hh}, and W_{hy} are weight matrices, b_h and b_y are biased vectors, and is an activation function (*sigmoid* or *tanh*). The output $y_{t(rnn)}$ is a predicted value for time step t.

4.2. LSTM

During backpropagation in the NN, the problem arises due to the vanishing gradient when the derivative becomes smaller and smaller with each layer, and an exploding gradient arises when the derivatives of each successive backpropagating layer increase exponentially, contrary to the disappearing gradients Hanin (2018). Due to difficulties with exploding and vanishing gradients Goodfellow et al. (2016), conventional versions of RNNs can have trouble learning long-range dependencies in the data Hochreiter et al. (2001); Bengio et al. (1994).

This is because they have an infinite look-back window. This can be seen intuitively as a type of resonance in the memory state. LSTM networks have been created to solve these constraints by enhancing gradient flow within the network Hochreiter & Schmidhuber (1997). This is accomplished by modulating a cell state c_t, which holds long-term information, via a sequence of gates as shown below:

$$Inputgate : i_t = (W_{xi} * x_t + W_{hi} * h_{t-1} + b_i) \tag{3}$$

$$Outputgate : o_t = (W_{xo} * x_t + W_{ho} * h_{t-1} + b_o) \tag{4}$$

$$Forgetgate : f_t = (W_{xf} * x_t + W_{hf} * h_{t-1} + b_f) \tag{5}$$

$$Cellstate : c_t = f_t * c_{t-1} + i_t * tanh(W_{xc} * x_t + W_{hc} * h_{t-1} + b_c) \tag{6}$$

$$h_t = o_t * tanh(c_t) \tag{7}$$

$$y_{t(lstm)} = W_{hy} * h_t + b_y \tag{8}$$

In this matrix notation, f_t, i_t, and o_t stand in for the forget, input, and output gateways in that order, while c_t signifies the memory cell. The weight matrices bear labels as W_{xf}, W_{xi}, W_{xo}, W_xc, W_{hf}, W_{hi}, W_{ho}, W_{hc}, and W_{hy}. Bias vectors find their expressions as b_f, b_i, b_o, b_c, and b_y. The hyperbolic tangent activation ($tanh$), and an asterisk * is a symbol for element-wise multiplication. The predictive output for the $t-th$ timestep is denoted as $y_{t(lstm)}$. The complete internal framework of the LSTM is delineated in Fig. 2.

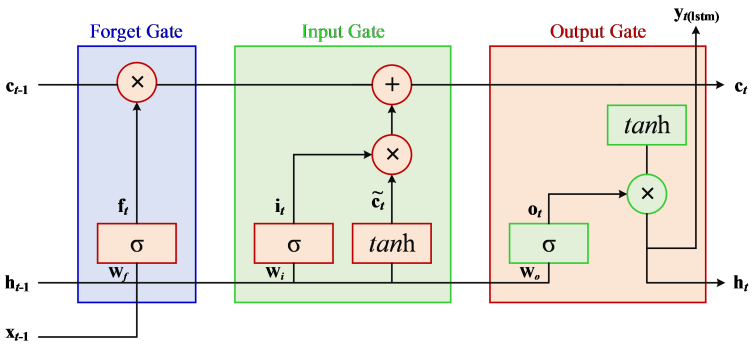


Figure 2: The inner structure of LSTM.

4.3. Conv-LSTM

CNNs have earned their stripes by mining features from lattice-like data patterns, often found within image structures Shi et al. (2015). Originally, CNN's design catered to the needs of two-dimensional data arrays, thereby laying a hierarchical foundation for pattern learning that would pervade the entire network. In addition, their one-dimensional counterparts are gaining momentum in processing linearly arranged data sequences, such as text or time-series data. However, its performance lags when compared to its counterparts such as LSTMs Essien & Giannetti (2020). The anatomy of a typical CNN includes a convolutional layer, a pooling layer, and a fully connected layer O'Shea & Nash (2015). Ingress data vectors find their place in the input layer, followed by the convolutional layer that gleans features using a filter or kernel, eventually feeding into a fully connected layer. The link between an LSTM network and the dense or fully connected layer of the CNN layout is established by swapping matrix addition and dot product operations with convolutional operators, as shown in Fig. 3.

The convolutional layers of the Conv-LSTM model operate on the input tensor, extracting spatial features that are then fed into the LSTM cells. The LSTM cells maintain a temporal context across time steps, allowing the model to capture dependencies between sequential data points. The equations governing the operation of a single Conv-LSTM cell are as follows:

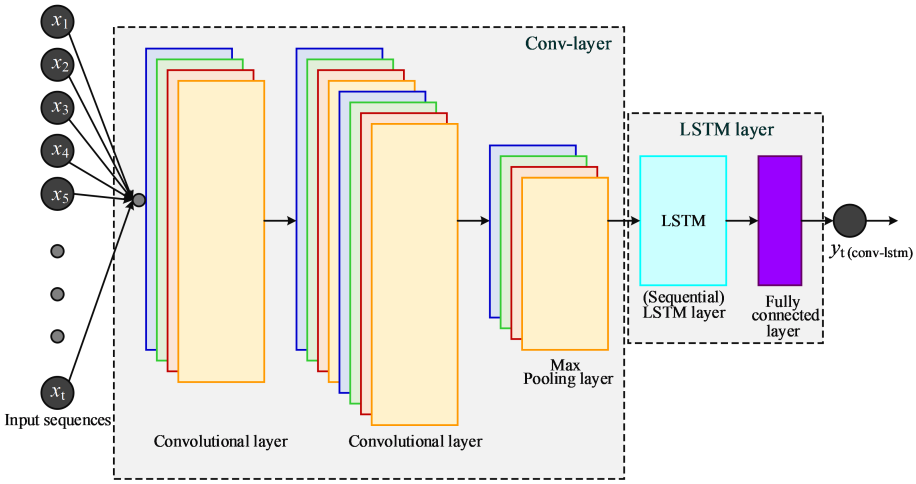


Figure 3: The block diagram of Conv-LSTM.

$$z_t = Conv(x_t) \tag{9}$$

$$Forgetgate :: f_t = (W_f * [h_{t-1}, z_t] + b_f) \tag{10}$$

$$Inputgate : i_t = (W_i * [h_{t-1}, z_t] + b_i) \tag{11}$$

$$outputgate : o_t = (W_o * [h_{t-1}, z_t] + b_o) \tag{12}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot tanh(W_c * [h_{t-1}, z_t] + b_c) \tag{13}$$

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

$$y_{t(conv-lstm)} = Dense(h_t) \tag{15}$$

where z_t is the output of the convolutional layer applied to x_t is the input tensor at time t, while h_{t-1} and c_{t-1} are the hidden state and cell state of the

LSTM cell at the previous time step, and W, and b are learnable weight matrices and bias vectors. $\odot$ is the element-wise multiplication operation. In the Conv-LSTM model, the input tensor x_t is passed through a series of convolutional layers before being fed into the LSTM cells. The output of each convolutional layer is a tensor of shape (batch_size, height, width, channels), which is then fed into the LSTM cells as input at each time step. The Conv-LSTM model can be trained using gradient descent to minimize a loss function, and can be used for time series forecasting tasks.

4.4. Guided Network

Ensemble models combine the predictions of multiple models to achieve better performance than any single model Müller et al. (2022); Dam et al. (2020). In the context of time series forecasting, an ensemble model can combine the strengths of different types of standard RNN ($y_{t(rnn)}$), LSTM ($y_{t(lstm)}$), and Conv-LSTM ($y_{t(conv-lstm)}$) to improve the accuracy of predictions. The output from the RNN layer, LSTM layer, and Conv-LSTM layer are concatenated using a merged layer as an ensemble model F_{concat} as the following representation,

$$y_{t(ens)} = F_{concat}(y_{t(rnn)}, y_{t(lstm)}, y_{t(conv-lstm)}) \tag{16}$$

where, $y_{t(rnn)}$, $y_{t(lstm)}$ and $y_{t(conv-lstm)}$ are the model function output respectively. The final output of the proposed model represented as $y_{t(ens)}$, is obtained by concatenating the output layers from three networks for the t-th samples. However, the challenge remains in determining which model function to select for the backward path. To tackle this issue in an elegant manner, we have introduced a novel strategy that involves leveraging a guide network to dynamically select the most suitable model function.

