Archives of Environmental Protection
Vol. 51no. 1 pp. 103–115

PL ISSN 2083-4772
DOI:10.24425/aep.2025.153754

Neural Network Prediction Model – Applied to U.S. Industrial Greenhouse Gas Emissions

Shih-Hsien Tseng*, Chia-Hsuan Wang, Thi Ha Trang Duong

National Taiwan University of Science and Technology, Taiwan

* Corresponding author's e-mail: shtseng@mail.ntust.edu.tw

Keywords: deep learning, greenhouse gas emission, GRU, RNN, transformer, time series prediction model

Abstract: This study explores the use of deep learning neural network models for predicting greenhouse gas emissions, focusing on small-sample time-series data sets, an area with limited prior research. It utilizes Recurrent Neural Networks (RNNs), Long Short-Term Memory Networks (LSTMs), Gated Recurrent Units (GRUs), and Transformers combined with Genetic Algorithms to forecast CO_2 emissions from industrial sources in Texas, a major contributor to U.S. greenhouse gas emissions. The analysis is based on the Environmental Protection Agency's (EPA) "Inventory of U.S. Greenhouse Gas Emissions and Sinks" dataset, spanning 1990 to 2020. The results indicate that LSTM and Transformer models are particularly effective, with LSTM outperforming Transformers in computational efficiency by 6.97 times. These findings highlight the potential of LSTM and Transformer models as accurate and stable tools for predicting CO_2 emissions in small-sample time-series data, offering valuable insights for future research and policy development in environmental management.

Introduction

Since the beginning of the Industrial Revolution in the 18th century, fast economic expansion and population growth have resulted in a steady increase in global greenhouse gas emissions, intensifying the negative impacts of climate change. Consequently, reducing greenhouse gas emissions has become a central focus of global initiatives aimed at energy conservation and environmental sustainability. Accurate forecasting of carbon emissions is essential to provide policymakers with the insights needed to mitigate the greenhouse effect (Hsu et al., 2022; Yin et al., 2022).

In past research, traditional time series forecasting and analysis methods, such as the ARIMA model, exponential smoothing method, moving average method, and trend analysis, were commonly used for time series data. Although machine learning has flourished with the advent of the big data era, how to use neural network models to predict small-sample time-series data sets has yet to be fully explored, especially for industrial greenhouse gas (CO_2) emissions. The primary purpose of this research is to establish a time-series data prediction model that can accurately predict future greenhouse gas emissions. We will compare the prediction performance of different deep learning neural network models RNN, LSTM, GRU, and Transformer on small-sample time series data sets, adjust the model parameters, and further apply Genetic Algorithms for optimization.

It is common to use time differencing before processing time series data. The value of each field in the data set is treated as a difference according to time, as introduced by Hyndman and Athanasopoulos (2018). The overall level of the time series can be eliminated over time, thereby stabilizing the mean value of the time series and eliminating or reducing the influence of trend and seasonality. The time series forecasting method is a technique with a long history of development. Traditional forecasting methods encompass the ARIMA model, exponential smoothing, moving averages, trend analysis, and more. Sen et al. (2016) employed ARIMA to predict energy consumption and greenhouse gas emissions in India.

In recent years, there has also been much literature on applying time-series analysis data to machine learning technology. For example, Fang et al. (2021) compared four machine learning models - Multiple Linear Regression (MLR), SVM, Back Propagation Artificial Neural Network (BPNN), and ground-based LiDAR Random Forest (RF) to predict PM2.5 concentrations in Beijing and concluded that the RF model had the highest accuracy. This method offers the benefit of both strong predictive performance and a relatively simple model structure (Szeląg et al., 2017). Sun and Liu (2016) published the least squares support vector machine (LSVM) method for the study of CO_2 emissions in China and showed the LSVM method in their experiment. The prediction results are better than the logistic model, BPNN, and GM (1, 1) model.

In addition, some technicians build models based on a small amount of data and mathematical models. For example, Şahin (2019) published a paper using the linear and nonlinear rolling metabolic gray model in Turkey.

With the rapid development of deep learning and neural networks in modern times, Rumelhart et al. (1986) published the concept of RNN back-propagation for neuron-type networks. It identifies important task features and captures regularities through internal hidden neurons by adjusting connection weights to minimize the difference between actual and desired outputs (Rumelhart et al., 1986). However, RNN has a serious problem of vanishing gradient problem. During back-propagation, the gradient gradually becomes smaller as the time step increases, causing earlier time steps to have less influence on the update of the model parameters.

Hochreiter and Schmidhuber (1997) proposed LSTM to improve RNN's lack of long-term memory. Compared with traditional RNN, LSTM can better handle long sequence data because it introduces memory units and a gating mechanism, which can better capture long-term dependencies in sequence data (Hochreiter & Schmidhuber, 1997).

Riekstin et al. (2020) used the LSTM of the RNN series model to conduct time series research related to GHG emissions. The experimental results confirmed that the performance of LSTM is better than that of Support Vector Regression (SVR).

Compared to LSTM, the structure of GRU is simpler, as it only consists of two gates - the reset gate and the update gate. This not only reduces the number of parameters in the model but also makes it more straightforward to train.

AlKheder and Almusalam (2022) used the Intergovernmental Panel on Climate Change (IPCC) and the United States Environmental Protection Agency (USEPA) data set in the 2022 study, including Kuwait electricity production, shares of the annual energy resources, and carbon dioxide emissions, and compared SVM, ANN, and Deep Learning. For the prediction of carbon dioxide emissions, the experimental results show that DL prediction is the best.

In 2017, Vaswani et al. proposed attention mechanisms, which use a neural network model that is completely different from the traditional model design of the RNN series. Sequence modeling is realized through attention mechanisms, which can capture associations at sequence positions at any distance (Vaswani et al., 2017). Through the multi-head self-attention mechanism, the relationship between different positions in the sequence can be better captured. Here Transformer also leads the neural network model to the next stage.

A Genetic Algorithm is a heuristic algorithm that simulates biological evolution in nature. Simulating the mechanism of inheritance, exchange, and survival of the fittest gradually evolves to reach the optimal solution. When the Genetic Algorithm is applied in machine learning, the independent variables in the experiment can be within a reasonable range and gradually optimized through the genetic rules, so that the experimental independent variable combination can achieve the most suitable solution in the experimental environment. In the past literature, a well-known example of Genetic Algorithm is the Traveling Salesman Problem (TSP), published by Potvin (1996). Given the number of cities and the coordinates of each city on the map, solve the problem of starting from a certain city, passing through the shortest distance and path it takes for all other cities to return to the original city after exactly one time (Potvin, 1996). In a detailed introduction to GA, Mitchell (1998) showed how the Genetic Algorithm is applied to optimization problems, such as parameter adjustment, feature selection, and model structure design.

Another widely used parameter optimization method is Grid Search. The method works by systematically searching all possible parameter combinations to ensure the likelihood of finding the best solution. However, the Grid Search method consumes much computing resources and time for models with many parameter combinations. A study by Alibrahim and Ludwig (2021) compared three approaches, including Genetic Algorithm, Grid Search, and Bayesian Algorithm hyperparameter optimization for the same problem. The research results show that the Genetic Algorithm performs excellently in similar experimental results.

Our research is crucial as it fills a gap in prior studies that have less frequently explored small-sample data sets. Given the constraints of data availability in forecasting, focusing on small data sets is a practical approach to tackling real-world issues. Developing a prediction model tailored for small data sets can also pave the way for innovative solutions to problems arising from data scarcity. By specifically delving into neural network models for industrial CO_2 emissions with limited data, our study seeks to improve forecast reliability, thereby transforming how organization use small data sets for informed decisions. The projected CO_2 emissions can serve as a guide in shaping policies, devising conservation strategies, and implementing effective emission reduction tactics, considering the significant impact of industrial emissions on the overall emissions landscape.

Methodology

Research Object

The Energy Information Administration (EIA) reports that Texas is the primary region for crude oil and natural gas production in the United States and has the highest level of industrial energy consumption (EIA, 2022).