4.4.1. Learning Strategy: Guided Network for Optimizing Model Selection through Loss Function

Selecting an appropriate model function is crucial for training a model to accurately predict outcomes on new data. One strategy for selecting the model

function is through guided networks using ensemble network outputs. Ensemble networks are collections of multiple models trained on the same data, which are then combined to improve accuracy and reduce over-fitting. Utilizing the results generated by an ensemble network, a guided network can acquire the ability to forecast a significantly improved model function for the given task. This approach proves highly beneficial, especially when dealing with intricate or uncertain real-world datasets that involve crucial seasonal patterns. By leveraging the ensemble network's capability to offer a broader range of reliable and varied outputs, the guided network facilitates the selection of an appropriate loss function, thereby enhancing overall accuracy. Ultimately, this approach can lead to improved performance and better generalization of new data. From the Eq. (16), the error between ensemble model output ($y_{t(ens)}$) and true output (y_t) can be described as follows,

$$\begin{aligned} & min_{j \in \{y_{\mathrm{rnn}}, y_{\mathrm{lstm}}, y_{\mathrm{conv-lstm}}\}} f(\mathrm{err}_{\mathrm{t}}^{\mathrm{j}}) \\ & where, \ \ \mathrm{err}_{\mathrm{t}}^{\mathrm{j}} = \mathcal{L}_{MSE/MAE}(y_t, y_{t(ens)}) \end{aligned} \tag{17}$$

where, the error term, denoted as err_t^j, represents the discrepancy between the ground truth value, y_t, and the ensemble output, $y_{t(ens)}$, corresponding to a specific time step. Typically, this error is evaluated using either the Mean Squared Error (MSE) or Mean Absolute Error (MAE) loss function. The objective is to minimize the loss defined in Eq. (17) in order to select a specific model output. However, this task poses challenges when employing the guided network (g_ϕ). To address this issue, we propose a novel strategy for model selection by minimizing the error defined in Eq. (17). This strategy aims to identify the model output that best aligns with the ground truth value, capitalizing on the guided network's capabilities.

Since the output of err_t^j consistently falls within continuous boundaries, making it more reliable to represent this variable using a one-hot encoding scheme. By adopting this approach, we can select the appropriate err_t^j for the j-th model, guided by its corresponding one-hot-code output Dam et al. (2021),

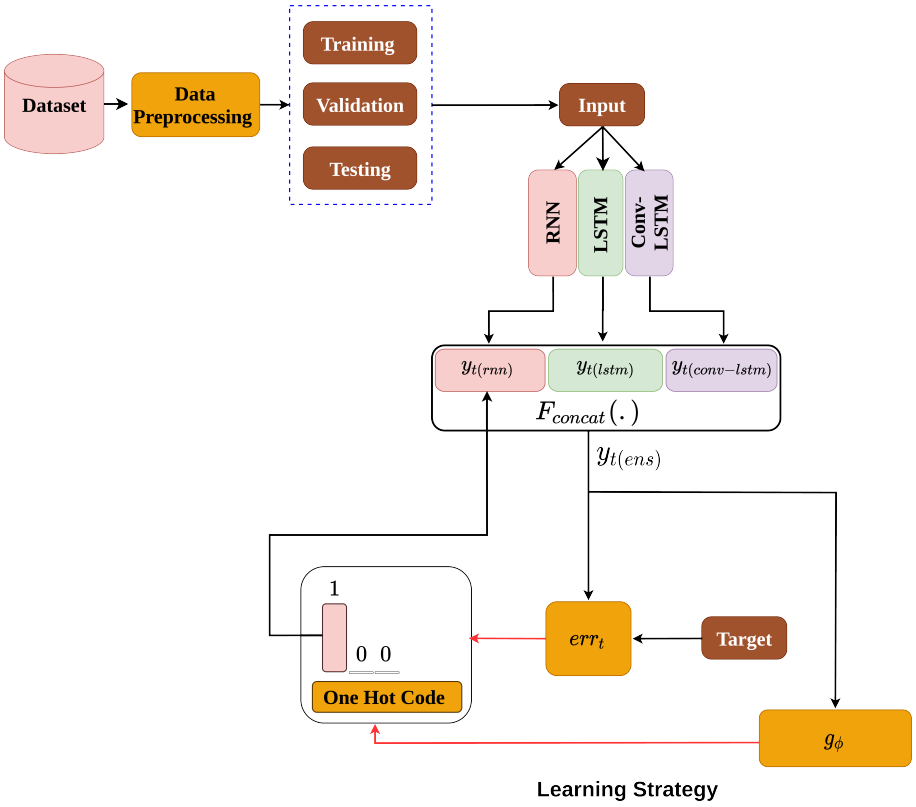


Figure 4: The workflow of GATE is depicted in the figure. GATE incorporates two fundamental components within its ensemble learning scheme. Firstly, the outputs of each network in the ensemble are concatenated using the function $F_{concat}(.)$. Subsequently, we estimate the output for each individual model using the loss function $\mathcal{L}_{MSE/MAE}$. However, the minimum loss value obtained from $\mathcal{L}_{MSE/MAE}$ is assigned as a one-hot code representation, facilitating the learning process of the g_ϕ model. Furthermore, the input to the g_ϕ model is $y_{t(ens)}$, which forces the model to function as a selecting classifier.

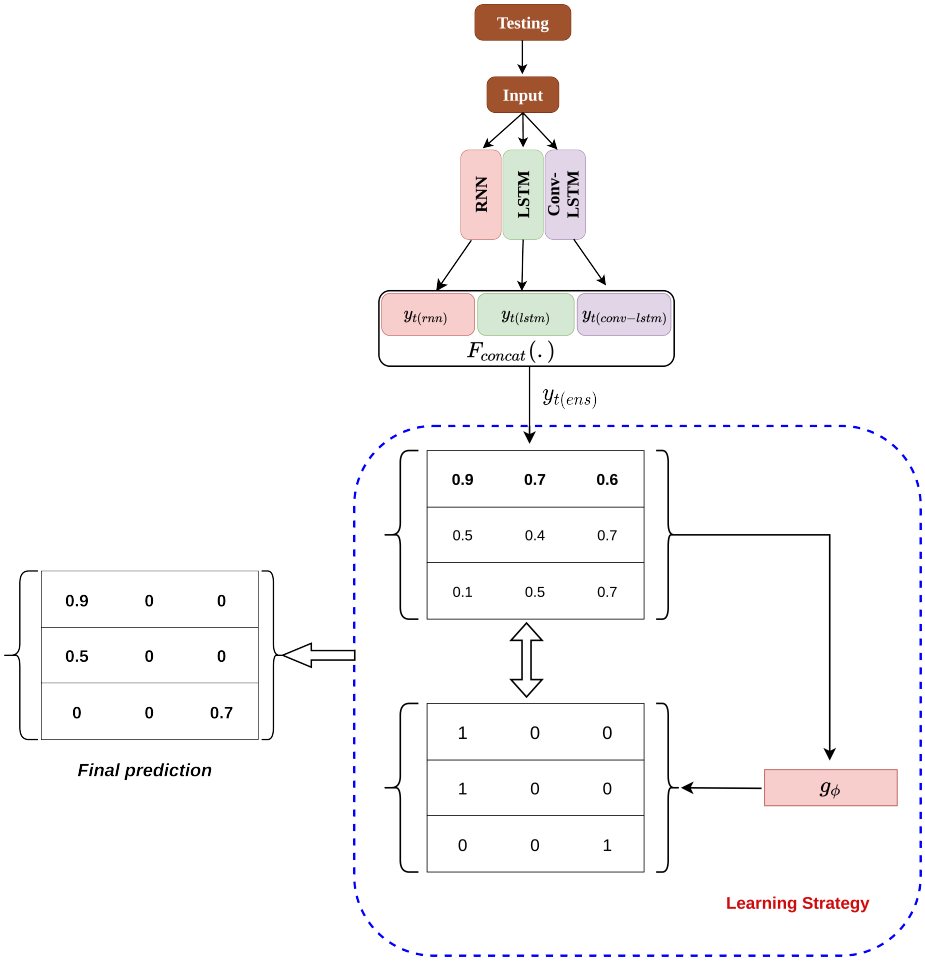


Figure 5: The inference stage of GATE is illustrated in the figure. To provide a clearer understanding of GATE's working principle, let's consider an example where three samples generate three different model outputs. However, the selected model output, determined during the inference stage, originates from the g_ϕ model. This selection process facilitated by g_ϕ assists in mitigating the problem of overfitting associated with individual model performances. By leveraging g_ϕ, GATE ensures that the most suitable model output is chosen, enhancing the overall performance and reducing potential overfitting issues.