Based on historical data from 1990 to 2020, Texas has consistently exhibited the highest average annual greenhouse gas emissions among all states in the US over the past 30 years. Consequently, this study has selected Texas as the primary research focus.

According to EIA statistics, the production of crude oil in the United States has continued to rise in recent years. From the perspective of the industrial sector, including refineries and petrochemical plants in Texas, the percentage of carbon dioxide emissions generated by crude oil production is as high as more than half of the total emissions in Texas, and the overall energy consumption is as high as 23% of the country.

It can be seen that Texas Field's crude oil production averaged 1,104.11 thousand barrels per day from 2001 to 2010. Between 2018 and 2022, oil production steadily increased, reaching an average of 4,833.93 thousand barrels per day.

In the part of natural gas extraction, Texas Natural Gas Gross withdraw that the production continues to increase, and these production increase factors will cause Texas to become the state with the highest carbon emissions in the industrial sector. This study believes there is a need for an accurate prediction, and the continuous supervision and control model helps decision-making units on carbon reduction targets achieve a balance between crude oil and natural gas production and greenhouse gas emissions control.

Data set

In the use of data sets, this study uses the public data set of Inventory of U.S. Greenhouse Gas Emissions and Sinks on the website of the Environmental Protection Agency (EPA) in the United States and selects the industry sector as the primary official public data set source in Greenhouse Gas Inventory Data Explorer and take US CO_2 emissions as the primary research category.

Texas is an important city for oil and natural gas production in the United States, the main sources of greenhouse gas emissions come from "Fossil fuel combustion: carbon dioxide" and "Natural gas and petroleum systems". In the greenhouse gas emission time-series data set, a total of greenhouse gas emission values for each year from 1990 to 2020 are included, including 10 different types of CO_2 emissions, such as Fossil fuel combustion: carbon dioxide, natural gas, and petroleum systems, other industrial categories, chemical industry, mineral industry, production and use of fluorinated gases, metal industry, fossil fuel combustion: other greenhouse gases, coal mining, etc. Carbon dioxide emissions are calculated in Million Metric Tons (MMT).

In terms of data set allocation, since this study uses a small sample data set and adheres to rigorous research methods, it is still trying to divide the data into a training set, verification set, and testing set based on limited data set resources at 60:20:20 proportion. Therefore, following pre-processing of the data set, there will be 15 training sets, 5 validation sets, and 5 test sets to serve as the research resources for this experiment.

In the time series window setting, this study uses a window size = 5. For example, data from the first year to the fifth year is used to predict the total carbon dioxide emissions for the sixth year. The training data employs a sliding window concept, shifting the time pane with each training set. For instance, data from 1991-1995 is used to predict carbon dioxide emissions for 1996, and data from 2005-2009 is used to predict emissions for 2010, resulting in a total of 15 data sets.

To validate the sliding time pane, data from 2006-2010 is used to predict carbon dioxide emissions for 2011, and data from 2010-2014 is used to predict emissions for 2015, yielding a total of 5 data sets. Similarly, for the test data sliding window concept, data from 2011-2015 is used to predict carbon dioxide emissions for 2016, and data from 2015-2019 is used to predict emissions for 2020, also resulting in a total of 5 data sets.

Research Methods

Since the data set type is time-series, there are two main parts in Data preprocessing. Time Series Differencing and Min-max Normalization are the techniques we will mainly use in data preprocessing.

The backward shift operator first difference can be written as Eq. (1):

$$y'_t = y_t - y_{t-1} \tag{1}$$

Where y_t is the value of time t, y_{t-1} is the last period before time t, y'_t is time after differencing.

Min-max normalization is a method commonly used for data standardization, which can help to scale features to a specified range. The commonly used scaling range is usually in the range of [-1, 1] or [0, 1]. Taking this study as an example, after the column data of different types of CO_2 emissions are pre-processed, Min-max normalization is performed on the data. In order to scale different types of value features to the same range, it can also avoid excessive influence of data with different eigenvalues on the model, so that the data retains the original distribution of eigenvalues in each category.

The formula can be written as Eq. (2) follows:

$$m(i) = \frac{X(i) - Xmin}{Xmax - Xmin'} \tag{2}$$

where $m(i)$ is the normalized value of $X(i)$, $X(i)$ is the i^{th} value in the X-type data, $Xmax$ is the largest value in the X-type data, and $Xmin$ is the smallest value in the X-type data (Fig. 1).

Neural Network Algorithm

In our research, four kinds of neural network model algorithms are used for implementation. RNN, LSTM, GRU, and Transformer are implemented on the small-sample time-series data set in this study, trying to compare the performance of each model with the best performance.

Recurrent Neural Network (RNN) is a deep learning neural network model that is often used to process sequential data, such as time series, speech signals, natural language

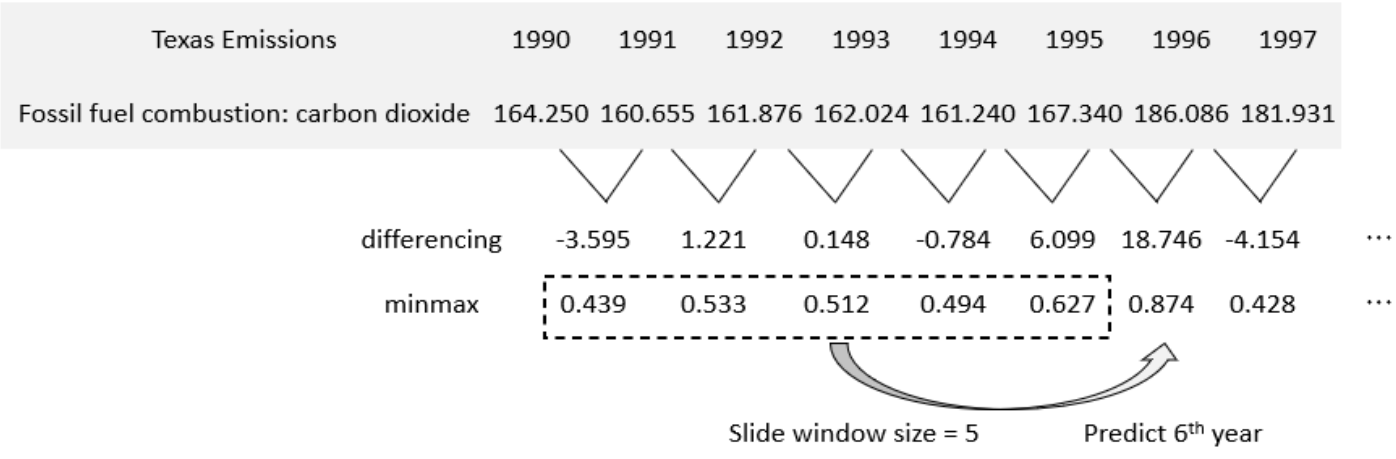

Fig. 1. Data differencing and min-max scaling

processing (NLP), and other datasets. In the traditional neural network, each input is independent, and RNN is characterized by its memory function, which can be used for time series analysis by referring to the data of the previous and subsequent time points through the structure of the recurrent cyclic neuron.

Long Short-Term Memory (LSTM) is a special form of RNN. Compared with traditional RNN, LSTM is characterized by using three different control valves in the neural unit, namely a forget gate, an input gate, and an output gate. These gates can control the flow of information updated with time, and then through the forget gate, an input gate decides whether to update the currently stored information.

Gated Recurrent Unit (GRU) is a type of neural network designed for modeling sequence data. Similar to LSTM, GRU addresses the issues of gradient vanishing and exploding in RNN. It achieves this through a gating mechanism, which effectively captures long-term dependencies. Moreover, GRU boasts a relatively simple structure, making it easy to train and adjust.

Transformer is a popular neural network model in recent years. It is a sequence model based on the attention mechanism.

The main structure is composed of two parts: encoder and decoder. The encoder is responsible for receiving the input sequence and converting it into a series of feature vectors through a multi-layer self-attention mechanism. The decoder receives the output of the encoder and converts the feature vector into a sequence through a multi-layer self-attention mechanism as the predicted output.

Neural Network Optimization Technology

Our research utilizes the Sigmoid function, with a preset Momentum reference value set to 0.9. The Dropout rate defaults to 0.5. The underlying concept is randomly deactivating neurons during training, ensuring that only a portion of the network structure is updated at each iteration. This approach helps prevent overfitting, which can be especially problematic with small sample sizes during training.

Considering resource constraints, model complexity, and the characteristics of our small-sample data set, we chose a batch size of 5. A smaller batch size helps mitigate model overfitting. Additionally, we consistently apply a dropout rate of 0.5 for subsequent experiments.

For other hyperparameters, we employ the Genetic Algorithm. Specifically, we focus on three parameters, each representing a gene; Learning Rates (lr): We predefine seven learning rates: [0.01, 0.005, 0.001, 0.0005, 0.0001, 0.00005, 0.00001]; Hidden Layers: We explore the number of hidden layers within the range of 1 to 10; Hidden Layer Neurons: We consider the number of neurons per hidden layer, ranging from 1 to 20.

Hyperparameter Selection-Genetic Algorithm (GA)

Genetic Algorithm Parameter Initial Setting

In our research, a Genetic Algorithm is used for hyperparameter selection, and the following are used as initial settings. Chromosome length=3, representing the three genes of the learning rate, num_layers, and hidden size.

The population is set to 100 randomly generated individuals in the first generation. Max_iteration is set to 100, represents the number of generations to be set, and this study is set to reproduce 100 generations of offspring.

Genetic Algorithm Fitness Calculation Strategy

In the Genetic Algorithm model, 100 sets of chromosomes are randomly selected in the population as the first generation. The first generation of parent chromosomes are randomly selected gene learning rate, num_layers, and hidden size, which become the parameters of the training performance of each set of chromosomes.

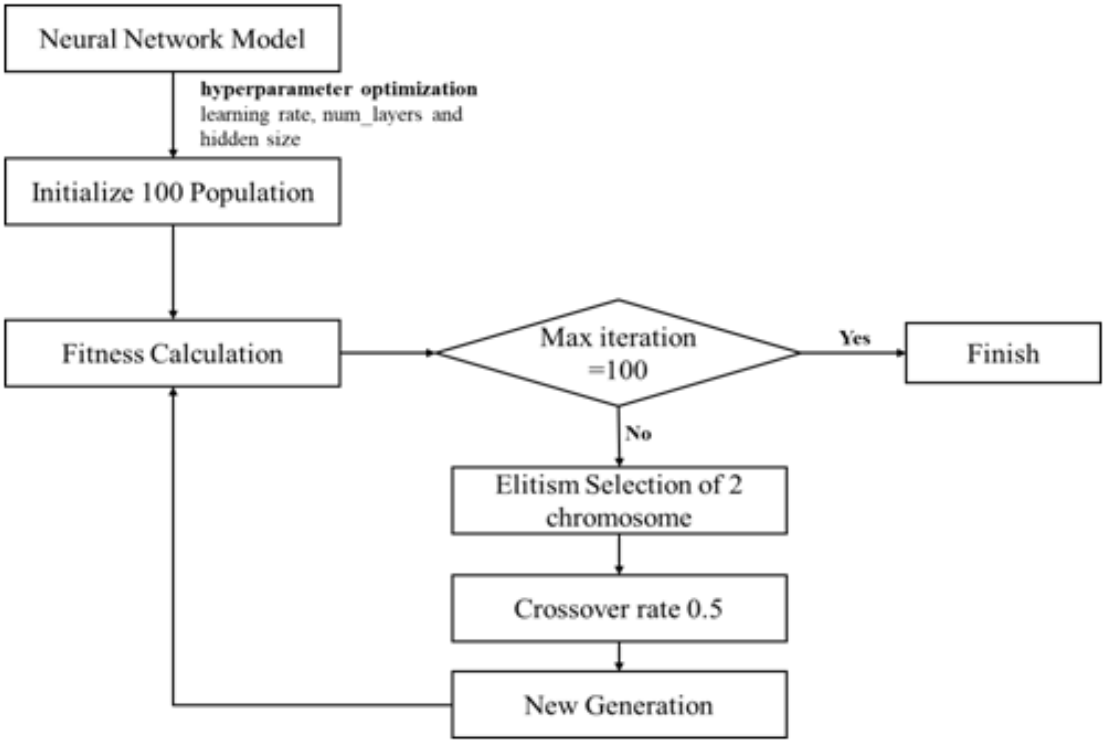

Fig. 2. Neural network model combined with Genetic Algorithm

The smaller the MSE in the verification set, the better. Here is defined as the reciprocal of the average MSE of the i^{th} chromosome (Eq. 3), which is the block of Fitness calculation in Fig. 2. Next, in the elite selection mechanism, use the $fit(i)$ of each gene individual in this generation to add up the denominator, the $fit(i)$ of the individual gene is the numerator and $FIT(i)$ represents the value of each chromosome in the elite selection mechanism probability (Eq. 2). The objective function is the maximum value of FIT, which means that better genes have a higher probability of being selected and passed on to the next generation.

The performance of chromosome *(i)* can be written as Eq. (3):

$$fit(i) = \frac{1}{\left(\frac{\sum MSE\_valid}{num\_seed}\right)'} \tag{3}$$

where $i = 1, 2,...$ number of populations, MSE_valid represents the MSE between the predicted result and the actual value of the three gene parameter combinations. The number of random seeds in the *num_seed* experiment, which represents the number of random experiments performed on chromosomes, and $fit(i)$ represents the reciprocal of the $(i)^{th}$ gene individual after the average MSE of the validation set, as the performance result of the gene individual.

The probability of each chromosome in the elite selection mechanism is defined as Eq. (4):

$$FIT(i) = \frac{fit(i)}{\sum fit(i)'} \tag{4}$$

where $i = 1, 2,...$ number of populations, $FIT(i)$ represents the probability of each chromosome in the elite selection mechanism.

Genetic Algorithm Selection and Crossover

"Selection" means ~~are~~ selected according to the performance of the population chromosome, and the selected chromosome individuals represent (s1, s2), respectively. The higher the fit score, the higher the probability 'prob' of being selected. "Crossover" means randomly selecting the gene fragments of excellent individuals (s1, s2) as exchange genes for the next generation of reproduction (Fig. 3). Select the crossover point as the cutting point of outstanding individual DNA fragments, and the probability of random selection is $p = 0.5$.

Merge and extract the fragments of excellent chromosomes (s1, s2) to become new chromosomes, and the resulting new generation of chromosomes will replace the chromosome with the lowest fit score in the original parent generation to become a new generation of population. (Fig. 4).

Then proceed to the next-generation selection process, select the next generation of excellent chromosomes (s1, s2) according to the level of 'prob', and perform crossover gene

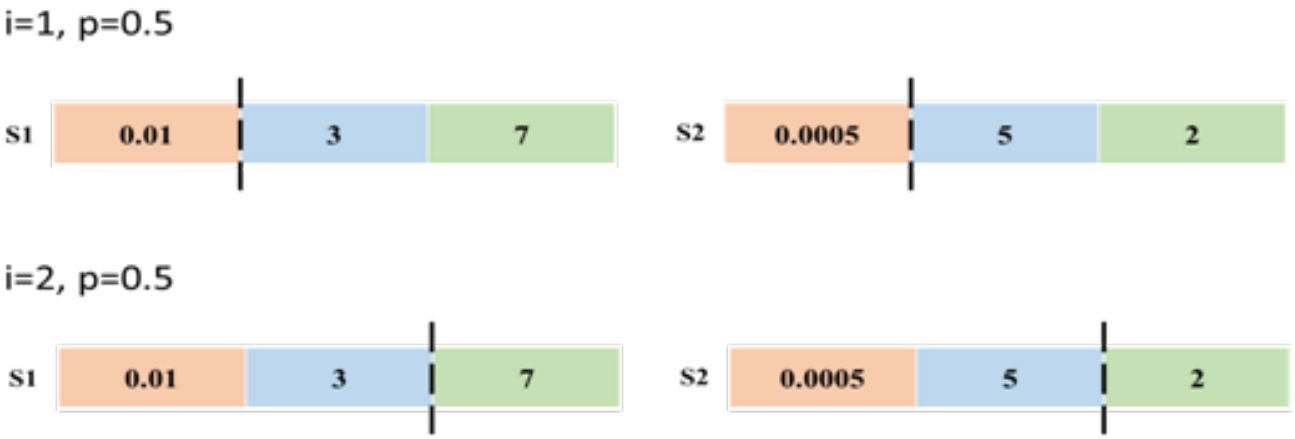

Fig. 3. Select crossover point between first and second fragment, or between second and third fragment