Dam (2022). This enables us to express err_t^j in the following manner:

$$err_t^j = \begin{cases} 1 & \text{if } j = \arg\min_k(err^k) \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

The Eq. (18) defines the computation of error values using a one-hot encoding approach. In this Eq. (18), the variable err_t^j represents the error value associated with a specific model category j at a given time step t. By employing a piecewise function, the equation determines whether the error at a particular index is the minimum among all the errors in a given set. If the index j matches the index k that corresponds to the minimum error among all err^k values, err_t^j is set to 1. Conversely, if j is not the index of the minimum error, err_t^j is assigned a value of 0. Specifically, the ensemble output is evaluated to determine the minimum loss, which is assigned a value of 1 while all other unselected loss values are assigned 0 using one-hot encoding. This serves as the output of g_ϕ, but since g_ϕ has no prior knowledge of loss selection during training, it adopts an unsupervised learning approach, enforcing the continuous output of $f(err_t^j)$ through g_ϕ.

$$\hat{y}_{t(ens)} = (y_{t(ens)} - \frac{1}{3}\sum^{3} y_{t(ens)}) \times 100 \tag{19}$$

where $\hat{y}_{t(ens)}$ is the normalized ensemble prediction for sample t obtained by applying the exponential function and using the mean as $\frac{1}{3}\sum^3 y_{t(ens)}$. By performing normalization, the ensemble predictions are scaled to a comparable range as $y_{t(ens)}$. This normalization ensures that the resulting $\hat{y}_{t(ens)}$ can be effectively utilized as an input for g_ϕ. Subsequently, the update process for g_ϕ entails minimizing the MSE loss between the normalized output and the error expressed in the one-hot-code form, which functions as a selecting classifier. This process can be mathematically expressed as follows:

$$\mathcal{L}_{MSE} = \sum \frac{1}{2}(err_t - g_\phi(\hat{y}_{t,\text{ens}}))^2 \tag{20}$$

Algorithm 1 GATE

Input: Time series data $X^{train} = \{(x_1, y_1), (x_2, y_2), ..., (x_T, y_{T+\tau})\}_{i=1}^{N}$, Similarly, $X^{test} = \{(x_1^t, y_1^t), (x_2^t, y_2^t), ..., (x_T^t, y_{T+\tau}^t)\}$, number of steps ahead m, three ensemble models outputs with the parameters $F_{concat}(.)$, $\{F_{concat}(.) = y_{rnn}, y_{lstm}, y_{conv-lstm}\}$, guided network parameters ϕ batch=B, epoch =n_{epoch}, Adam hyper− parameters (l_r, π_1, π_2), l_{lag} = selected number of lags (30), s_{step} = selected number of step (10,20,30).

Output: Ensemble output $F_{concat}(.)$

$F_{concat}(.)$ and $g_\phi \sim \mathcal{N}(0, 0.02^2)$

for p in $\{1, 2, ..., n_{epoch}\}$ **do**

  /* Ensemble output the input with batch, $B \in X^{train}$ */

  $\{y_{t(ens)}\}_{i=1}^{B} \leftarrow F_{concat}(.)\{x_i\}_{i=1}^{B}$

  /* Calculate the error residuals in Eq. (17 */

  $\{\mathrm{err_t}\}_{i=1}^{B} = (\mathcal{L}_{MSE/MAE}(y_t - y_{t(ens)})$

  /* Update the ensemble model function parameters $F_{concat}(.)$ */

  $F_{concat} \rightarrow (\nabla_{F_{concat}} \frac{1}{B} \mathcal{L}_{MSE/MAE}), \beta, l_r, \pi_1, \pi_2)$

  /* Calculate the one-hot-code form of the err as in Eq (18) */

  $$err_t^j = \begin{cases} 1 & \text{if } j = \arg\min_k(err^k) \\ 0 & \text{otherwise} \end{cases}$$

  /* The normalisation ensemble output as in Eq. (19) */

  $\hat{y}_{t(ens)} = (y_{t(ens)} - \frac{1}{3}\sum^{3} y_{t(ens)}) \times 100$

  /* Guide network is updated as a selecting classifier in Eq. (20) */

  g_ϕ update through MSE loss

  $\phi \rightarrow (\nabla_\phi \frac{1}{B} \mathcal{L}_{MSE}), \beta, l_r, \pi_1, \pi_2)$

  if n_{epoch} % 1 == 0 **then**

    /* Estimate prediction values $F_{concat}(.)$ through g_ϕ */

    $F_{concat}(.)$ for X^{test} datasets through g_ϕ.

  end if

end for

To gain a deeper comprehension of the GATE pipeline, which encompasses the selection of the learning strategy, refer to Fig. 4. Additionally, Fig. 5 illustrates the inference mechanism employed during testing, featuring a simple example for better clarity. Furthermore, algorithm 1 contains a description of the algorithm's underlying structure.

4.5. Performance Indicators

In addition to MAE and RMSE, the coefficient of determination (R^2) in Eq. (23) is used in time series forecasting to measure how well the model fits the data and how accurately it can make predictions. A high R^2 value indicates a good fit and accurate predictions, while a low R-squared value suggests a poor fit and less accurate predictions. The final loss function is defined as follows,

$$\mathcal{L}_{MAE} = \sum_{t=1}^{N} |(y_t - \hat{y})| \tag{21}$$

$$\mathcal{L}_{RMSE} = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(y_t - \hat{y})^2} \tag{22}$$

$$R^2 = 1 - \frac{\sum_{t=1}^{N}(y_t - \hat{y_t})^2}{\sum_{t=1}^{N}(y_t - \bar{y_t})^2} \tag{23}$$

where y_t $\bar{y_t}$ are the real value and mean real value respectively, the predicted value is $\hat{y}$, and N is the total number of sample points.

5. Experiments and Analysis

In this study, the proposed GATE model's accuracy, stability, and generalization capability are verified using three distinct types of time series datasets shown in Fig. 6, three baseline models namely RNN, LSTM, and Conv-LSTM, and four widely accepted evaluation metrics such as MAE, MSE, RMSE and R^2. To ensure fair comparison and assess the models' ability to generalize across diverse datasets, identical experimental parameters were adopted for each model,

including a past 30-time step sequence length for the input data and other essential parameters listed in Table 1. The following subsections elaborate on the datasets, and experimental outcomes with detailed analysis.

Hardware & Softwares	**Specifications**
Computer	Operating System: Ubuntu 20.04.5 LTS CPU: 11th Gen Intel(R) Core(TM) i9-11950H @ 2.60GHz × 16; RAM: 32 G) GPU: GeForce GTX 2060 (6G memory
Software	Spyder-5.3.3 + TensorFlow-2.10.0 + Keras- 2.10.0 + NVIDIA Driver-470.161.03 + CUDA-10.2

Table 1: Hardware and software specifications.

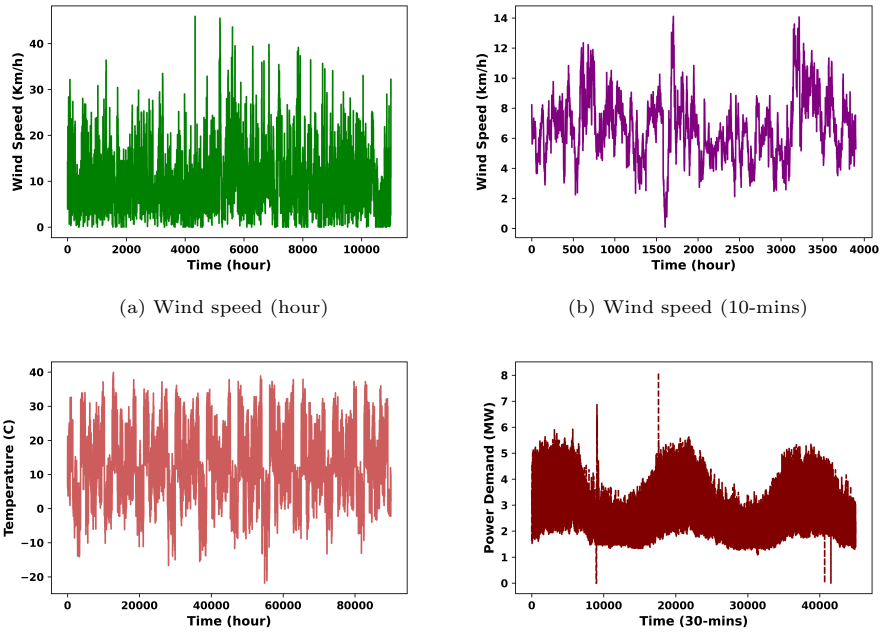
5.1. Time Series Datasets

5.1.1. Wind Speed and Temperature (Hour)

The dataset for weather conditions in the Szeged region between 2006 and 2016 is a detailed and extensive collection of hourly and daily weather summaries. It provides a comprehensive overview of various weather variables, such as temperature, humidity, wind speed, precipitation, and atmospheric pressure in the area. The dataset can be utilized by local governments and emergency services to develop effective strategies for managing extreme weather events such as heatwaves, droughts, and floods. Additionally, temperature and weather time series data have been considered in this forecasting experiment which is shown in Fig. 6(a) and 6(c) respectively Budincsevity (2018). In this dataset, the total time step of wind speed is 11000 shown in Fig. 6(a), where the time interval of the dataset is 1-hour. Moreover, there is the 90000-time step of the temperature dataset shown in Fig. 6(c), where the time interval is 1-hour.