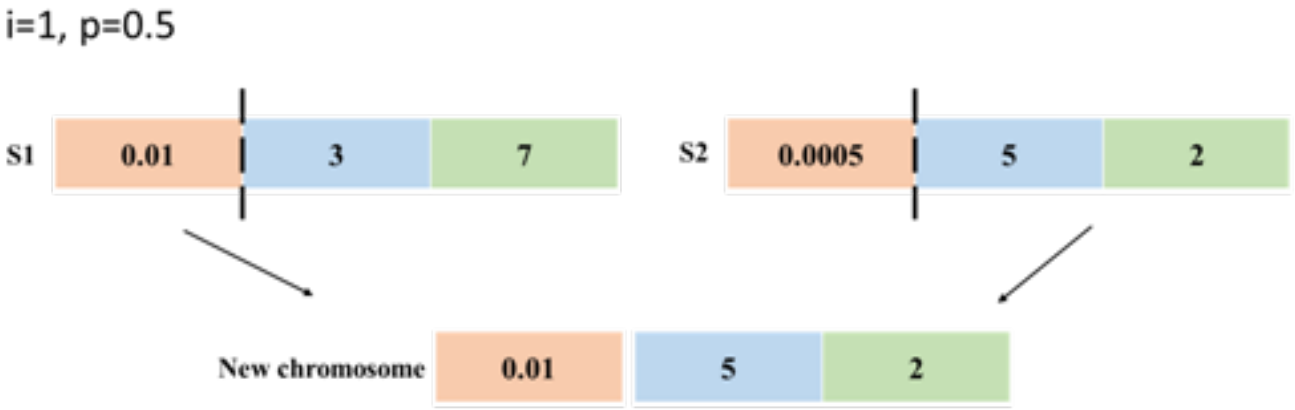

Fig. 4. Crossover process to a new chromosome

fragment exchange to replace the parent chromosome with the lowest fit score. This cycle will repeat *max_iter* = 100, leaving the best 100 individuals. In our research, the highest fit score among the selected results means that the average result of *MSE_valid* after *num_seed* (times) is the lowest, which means that the performance under this gene combination (the hyperparameter combination of this study) is the best.

In addition, this study did not use the mutation mechanism in the Genetic Algorithm. Since we have set 100 chromosomes in the first-generation population design, in terms of expected value, the seven learning rates can draw an average of 14.29 times. In the setting of 10 kinds of "number of layers", each can draw 10 times on average. In the setting of 20 kinds of "number of hidden sizes", each can draw five times on average. After repeating the Elitism Selection for 100 generations, selection and crossover are performed. This method can quickly converge and find an excellent parameter combination easily.

Evaluation Metrics

Our research focuses on three evaluation indicators: *MSE*, *MAE* and *MAPE*. In the process of model training, this research focuses on the *MSE* of the verification set to select the model and uses *MAE* and *MAPE*% as the assistant to evaluate the effectiveness of model training and the basis for selecting parameters.

Running Time

In addition to comparing the performance of the data set *MSE* in the four neural network models, this study also compared the calculation time of the Genetic Algorithm using the same equipment, environment, and computing resources according to the algorithm characteristics of different models. We attempt to find the most efficient algorithmic models for predicting CO_2 emissions research.

Model Selection

We use RNN, LSTM, GRU, and Transformer to conduct 10 different random seed experiments at the validation stage, respectively, to confirm whether the optimal solution stays in the local optimal as the basis for model selection. And 100 random seed experiments were performed with the best parameter combination (learning rate, num_layers, and hidden size) of each model to determine the stability of the model and conduct subsequent verification analysis of experimental results.

Kolmogorov-Smirnov (KS) Normality Test

The Kolmogorov-Smirnov (KS) test is a statistical test used to determine whether a data set follows a particular distribution, most commonly a normal distribution. As a nonparametric test, it does not require any assumptions about the underlying distribution of the data to assess normality. Therefore, due to its robust properties, we use the Kolmogorov-Smirnov (KS) test for normality testing of the samples.

Experimental Results and Discussion

Experimental Results

After 10 times, 30 times, and 100 times of random experiments, the experimental results of the four models TransformerGA, RNNGA, LSTMGA, and GRUGA, are detailed below. The parameter combinations of the best-performing models and the comprehensive performance results of the models are listed.

In the GA experiment, the search range for feasible solutions was set to an initial population size of 100 and a maximum of 100 iterations. The learning rate varied between 0.01, 0.005, 0.001, 0.0005, 0.0001, 0.00005, and 0.00001. The number of layers ranged from 1 to 10, and the hidden size ranged from 1 to 20.

Experimental Results in Test Data Set for 10 Times

Based on the experimental results of 10 Valid data sets, the best parameter combinations of the RNNGA model performance learning rate, number of layers, and hidden size are shown in Table 1. They are 0.0001, 2, 8, respectively; the best parameter combinations of the LSTMGA model learning rate, number of layers, and hidden size are 0.01, 2, 9, respectively; the best parameter combinations of the GRUGA model learning rate, number of layers, and the hidden sizes are 0.0005, 1, and 9 respectively; the best parameter combinations of TransformerGA model learning rate, number of layers, and hidden size are 0.01, 10, 12 respectively.

According to the experimental results in Table 2, TransformerGA performed better than other models in the verification set *MSE*. *MSE* found the best solution at 78.44. Although in the process of finding the optimal solution, more layers and hidden sizes are required than other neural network models, better results can still be found. In addition, the average result of GRUGA in the 10-time validation set *MSE* is the best at 100.31, and the standard deviation is the smallest at 12.17, showing that the model is relatively stable in the 10-time experimental results. It is worth noting that LSTMGA and TransformerGA use a learning rate of 0.01 to find a better optimal solution under the same conditions. TransformerGA and LSTMGA are sequence-based models and perform better in capturing temporal dependencies within the sequence. On the other hand, GRUGA and RNNGA may need more modeling capability for long-term dependencies within the sequence, which could lead to slower convergence or failure to converge, especially with high learning rates. Overall, the experimental results from 10 runs show that LSTMGA and TransformerGA perform better in sequence prediction and are the top-performing models in the experiment.

Experimental Results in Test Data Set for 30 Times

Following the results of 10 random experiments on the validation set, we combined the best parameters of each

Table 1. The best parameter combination of 10 valid dataset experiments

Model	lr	num_layers	hidden_size
RNNGA	0.0001	2	8
LSTMGA	0.01	2	9
GRUGA	0.0005	1	9
TransformerGA	0.01	10	12

Table 2. Result of valid dataset for 10 random experiments

Model	*num_seed*	*Min_MSE*	*Max_MSE*	*MSE±Std*	*MAE±Std*	*MAPE±Std (%)*
RNNGA	10	81.34	171.33	102.12 ± 27.92	8.50 ± 0.80	2.71 ± 0.26
LSTMGA	10	83.32	159.70	108.31 ± 24.60	8.22 ± 0.65	2.61 ± 0.22
GRUGA	10	80.97	111.97	**100.3l ± 12.17**	**7.93 ± 0.22**	**2.52 ± 0.07**
TransformerGA	10	**78.44**	171.76	127.77 ± 34.43	8.83 ± 1.41	2.82 ± 0.46

Table 3. Result of test dataset for 30 random experiments

Model	*num_seed*	*Min_MSE*	*Max_MSE*	*MSE±Std*	*MAE±Std*	*MAPE±Std (%)*
RNNGA	30	286.76	489.02	334.57 ± 58.89	15.60 ± 1.29	4.61 ± 0.34
LSTMGA	30	286.80	370.22	**300.54 ± 17.39**	15.50 ± 0.58	4.57 ± 0.15
GRUGA	30	286.76	491.48	325.98 ± 62.06	15.51 ± 1.26	4.59 ± 0.34
TransformerGA	30	**225.71**	**355.53**	**303.01 ± 30.29**	15.52 ± 0.98	4.57 ± 0.28

Table 4. Results of test dataset for 100 random experiments

Model	*num_seed*	*Min_MSE*	*Max_MSE*	*MSE±Std*	*MAE±Std*	*MAPE±Std (%)*
RNNGA	100	286.08	593.00	330.14 ± 60.20	15.66 ± 1.26	4.62 ± 0.34
LSTMGA	100	286.77	**370.22**	**304.42 ± 16.09**	15.63 ± 0.58	4.61 ± 0.15
GRUGA	100	286.76	631.73	345.40 ± 75.40	15.91 ± 1.56	4.69 ± 0.42
TransformerGA	100	**186.19**	415.34	**305.13 ± 34.54**	15.62 ± 0.91	4.60 ± 0.26

model, conducted 30 experiments respectively, and obtained the following observations with the results of the test set. From Table 3, TransformerGA MSE in the test data set, the minimum value is 225.71, and there is a clear gap with RNNGA, LSTMGA, and GRUGA. In the evaluation of MSE, LSTMGA, and TransformerGA have similar results, which are 300.54 ± 17.39 and 303.01 ± 30.29, respectively. In the evaluation results of MAE and $MAPE$ (%), the four models did not exhibit significant differences. If we only focus on the minimum values of MSE, LSTMGA and TransformerGA will perform best. It can be seen that the test set distribution of LSTMGA and TransformerGA is concentrated, and the median and mean positions are very close. Compared with GRUGA and RNNGA, the distribution is uneven, and there are more outlier values (* symbol). RNNGA has two outliers, LSTMGA and TransformerGA each have one outlier, and GRUGA has four outliers. The length of the line extending outward from the box-and-whisker plot can also be seen as the degree of model variation, and the most variable degree is the RNNGA model. Our research believes that LSTM and TransformerGA have shown a relatively stable trend in 30 experiments.

Experimental Results in Test Data Set for 100 Times

Following the results of 10 random experiments, we used the best parameters of each model, performed 100 experiments respectively, and obtained the following observations with the results of the test set.

From Table 4, TransformerGA in the test set MSE, the minimum value can be found to be 186.19, and there is a clear gap with RNNGA, LSTMGA, and GRUGA. In the evaluation of MSE, LSTMGA and TransformerGA have similar results, both of which belong to models with higher accuracy, which are 304.42 ± 16.09 and 305.13 ± 34.54, respectively. In the evaluation results of MAE and $MAPE$ (%), the four models did not show significant differences. If we only focus on the minimum value of MAE and $MAPE$, TransformerGA will perform best.

According to the experimental results of this study, the average performance of LSTMGA in the 100-time test set MSE is relatively stable, and the standard deviation of the LSTMGA model in the test set is only 16.09, showing that the model is relatively stable. In addition, it can be seen from the box-and-whisker diagram that the MSE distribution of the test set of

Table 5. Time series prediction and actual value comparison table of neural network model

Time Series Forecast and Actual Value			
Model	**LSTMGA**	**TransformerGA**	**Actual value**
valid_2011	291.97	289.27	287.60
valid_ 2012	295.31	295.67	304.06
valid_2013	311.76	310.57	323.70
valid_2014	331.41	327.82	320.05
valid_2015	327.76	314.96	321.64
test_2016	329.35	321.14	312.47
test_2017	320.17	312.70	321.36
test_2018	329.06	320.65	348.08
test_2019	355.78	348.41	360.11
test_2020	367.81	351.46	338.46

LSTMGA and TransformerGA is concentrated, the median and the average are close to each other, and there are few outliers, which further indicates that LSTMGA and TransformerGA are relatively stable models. Taking these results together, this study concludes that in this specific time series forecasting task, the LSTMGA and TransformerGA models perform well, especially in forecasting accuracy and stability (Fig. 5).

Prediction Graph Comparison of Four Models

In the experimental results of the previous stage, we observed that the LSTMGA and TransformerGA models performed well. Table 5 is the predicted value of the LSTMGA and TransformerGA models in valid data and test data. The last column is the actual value of Texas CO_2 emissions for comparison.

In Fig. 6, the time series prediction diagram of the neural network model is a data example, and it can be observed that LSTMGA, TransformerGA, and actual value show a trend in the same direction.

Sonata and Heryadi (2024) identified LSTM and Transformer models as particularly suitable for time series prediction, noting that LSTM performs better overall. Aligning with their findings, our study reaffirms that combining LSTM or Transformer models with Genetic Algorithms further enhances their performance,

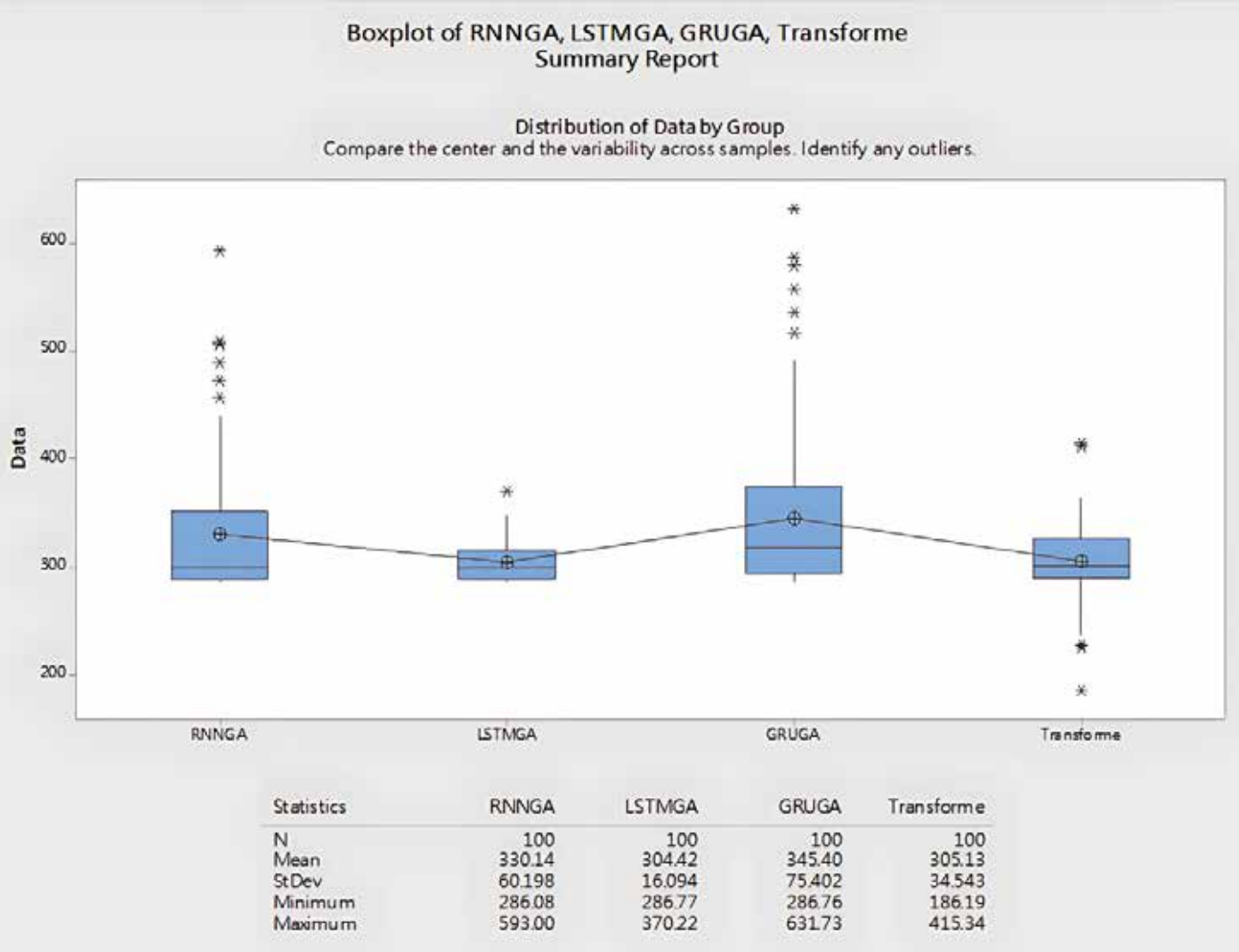

Fig. 5. The summary report of test dataset MSE for 100 experimental results

highlighting their ability to capture complex patterns and dependencies, even in data sets with limited sample sizes.