5.1.2. Wind Speed (10-mins)

The dataset comprises wind speed measurements from a single turbine situated in an inland wind farm Ding (2019). Data was collected over a two-year



(a) Wind speed (hour)

(b) Wind speed (10-mins)

(c) Temperature (hour)

(d) Power demand (30-mins)

Figure 6: Real time datasets.

period, spanning from December 2014 to December 2015, with a sampling interval of 10-minutes. There are no missing values in the dataset, which contains purely numerical data. No significant outliers or anomalies were observed in the data. Fig. 6(b) depicts 3900-time steps with the period of 10-minutes wind speed dataset.

5.1.3. Power Demand (30-Mins)

The Western Power Distribution (WPD) Presumed Open Data (POD) competition in 2021 was a platform where participants could showcase their analytical and predictive skills by working with a large dataset of half-hourly electricity demand data from a substation near Plymouth, UK Singleton (2021). This dataset is available under the WPD Open Data Licence, which allows for free access, use, and distribution of the data shown in Fig. 6(d). It covers a period of almost three years, starting from 3^{rd} November 2017 to 2^{nd} July 2020, and provides a comprehensive view of the energy usage patterns in the area over time. The dataset is measured in megawatts (MW), which is a unit of power that represents the amount of energy that can be generated or consumed in a given time period. The 30-minutes data points provide detailed information on the electricity demand at the substation where the total time-steps was 45000. This dataset is an invaluable resource for anyone interested in the energy sector, especially in the UK, as it can be used to develop new insights and strategies for managing electricity demand and reducing energy consumption in the future.

5.2. Data Preprocessing

Although the data is sourced from open data, it requires pre-processing. The process of data cleansing is carried out to handle missing data and rectify any inconsistencies. Time series datasets possess both strong randomness and trend characteristics, and they are related to numerous complex factors. To ensure compatibility with the output range of the activation functions in RNN, LSTM, and Conv-LSTMs, the data is scaled to the range of [0, 1] using min-max normalization. This guarantees that each input data contributes proportionally

to the outputs. Additionally, data scaling is used to ensure that numeric data has the same range of values. Normalizing the dataset enhances the model accuracy. The normalized x_i can be defined as Tan et al. (2019),

$$\bar{x}_i = \frac{x_i - x_{min}}{x_{max} - x_{min}} \tag{24}$$

The variables x_{min} and x_{max} correspond to the minimum and maximum values of the variables in the time series. Simultaneously, the time-related variables are provided for a given time period i.

Each dataset has been separated into training, validation, and testing portions for the proposed model. The 70% training dataset was used to train the model, enabling it to acquire knowledge from a wide variety of observations. As the validation set for hyperparameter tuning and model selection, we set aside 10% of the data. This validation assisted in evaluating the performance of the model on unobserved data during the training process. The 20% of the remaining data was designated as the experimental set. It was used solely for evaluating the performance of the final model and its ability to predict on wholly unobserved data, as illustrated in Fig. 10, where 10-step prediction for all datasets is shown. In addition, to decide the number of previous time steps to take into consideration while generating predictions for future time steps, we evaluated 30-time windows for forecasting in our suggested model. In order to provide precise predictions, it enables the model to capture temporal patterns and relationships in the data. Additionally, we have chosen the time horizon that determines the number of forecasting time steps. We have forecasted 10, 20, and 30-time horizons in our proposed prediction model for the proposed model shown in Table 3, Table 4, and Table 5 respectively.

The dispersion refers to the amount of spread in a dataset. For the dispersions in Fig. 7, Fig. 8, and Fig. 9, the x-axis represents the range of the observation, and the y-axis represents the density of observation a given score. The dispersion of the observation is divided into before and after the transformation for RNN, LSTM, and Conv-LSTM for each dataset. The data

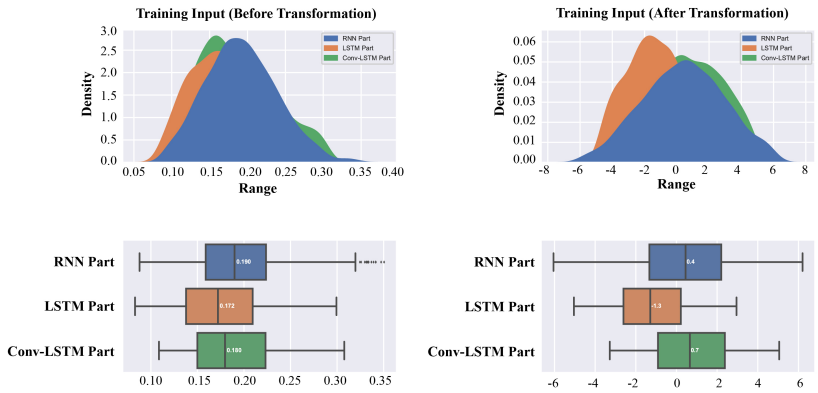


Figure 7: Dispersion of training input.

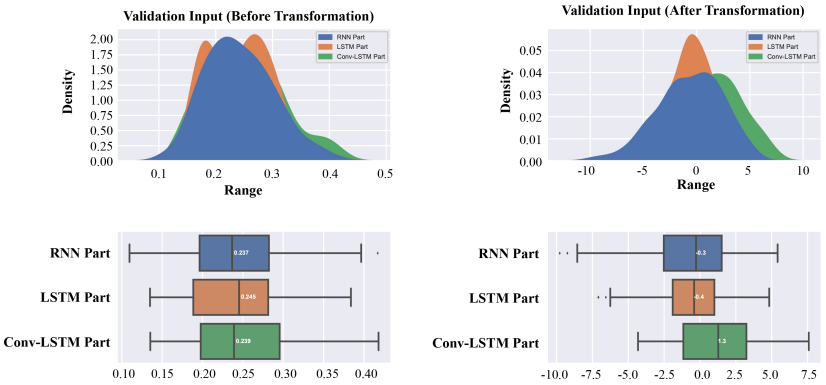


Figure 8: Dispersion of Validation input.

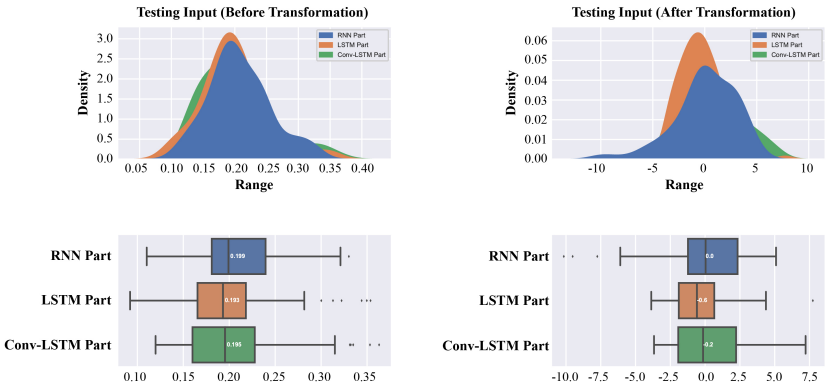


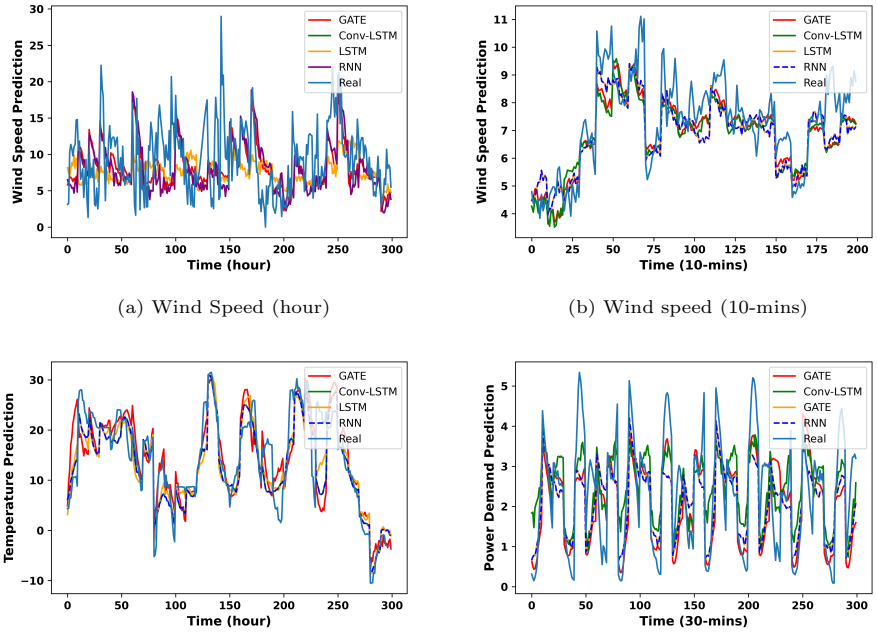
Figure 9: Dispersion of testing input.

transformation process reflects the changes in the values of data points in a dataset for improving the accuracy of a model and making the data more understandable. When data is transformed, it is possible that the dispersion of the data will change. This means that the distribution of the data points may become more spread out or more clustered. The dispersion of the data can have a significant impact on the performance of a model. Overfitting of the model can be affected by more dispersion. Fig. 7 displays the dispersion of training input observation of wind speed (hour), and Fig. 8 and Fig. 9 show the dispersion of validation and testing input observations respectively. Each figure shows the 'before' and 'after' transformation of observation. From Fig. 7, the data before the transformation of data is quite disperse compared to after transformation. Similarly, the validation and testing dispersion of data after transformation is less compared to that before dispersion. Based on the above discussion, it can be concluded that, the data transformation assists the models to overcome the overfitting problem.

5.3. Results Analysis

Fig. 10 displays the results of four prediction models, namely GATE, RNN, LSTM, and Conv-LSTM, along with the real data. In addition, the subplots of Fig. 10 present the four different datasets. It is evident from Fig. 10 that GATE outperforms the other models in terms of prediction accuracy. The Conv-LSTM and LSTM models follow GATE in terms of prediction accuracy, while the RNN model is the least accurate among the four models. The GATE model accurately captures the complex temporal relationships in the data and is able to make more precise predictions. On the other hand, the Conv-LSTM and LSTM models are also able to capture the temporal dependencies but with relatively lower accuracy than GATE. The RNN model, however, fails to capture the temporal dependencies effectively, resulting in poor prediction accuracy. These results have significant implications for practical applications that rely on time series prediction for making a decision for the future. The GATE model can be used as a more reliable and accurate alternative to other models, especially in situations where stable and accurate predictions are critical. The LSTM and Conv-LSTM models can also be used in situations where high accuracy is desirable but not necessarily critical. However, caution should be exercised while using the RNN model for time series prediction, as it may not produce reliable results.

Fig. 11 displays the RMSE values for four different datasets using four different algorithms: RNN, LSTM, Conv-LSTM, and GATE. The x-axis represents the algorithm used, while the y-axis represents the RMSE value. For the wind speed (hour) dataset shown in Fig. 11(a), the GATE algorithm had the lowest RMSE of 0.321, followed by Conv-LSTM with an RMSE of 0.351, LSTM with an RMSE of 0.353, and RNN with an RMSE of 0.367. The GATE algorithm achieved the lowest RMSE of 0.221 for the wind speed dataset (10 minutes) in Fig. 11(b). LSTM followed with an RMSE of 0.267, Conv-LSTM with an RMSE of 0.316, and RNN with an RMSE of 0.278. From the Fig. 11(c), GATE had the lowest RMSE of 0.150, followed by Conv-LSTM with an RMSE of 0.193, LSTM with an RMSE of 0.223, and RNN with an RMSE of 0.198.



(a) Wind Speed (hour)

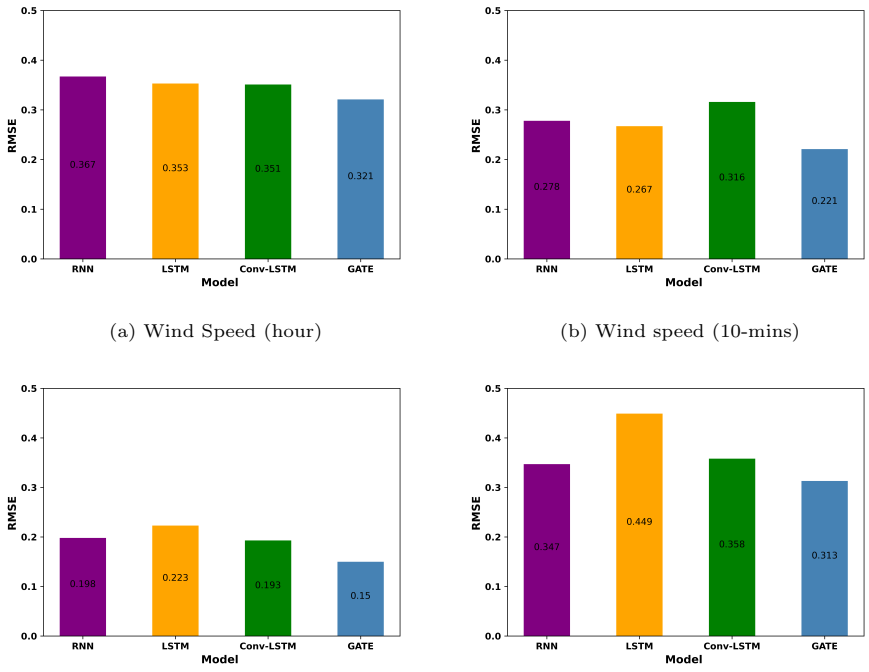
(b) Wind speed (10-mins)

(c) Temperature (hour)

(d) Demand (30-mins)

Figure 10: Forecasting of different models and datasets.

Finally, power demand (30-mins) dataset shown in Fig. 11(d), GATE had the lowest RMSE of 0.313, followed by Conv-LSTM with an RMSE of 0.358, RNN with an RMSE of 0.347, and LSTM with an RMSE of 0.449. GATE algorithm consistently outperformed the other algorithms with percentage improvements ranging from 8.8% to 30.9%, indicating superior performance.



(a) Wind Speed (hour)

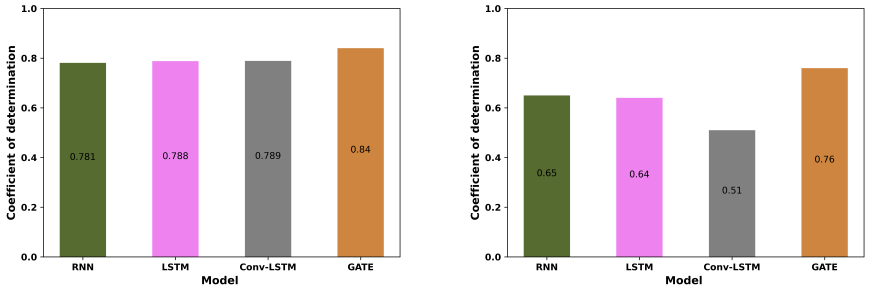
(b) Wind speed (10-mins)

(c) Temperature (hour)

(d) Power demand (30-mins)

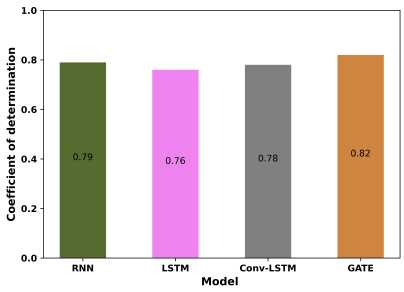
Figure 11: Root mean square error (RMSE) of the models.