Kolmogorov-Smirnov (K-S) Normality Test

K-S Normality Test - RNNGA Model

In Fig. 7, the RNNGA model was used to conduct 100 non-repetitive experiments, the average MSE of the experimental results of the test data set was used as the sample, and the K-S Normality Test was conducted at the significance level $\alpha = 0.05$.

The test result shows a $p - value < 0.05$, rejecting the hypothesis that H_0 = samples follow a normal distribution. Our research supposes that the average MSE of the experimental results of the test data set in the RNNGA model does not follow a normal distribution.

K-S Normality Test - LSTMGA Model

In Fig. 8, the LSTMGA model was used to conduct 100 non-repetitive experiments, the average MSE of the experimental results of the test data set was used as the sample, and the K-S Normality Test was carried out at the significance level $\alpha = 0.05$.

The test result shows a $p - value < 0.05$, rejecting the hypothesis that H_0 = samples follow a normal distribution. Our research supposes that the average MSE of the experimental results of the test data set in the LSTMGA model does not follow a normal distribution.

K-S Normality Test - GRUGA Model

In Fig. 9, the GRUGA model was used to conduct 100 non-repetitive experiments, the average of the experimental results of the test data set was used as the sample, and the K-S Normality Test was carried out at the significance level $\alpha = 0.05$.

The test result shows a $p - value < 0.05$, rejecting the hypothesis that H_0 = samples follow a normal distribution. Our research supposes that the average MSE of the experimental results of the test data set in the GRUGA model does not follow a normal distribution.

K-S Normality Test - TransformerGA Model

In Fig. 10, the TransformerGA model was used to conduct 100 non-repetitive experiments, the average of the experimental results of the test data set was used as the sample, and the K-S Normality Test was carried out at the significance level $\alpha = 0.05$.

The test result shows a $p - value < 0.05$, rejecting the hypothesis that H_0 = samples follow a normal distribution. Our research supposes that the average MSE of the experimental results of the test data set in the TransformerGA model does not follow a normal distribution.

Based on the normality test results of the above RNNGA, LSTMGA, GRUGA, and TransformerGA models, none of them follow a normal distribution. However, this research model carries out 100 non-repetitive experiments, and according to Slutsky's theorem, the sampling distribution under large samples is still normal. Therefore, the next step is to use the Z test as the test data set MSE of RNNGA, LSTMGA, GRUGA, and TransformerGA models to test the difference between the mean of two independent populations.

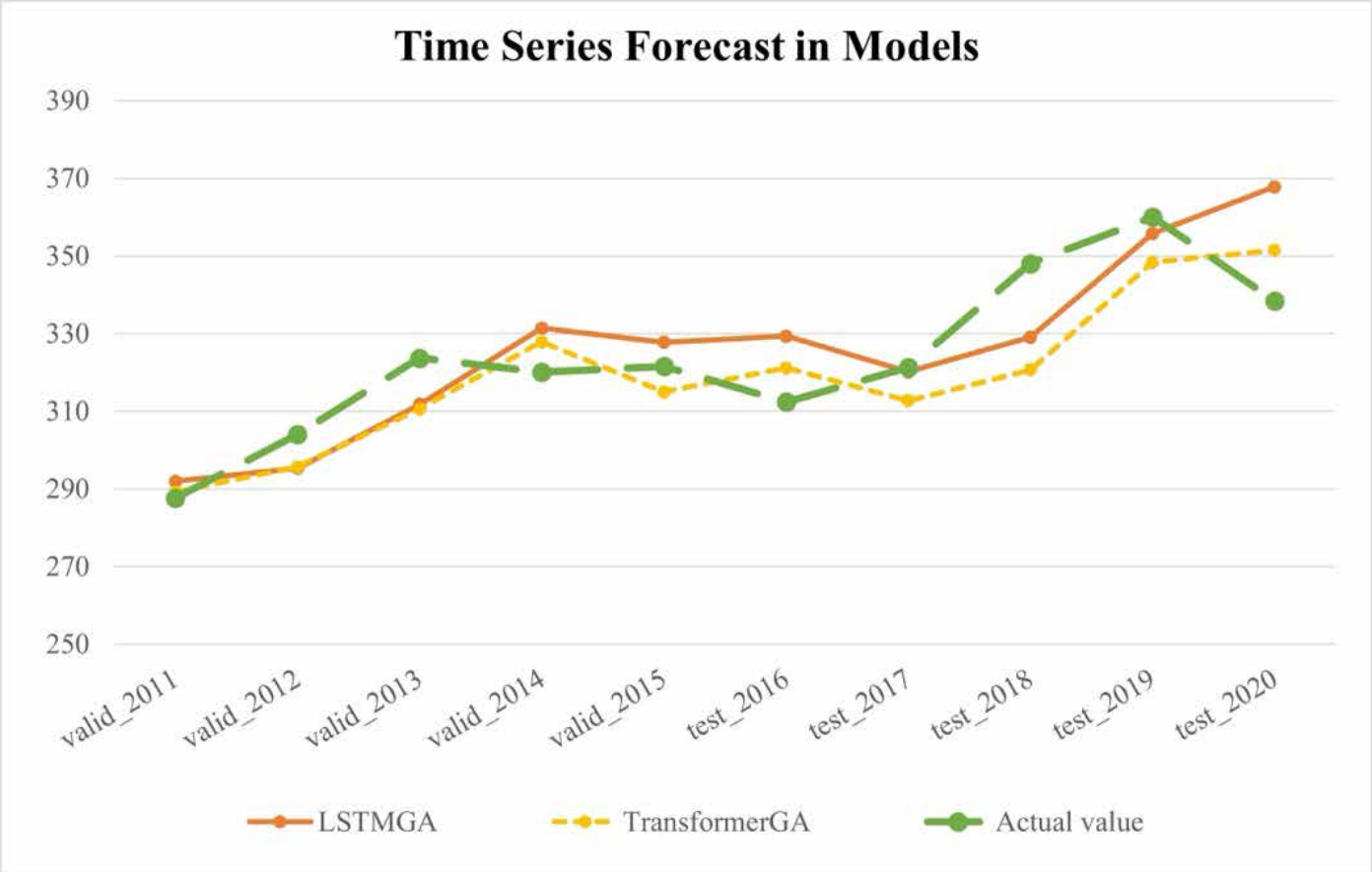

Fig. 6. Time series prediction diagram of neural network model

Z-Test

Z-Test for LSTMGA and TransformerGA

From the experimental data, we observed that the average of LSTMGA in the test set MSE is very similar to that of TransformerGA, and the variance of LSTMGA is smaller than that of TransformerGA. From the small degree of variation of the LSTMGA model, we can infer that the LSTMGA model is slightly more stable than the TransformerGA model. At the Z-test significance level $\alpha = 0.05$, the test result $p - value < 0.05$, do not reject H_0:$u1 = u2$. Statistically, we believe that there is no significant difference between the TransformerGA model MSE_test and the LSTMGA model MSE_test.