Fig. 12 presents the R^2 which is a statistical metric to determine the performance of the model. In this case, there were four datasets used, each containing the R^2 values for four different models: RNN, LSTM, Conv-LSTM, and GATE. The R^2 values range from 0 to 1, with higher values indicating a better fit between the model and the data. The wind speed (h) shown in Fig. 12(a) has R^2 values of 0.781, 0.788, 0.789, and 0.84. This suggests that the GATE model has the best fit with the data, followed closely by the Conv-LSTM and LSTM

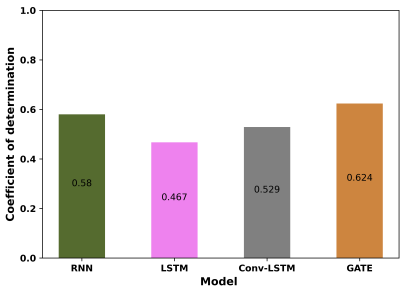


(a) Wind Speed (hour)

(b) Wind speed (10-mins)



(c) Temperature (hour)



(d) Power demand (30-mins)

Figure 12: Coefficient of determination (R^2) of the models.

models. The RNN model has the lowest R^2 value, indicating that it has the poorest fit. In contrast, for the second wind speed (10-minus) dataset as shown in Fig. 12(b), the R^2 values for the four models are lower than those in the first wind speed (h) dataset, ranging from 0.51 to 0.76. The GATE model still has the highest R^2 value, while the difference between the models is not as significant as in the first wind speed (h) dataset. Similarly, for the temperature and demand power demand dataset, the R^2 value of the GATE model is higher than the baseline models are shown in Fig. 12(c) and Fig. 12(d) respectively. Based on Fig. 12 the GATE model consistently performs the best across all four datasets, as it has the highest R^2 value in each dataset and the Conv-LSTM model consistently outperforms the RNN and LSTM models.

Table 2 depicts the experimental hyper-parameters of the proposed model

for four different algorithms: RNN, LSTM, Conv-LSTM, and guided network g_ϕ. The hyper-parameters include batch size, learning rate, number of epochs, number of layers, dropout, dense layer, and activation functions have been used. For the RNN algorithm, a batch size of 366, 129, 2999, and 1499 for datasets was used along with a learning rate of 0.01 and the number of epochs as 50. The activation function *tanh* has been used and the number of layers was 30 and 20. For LSTM and Conv-LSTM, the same hyperparameters have been used as that for RNN, while the activation functions were *tanh* and *ReLu*. Finally, for the guided network algorithm, a batch size of 366, 129, 2999, and 1499 was employed along with a learning rate of 0.01, number of epochs of 50, dropout of 0.05 and 0.3, and a dense layer of 5 and 8. The activation functions were *tanh*, *ReLu*, and *SELU*. Table 2 provides the different hyper-parameters used for each algorithm in the proposed model, which can be helpful for reproducing the experiments and comparing the results with other models.

Table 3 presents the performance indicators, namely MAE and MSE of four different models, namely RNN, LSTM, Conv-LSTM, and GATE, on four different datasets: wind speed (h), wind speed (10-mins), temperature (h), and power demand (30-mins) for 10-step forecasting. For each dataset, Table 3 shows the MAE and MSE values for each model during both the training and testing phases. Additionally, Table 3 highlights the best performance achieved among the models for each dataset and performance indicator. From Table 3, it can be noticed that the GATE model performs the best among the models on all four datasets for both MAE and MSE indicators, during both the training and testing phases. For instance, in the wind speed (h) dataset, the GATE model achieved an MAE of 0.142 and an MSE of 0.044 during training, while the best-performing from other models (LSTM) achieved an MAE of 0.168 and an MSE of 0.054. This represents an improvement of around 15% in MAE and 19% in MSE compared to the best non-GATE model on this dataset during training. During training, for the wind speed (10-mins) dataset, the GATE model outperformed other models with an MAE of 0.116 and an MSE of 0.022, while the LSTM model achieved an MAE of 0.148 and an MSE of 0.026. This corresponds

Algorithms	Experimental Parameter	Parameter Value
RNN	Batch size	366, 129, 2999, 1499
	Learning rate	0.01
	Number of epochs	50
	Number of layer	30 and 20
	Activation function	*Tanh*
LSTM	Batch size	366, 129, 2999, and 1499
	Learning rate	0.01
	Number of epochs	50
	Number of layer	30 and 20
	Activation function	*Tanh* and *ReLU*
Conv-LSTM	Batch size	366, 129, 2999, and 1499
	Learning rate	0.01
	Number of epochs	50
	Number of layer	30 and 20
	Activation function	*Tanh* and *ReLU*
Guided Network g_ϕ	Batch size	366, 129, 2999, and 1499
	Learning rate	0.01
	Number of epochs	50
	Dropout	0.05 and 0.3
	Dense layer	5 and 8
	Activation function	*Tanh*, *ReLU* and *SELU*

Table 2: Experimental hyper-parameters parameters of the proposed model.

to a 22% improvement in MAE and 15% improvement in MSE compared to the best non-GATE model on this dataset. The dataset of temperature showed that the GATE model achieved the most accurate results with the lowest MAE and MSE values for both training and testing data. The other models had comparable performances, except for the RNN model, which had the highest MSE value for testing data. For the power demand, the GATE model achieved the most accurate results with the lowest MAE and MSE values for both training and testing data. In contrast, the Conv-LSTM model had the highest MAE and MSE values, indicating the lowest accuracy. The LSTM and RNN models had comparable performances but were better than the GATE model. Similarly, from Table 4 and Table 5, it can be concluded that the GATE model shows better performance for 20 and 30-step forecasting compared with the baseline

Dataset	Error	Model	RNN	LSTM	Conv-LSTM	GATE
Wind speed (h)	MAE	Training	0.172	0.168	0.199	**0.142**
	MSE		0.049	0.054	0.068	**0.044**
	MAE	Testing	0.269	0.267	0.279	**0.242**
	MSE		0.148	0.153	0.139	**0.101**
Wind speed (10-mins)	MAE	Training	0.149	0.148	0.154	**0.116**
	MSE		0.025	0.026	0.039	**0.022**
	MAE	Testing	0.209	0.201	0.234	**0.152**
	MSE		0.055	0.065	0.093	**0.048**
Temperature (h)	MAE	Training	0.069	0.068	0.068	**0.061**
	MSE		0.016	0.017	0.016	**0.010**
	MAE	Testing	0.148	0.169	0.150	**0.127**
	MSE		0.037	0.041	0.036	**0.032**
Power demand (30-mins)	MAE	Training	0.053	0.054	0.063	**0.049**
	MSE		0.006	0.006	0.009	**0.003**
	MAE	Testing	0.258	0.355	0.275	**0.237**
	MSE		0.129	0.198	0.125	**0.108**

Table 3: Performance indicators of 10-step forecasting.

models. In summary, Table 3, Table 4, and Table 5 results indicate that the GATE model exhibits superior performance compared to the other models for all four datasets for 10, 20 and 30 step forecasting, with the lowest MAE and MSE values observed during both the training and testing phases.

Therefore, GATE with the guiding network can help prevent overfitting by combining the outputs of multiple models, each of which may have different biases and variances. By combining multiple models in GATE, the ensemble can better capture the underlying patterns and relationships in the data and reduce the variance in the predictions. In addition, the guiding network plays a crucial role in ensemble learning by selecting the best model for each batch of data based on its performance on training samples. By doing so, the GATE can prevent overfitting by choosing models that have learned to generalize well to the data rather than models that simply memorize the current batch. Moreover, the guiding network has the ability to modify the weights allocated to the output of each model by evaluating its performance as a one-hot code. This strategy helps avoid any single model from overwhelming the ensemble. By merging

20 Step Forecasting						
Dataset	**Error**	**Model**	**RNN**	**LSTM**	**Conv-LSTM**	**GATE**
Wind speed (h)	MAE	Training	0.221	0.229	0.217	**0.162**
	MSE		0.073	0.071	0.068	**0.052**
	MAE	Testing	0.287	0.283	0.267	**0.251**
	MSE		0.168	0.159	0.149	**0.127**
Wind speed (10-mins)	MAE	Training	0.178	0.168	0.181	**0.138**
	MSE		0.034	0.035	0.033	**0.028**
	MAE	Testing	0.209	0.206	0.214	**0.182**
	MSE		0.07	0.071	0.074	**0.05**
Temperature (h)	MAE	Training	0.09	0.086	0.077	**0.07**
	MSE		0.024	0.023	0.019	**0.016**
	MAE	Testing	0.237	0.258	0.228	**0.173**
	MSE		0.058	0.058	0.057	**0.048**
Power demand (10-mins)	MAE	Training	0.079	0.079	0.071	**0.045**
	MSE		0.009	0.009	0.008	**0.004**
	MAE	Testing	0.398	0.377	0.349	**0.313**
	MSE		0.205	0.235	0.196	**0.163**