Z- Test for TransformerGA and RNNGA

Based on our experimental findings, we noticed that the mean squared error (MSE) of the TransformerGA in the test set is lower compared to that of the RNNGA, and the variance of the TransformerGA is significantly less than that of the RNNGA. At the Z-test significance level $\alpha = 0.05$, the test result $p - value < 0.05$, reject H_0:$u1 = u2$. Statistically, there

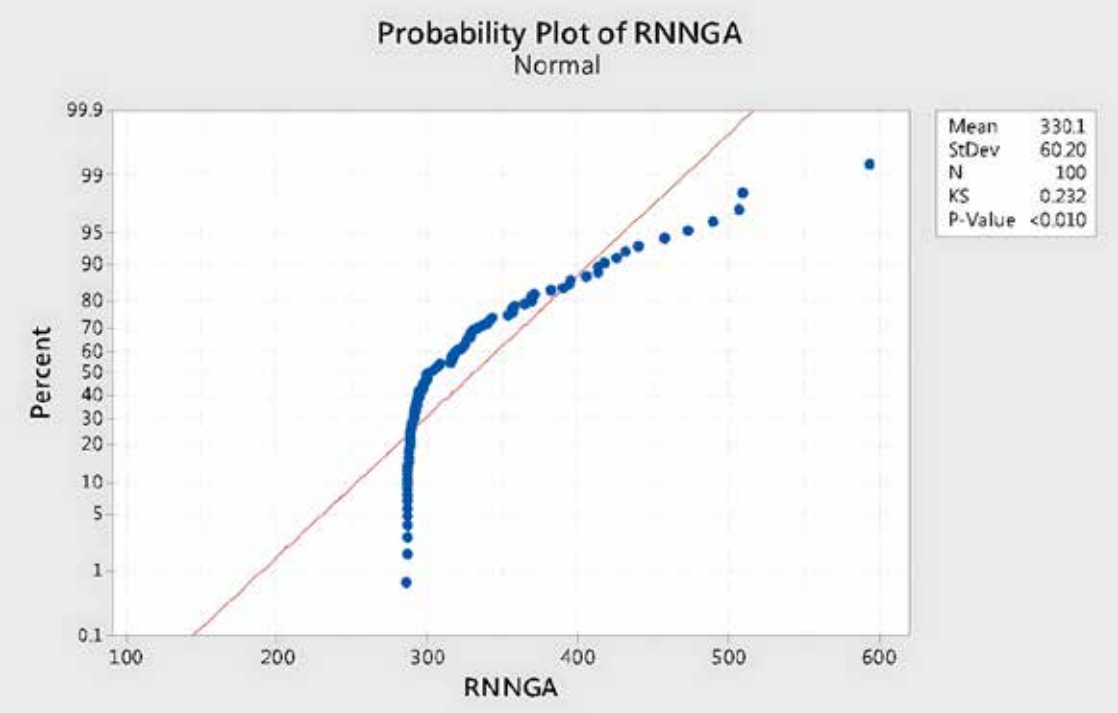

Fig. 7. K-S Normality Test Result - RNNGA Model

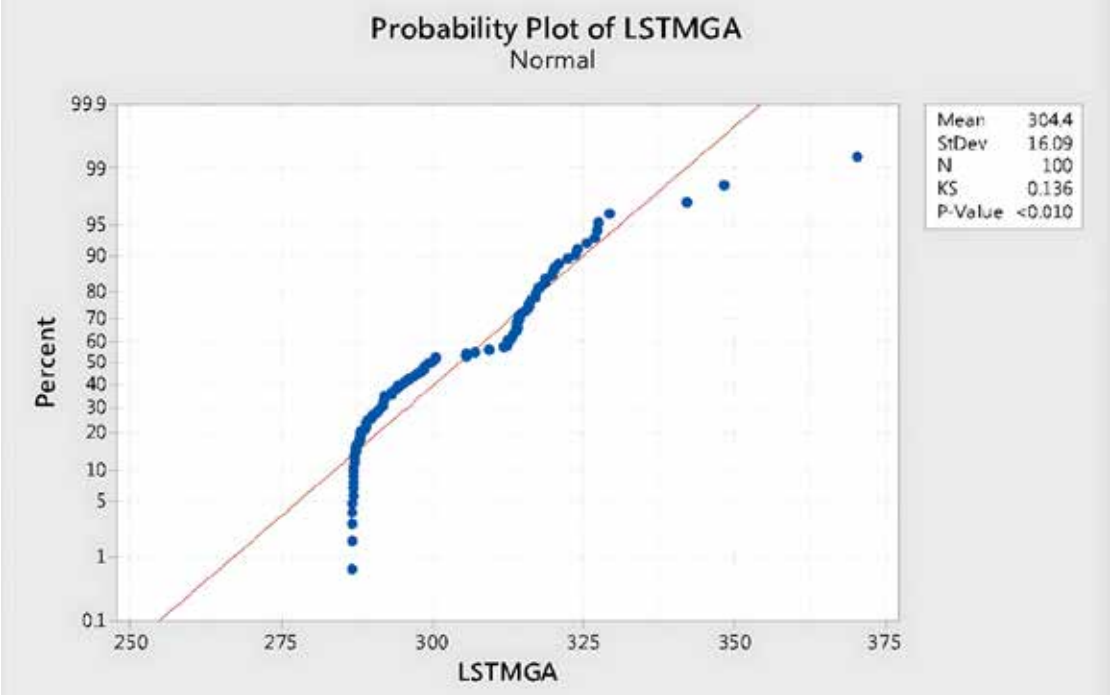

Fig. 8. K-S Normality Test Result - LSTMGA Model

is a significant difference between the TransformerGA model *MSE_test* and the RNNGA model *MSE_test*.

To summarize the Z-test results in this study: (1) There is no statistically significant difference between the two models with the best predictive performance, LSTMGA and TransformerGA. (2) There is a statistically significant difference between TransformerGA and RNNGA.

The results of this experiment are almost as good when using LSTMGA and TransformerGA, and the reason why there is no significant difference in prediction performance, we believe that the possible reasons are as follows: (1) Algorithm model capabilities are similar: LSTMGA and TransformerGA are powerful sequence modeling models that can process time series data. Although they are structured and operate differently, they may have similar learning capabilities and representational capabilities, capable of capturing patterns and structures in time series. (2) Data set characteristics: small sample data has a simpler structure, and LSTMGA and TransformerGA may be able to model at a similar level effectively. In this case, the characteristics of the data may cause the two models