Table 4: Performance indicators of 20-step forecasting

30 Step Forecasting						
Dataset	**Error**	**Model**	**RNN**	**LSTM**	**Conv-LSTM**	**GATE**
Wind speed (h)	MAE	Training	0.128	0.126	**0.124**	**0.196**
	MSE		0.075	0.081	**0.074**	**0.055**
	MAE	Testing	0.321	0.332	**0.301**	**0.274**
	MSE		0.197	0.225	**0.189**	**0.137**
Wind speed (10-mins)	MAE	Training	0.239	0.246	**0.227**	**0.178**
	MSE		0.045	0.043	**0.04**	**0.34**
	MAE	Testing	0.246	0.258	**0.235**	**0.215**
	MSE		0.096	0.098	0.096	**0.059**
Temperature (h)	MAE	Training	0.198	0.196	0.187	**0.146**
	MSE		0.025	0.025	0.023	**0.02**
	MAE	Testing	0.257	0.268	0.239	**0.212**
	MSE		0.081	0.081	0.075	**0.051**
Power demand (10-mins)	MAE	Training	0.079	0.077	0.075	**0.052**
	MSE		0.007	0.009	0.007	**0.005**
	MAE	Testing	0.437	0.416	0.397	**0.337**
	MSE		0.279	0.285	0.261	**0.196**

Table 5: Performance indicators of 30-step forecasting

the results from GATE and preventing overfitting, the GATE generated by the guiding network can deliver more precise and resilient forecasts. This is particularly useful in scenarios where the individual models have a tendency to overfit, a phenomenon that has been detected during training and testing across all datasets.

5.4. Ablation Studies

The findings of the ablation investigation are shown in Table 6. In order to comprehend the effects of our suggested GATE model, we investigated the model performance in two alternative model settings using guided methodologies. The claimed GATE model is tested against three existing guided network models for various datasets and time intervals: RNN, LSTM; LSTM, Conv-LSTM; and Conv-LSTM, RNN. The GATE model prevails over the other models significantly for the "Wind speed (h)" dataset. It obtains an MAE of 0.242, an improvement of 7.66% in comparison to the RNN, LSTM model (0.261). Similar to this, the GATE model shows a percentage improvement of 5.84% over the LSTM, Conv-LSTM model (0.257). Additionally, the GATE model surpasses the other models in terms of MSE with a value of 0.101, which represents improvements of 27.34% and 21.79%, respectively, over the RNN, LSTM model (0.139) and the LSTM, Conv-LSTM model (0.128).

Dataset	**Error**	**RNN, LSTM**	**LSTM, Conv-LSTM**	**Conv-LSTM, RNN**	**GATE**
Wind speed (h)	MAE	0.261	0.257	0.267	**0.242**
	MSE	0.139	0.128	0.136	**0.101**
Wind speed (10-mins)	MAE	0.198	0.218	0.193	**0.152**
	MSE	0.061	0.069	0.063	**0.048**
Temperature (h)	MAE	0.147	0.152	0.149	**0.127**
	MSE	0.038	0.036	0.034	**0.032**
Power demand (30-mins)	MAE	0.323	0.296	0.251	**0.237**
	MSE	0.159	0.143	0.123	**0.108**

Table 6: Comparison of ablation study results for various ensemble network configurations: dual networks (comprising any two of RNN, LSTM, and Conv-LSTM vs. GATE)

The performance of the GATE model is still better for the "wind speed (10-mins)" dataset. It obtains an MAE of 0.152, which represents improvements of

23.23% over the RNN, LSTM model and 30.28% over the LSTM, Conv-LSTM model, respectively. Additionally, the GATE model outperforms the others with an MSE of 0.048, showing percentage improvements of 21.31% over the RNN, LSTM model (0.061), and 30.23% over the LSTM, Conv-LSTM model (0.069). On the "Temperature (h)" dataset, the GATE model achieves better as well. It obtains an MAE of 0.127, which represents improvements of 13.61% and 16.45% over the RNN, LSTM model (0.147), and LSTM, Conv-LSTM model (0.152), respectively. Additionally, the GATE model obtains an MSE of 0.032, demonstrating percentage improvements of 15.79% over the RNN, LSTM model (0.038) and 11.11% over the LSTM, Conv-LSTM model (0.036). Finally, the GATE model carries out significantly for the "Power demand (30-mins)" dataset. It becomes an MAE of 0.237, reflecting improvements of 19.26% over the LSTM, Conv-LSTM model (0.296), and improvements of 26.71% over the RNN, LSTM model (0.323). Furthermore, the MSE for the GATE model is 0.108, which is a 32.08% increase over the RNN, LSTM model's 0.159, and a 24.13% improvement over the LSTM, Conv-LSTM model's 0.143 percentages. It can be concluded that the GATE model consistently beats the competition. Compared to the RNN, LSTM, LSTM, Conv-LSTM, and Conv-LSTM, RNN models, it shows considerable percentage gains in terms of MAE and MSE. These findings reveal the higher predictive capabilities of the GATE model, showing its potential to enhance accuracy.

6. Conclusions and Future Directions

This research introduces an advanced ensemble-influenced approach for precise and consistent forecasting of time series, a tool that holds promise to augment efficiency and various facets of quotidian existence. The proposed methodology was scrutinized utilizing four real-time series databases, encompassing wind velocity (measured at the minute and hourly periods), ambient temperature, and power utilization. To uphold the credibility of the conducted experiments, the data clusters were harvested from a trio of geographically dis-

tinct locales.

The model at hand was juxtaposed against an array of other forecasting frameworks, utilizing four frequently employed statistical yardsticks to appraise its efficacy. This inquiry ascertained that the newly suggested model trumped its counterparts, and a trinity of inferences was extrapolated from the analytical examination of the experimental data.

- The newly proposed GATE framework, the brainchild of this research, exhibits superior performance in comparison to its competitors (RNN, LSTM, and Conv-LSTM) in the realm of time series forecasting across diverse application arenas, such as wind speed, temperature, and power demand. This model excels in precision, consistency, and extrapolation during the process of forecasting.

- The employed unsupervised learning strategy in this model effectively mitigates the issue of overfitting and enhances the precision and heterogeneity of rudimentary predictors by modulating the sample distribution, even when constrained by limited sample quantities.

- Furthermore, this model demonstrates exceptional exploratory and exploitative prowess when handling both voluminous and compact data sets. In addition, it possesses the capability to pinpoint an optimal combination solution with noteworthy accuracy and robust stability.

Enlightened by the results of the present study, we are motivated to venture into uncharted territories in the realm of time series forecasting. As we turn our gaze toward the horizon, we envisage several promising directions for future research:

- While the present GATE model has shown its mettle across diverse application fields, further studies could aim to customize and refine it for specific industries or sectors. For instance, specialized versions of the model could be developed for the renewable energy sector, transportation planning, or healthcare data analysis.

- The unsupervised learning strategy employed in this model has proven its worth in enhancing prediction accuracy and diversity, even with limited samples. However, the performance of this strategy could be further investigated under varying conditions of data availability and quality. This could include exploring ways to better handle missing or inconsistent data, as well as developing methods to automatically adjust the sample distribution based on the characteristics of the data.

- The model's exceptional ability to handle both large and small datasets opens up avenues for exploring hybrid data approaches. Future studies could investigate how the model performs when dealing with mixed data types, such as combining structured and unstructured data or integrating real-time data streams with historical data.

- Although the GATE model has demonstrated strong stability, it would be valuable to examine how it performs under different types of uncertainty, such as sudden changes in the data patterns or the introduction of unexpected outliers. This could lead to the development of more robust and resilient forecasting models that can better withstand real-world uncertainties.

- Lastly, future research could also focus on increasing the transparency and interpretability of the model. While the GATE model achieves high accuracy, it is crucial that users understand how the model arrives at its predictions. This will increase trust in the model's outputs and facilitate their practical application.

By forging ahead along these paths, we aspire to propel the GATE model to new heights and make even greater strides in the field of time series forecasting.

CRediT authorship contribution statement

Md Rasel Sarkar: Design, Methodology, Investigation, & Writing. **Sreenatha G. Anavatti**: Review, Editing, & Supervision. **Tanmoy Dam**: Writing &

Editing. **Md Meftahul Ferdaus**: Writing & Editing. **Murat Tahtali**: Review & Editing, Supervision. **Savitha Ramasamy**: Review & Editing. **Mahardhika Pratama**: Review, Editing, & 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.

References

Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, *5*, 157–166.

Budincsevity, N. (2018). Dataset: Weather in szeged 2006–2016, kaggle, san francisco, ca, usa. [Online]. Available: https://www.kaggle.com/budincsevity/szeged-weather. [Accessed 14 May 2023].

Chen, J., Zeng, G.-Q., Zhou, W., Du, W., & Lu, K.-D. (2018). Wind speed forecasting using nonlinear-learning ensemble of deep learning time series prediction and extremal optimization. *Energy conversion and management*, *165*, 681–695.

Cirstea, R.-G., Micu, D.-V., Muresan, G.-M., Guo, C., & Yang, B. (2018). Correlated time series forecasting using multi-task deep neural networks. In *Proceedings of the 27th acm international conference on information and knowledge management* (pp. 1527–1530).

Dam, T. (2022). *Developing Generative Adversarial Networks for Classification and Clustering: Overcoming Class Imbalance and Catastrophic Forgetting*. Ph.D. thesis UNSW Sydney.

Dam, T., Anavatti, S. G., & Abbass, H. A. (2020). Mixture of spectral generative adversarial networks for imbalanced hyperspectral image classification. *IEEE Geoscience and Remote Sensing Letters*, *19*, 1–5.

Dam, T., Anavatti, S. G., & Abbass, H. A. (2021). Improving clustergan using self-augmentedinformation maximization of disentangling latent spaces. *arXiv preprint arXiv:2107.12706*, .

Ding, M., Zhou, H., Xie, H., Wu, M., Liu, K.-Z., Nakanishi, Y., & Yokoyama, R. (2021). A time series model based on hybrid-kernel least-squares support

vector machine for short-term wind power forecasting. *ISA transactions*, *108*, 58–68.

Ding, Y. (2019). *Data science for wind energy*. CRC Press.

Elman, J. L. (1991). Distributed representations, simple recurrent networks, and grammatical structure. *Machine learning*, *7*, 195–225.

Essien, A., & Giannetti, C. (2020). A deep learning model for smart manufacturing using convolutional lstm neural network autoencoders. *IEEE Transactions on Industrial Informatics*, *16*, 6069–6078.

Fan, C., Wang, J., Gang, W., & Li, S. (2019). Assessment of deep recurrent neural network-based strategies for short-term building energy predictions. *Applied energy*, *236*, 700–710.

Fang, X., & Yuan, Z. (2019). Performance enhancing techniques for deep learning models in time series forecasting. *Engineering Applications of Artificial Intelligence*, *85*, 533–542.

Faruk, D. Ö. (2010). A hybrid neural network and arima model for water quality time series prediction. *Engineering applications of artificial intelligence*, *23*, 586–594.

Gajamannage, K., Park, Y., & Jayathilake, D. I. (2023). Real-time forecasting of time series in financial markets using sequentially trained dual-lstms. *Expert Systems with Applications*, *223*, 119879.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press.

Hanin, B. (2018). Which neural net architectures give rise to exploding and vanishing gradients? *Advances in neural information processing systems*, *31*.

He, W. (2017). Load forecasting via deep neural networks. *Procedia Computer Science*, *122*, 308–314.

Hochreiter, S., Bengio, Y., Frasconi, P., Schmidhuber, J. et al. (2001). Gradient flow in recurrent nets: the difficulty of learning long-term dependencies.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*, 1735–1780.

Khaldi, R., El Afia, A., Chiheb, R., & Tabik, S. (2023). What is the best rnn-cell structure to forecast each time series behavior? *Expert Systems with Applications*, *215*, 119140.

Kong, W., Dong, Z. Y., Hill, D. J., Luo, F., & Xu, Y. (2017). Short-term residential load forecasting based on resident behaviour learning. *IEEE Transactions on Power Systems*, *33*, 1087–1088.

Kuan, L., Zhenfu, B., Xin, W., Xiangrong, M., Honghai, L., Wenxue, S., Zijian, Z., & Zhimin, L. (2017). Short-term chp heat load forecast method based on concatenated lstms. In *2017 Chinese Automation Congress (CAC)* (pp. 99–103). IEEE.

Livieris, I. E., & Pintelas, P. (2022). A novel multi-step forecasting strategy for enhancing deep learning models' performance. *Neural Computing and Applications*, *34*, 19453–19470.

Livieris, I. E., Stavroyiannis, S., Pintelas, E., & Pintelas, P. (2020). A novel validation framework to enhance deep learning models in time-series forecasting. *Neural Computing and Applications*, *32*, 17149–17167.

Makridakis, S., & Hibon, M. (1997). Arma models and the box–jenkins methodology. *Journal of forecasting*, *16*, 147–163.

Mancuso, P., Piccialli, V., & Sudoso, A. M. (2021). A machine learning approach for forecasting hierarchical time series. *Expert Systems with Applications*, *182*, 115102.

Marino, D. L., Amarasinghe, K., & Manic, M. (2016). Building energy load forecasting using deep neural networks. In *IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society* (pp. 7046–7051). IEEE.

de Mattos Neto, P. S., Cavalcanti, G. D., de O Santos Júnior, D. S., & Silva, E. G. (2022). Hybrid systems using residual modeling for sea surface temperature forecasting. *Scientific Reports*, *12*, 1–16.

Müller, D., Soto-Rey, I., & Kramer, F. (2022). An analysis on ensemble learning optimized medical image classification with deep convolutional neural networks. *Ieee Access*, *10*, 66467–66480.

O'Shea, K., & Nash, R. (2015). An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, .

Panigrahi, S., & Behera, H. S. (2017). A hybrid ets–ann model for time series forecasting. *Engineering applications of artificial intelligence*, *66*, 49–59.

Pintelas, E., Livieris, I. E., Stavroyiannis, S., Kotsilieris, T., & Pintelas, P. (2020). Investigating the problem of cryptocurrency price prediction: a deep learning approach. In *Artificial Intelligence Applications and Innovations: 16th IFIP WG 12.5 International Conference, AIAI 2020, Neos Marmaras, Greece, June 5–7, 2020, Proceedings, Part II 16* (pp. 99–110). Springer.

Rangapuram, S. S., Seeger, M. W., Gasthaus, J., Stella, L., Wang, Y., & Januschowski, T. (2018). Deep state space models for time series forecasting. *Advances in neural information processing systems*, *31*.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, *323*, 533–536.

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

Sarkar, M. R., Anavatti, S. G., Dam, T., Pratama, M., & Al Kindhi, B. (2023). Enhancing wind power forecast precision via multi-head attention transformer: An investigation on single-step and multi-step forecasting. In *2023*

International Joint Conference on Neural Networks (IJCNN) (pp. 1–8). IEEE.

Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-K., & Woo, W.-c. (2015). Convolutional lstm network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*, *28*.

Singleton (2021). Electricity demand data and solar generation data from Plymouth. UK. URL: `https://doi.org/10.5281/zenodo.5500457`. doi:`10.5281/zenodo.5500457`.

Tan, M., Yuan, S., Li, S., Su, Y., Li, H., & He, F. (2019). Ultra-short-term industrial power demand forecasting using lstm based hybrid ensemble learning. *IEEE transactions on power systems*, *35*, 2937–2948.

Wang, H., Lei, Z., Liu, Y., Peng, J., & Liu, J. (2019). Echo state network based ensemble approach for wind power forecasting. *Energy Conversion and Management*, *201*, 112188.

Whittle, P. (1952). Tests of fit in time series. *Biometrika*, *39*, 309–318.

Yan, L., Elgamal, A., & Cottrell, G. W. (2013). Substructure vibration narx neural network approach for statistical damage inference. *Journal of Engineering Mechanics*, *139*, 737–747.

Zhang, S., Chen, Y., Zhang, W., & Feng, R. (2021). A novel ensemble deep learning model with dynamic error correction and multi-objective ensemble pruning for time series forecasting. *Information Sciences*, *544*, 427–445.