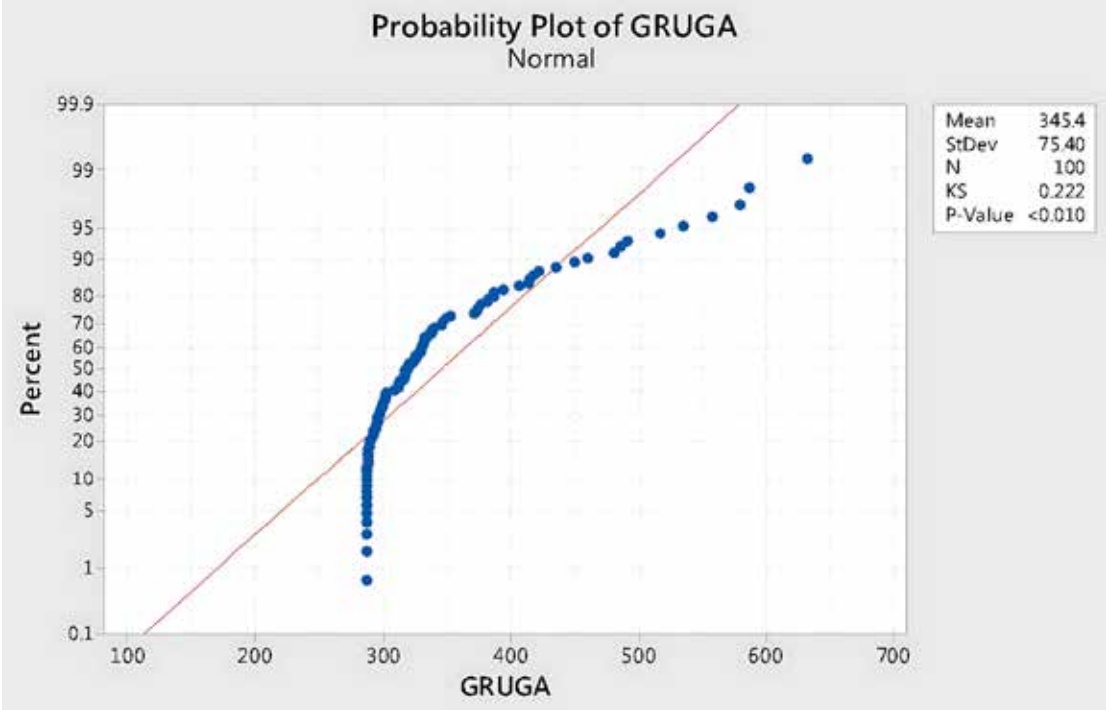

Fig. 9. K-S Normality Test Result - GRUGA Model

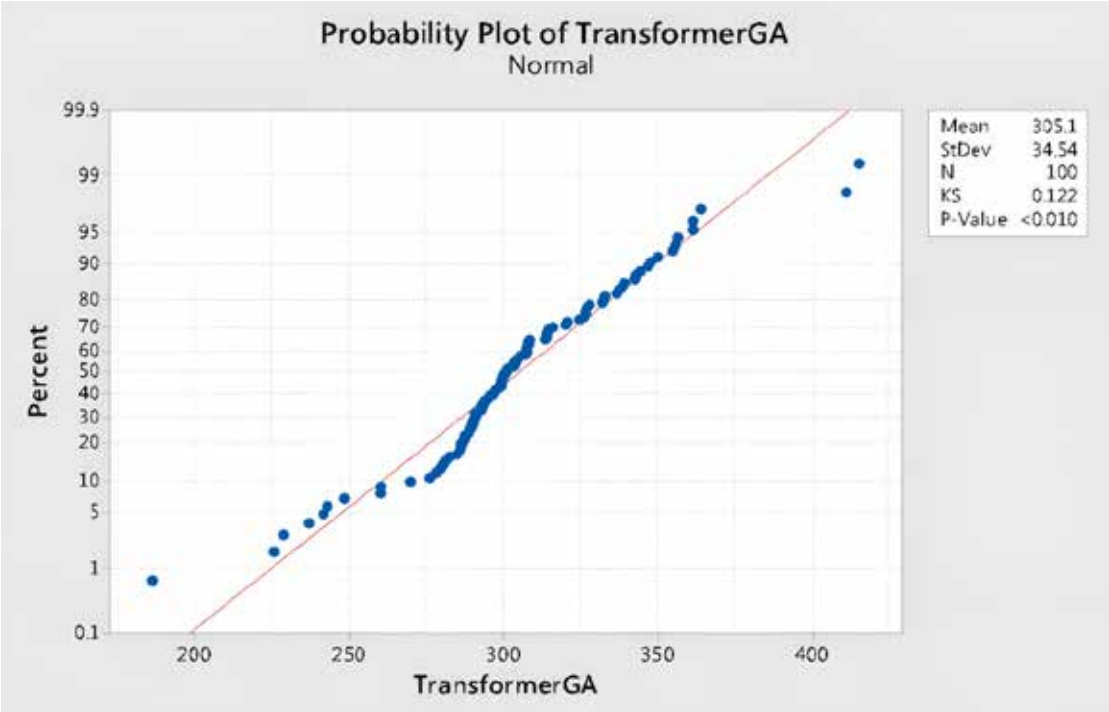

Fig. 10. K-S Normality Test Result - TransformerGA Model

to be comparable in predictive performance. (3) Parameter adjustment: In the experiment, we used Genetic Algorithm to perform similar parameter adjustment and optimization training on models such as LSTMGA and TransformerGA. By adjusting the hyperparameters of the model, such as learning rate, hidden layer size, etc., the performance of the two models can be maximized so that they can achieve similar results on small samples. (4) Specific problem domains: Some specific temporal forecasting problems may not be sensitive to the choice of model. Another possible situation is that LSTMGA and TransformerGA have similar strengths in CO_2 emission prediction problems.

On the other hand, the results of this experiment are worse than those of TransformerGA and LSTMGA when using the RNNGA model for CO_2 prediction experiments. We deduce the possible reasons as follows: (1) Small sample limitation: The RNNGA model is prone to overfitting when dealing with small sample data. Since RNN has a recurrent structure, it may be more sensitive to noise and incompleteness in small sample data, which may lead to its performance degradation. (2) Model structure limitation: Compared with the TransformerGA and LSTMGA, the model structure of RNNGA is relatively simple. RNNGA mainly relies on memory cells and recurrent networks, which may not be able to adequately capture complex relationships in data, limiting its predictive performance. Next, we will compare the computing performance of the four models and select the most suitable neural network model for greenhouse gas emission prediction.

Run Time Comparison

In addition to comparing the performance of the verification set *MSE* in the four types of neural network models, this study also conducts a trial calculation comparison of the calculation time of the model using GA.

Comparing the running time of GA under the same equipment, environment, and computing resources, we can find that under the condition of limited computing resources, to find the optimal solution under the same parameters and conditional assumptions, the computing time of RNNGA, LSTMGA, and GRUGA are significantly faster than TransformerGA.

After comparing the four algorithms under GA algorithm conditions, TransformerGA can be used if the goal-oriented search for the best solution is desired. It means that if the goal of the experiment is to find the best solution, you can try to use the TransformerGA model with the Genetic algorithm. This method will consume more computing resources but has a higher chance of obtaining better prediction results. However, other algorithms are recommended if we want to search for the best solution in a more efficient manner.

Conclusions

In recent years, greenhouse gas emissions have emerged as a critical global issue, prompting increased research attention toward prediction methods. Our study focuses explicitly on constructing a forecasting model for small-sample time series data, considering limited computational resources and time constraints. We evaluate model performance using *MSE*, *MAE*, and *MAPE* metrics.

Our experimental findings indicate that the LSTMGA (LSTM with Genetic Algorithm) model outperforms other neural network models when applied to small-sample time series data sets. We employ a Z-test for comparative analysis after conducting a Kolmogorov-Smirnov (K-S) normality test on *MSE* across the four model groups. Interestingly, there is no significant difference in performance between the LSTMGA and TransformerGA models.

Beyond performance comparison, our study involves adjusting model parameters and applying the Genetic Algorithm for optimization. The experimental results demonstrate that these deep-learning models outperform traditional statistical approaches in predicting greenhouse gas emissions.

Based on these results, we conclude that the TransformerGA model holds promise for effectively handling future greenhouse gas emission data sets, even in small sample scenarios. Additionally, the LSTMGA model excels in small-sample time-series data, and parameter optimization via the Genetic Algorithm enhances training efficiency.

In the research process, we use CO_2 emission data sets, and our research results have substantial application value in the management of greenhouse gas emissions. (1) Monitoring and reporting of greenhouse gas emissions: by establishing a time-series analysis and prediction model of greenhouse gas emissions, the monitoring values of greenhouse gas emissions within a specific period of time can be accurately estimated and used for greenhouse gas control plan emissions. (2) Energy demand management: accurate time forecasting models can help predict future energy demand, optimize energy use and supply, and reduce greenhouse gas emissions. It is an essential tool for national and corporate energy planning, urban planning, and energy management agencies. (3) Resource planning and emission reduction strategies: time prediction can provide critical information for carbon emission management and help formulate resource planning and emission reduction strategies. By predicting future changes in carbon emissions, we can better allocate resources, set carbon emission reduction targets, and implement corresponding emission reduction measures. (4) Decision support system: establish a carbon emission management model based on time prediction and provide a decision support system. Such a system can help governments, businesses, and organizations make informed decisions on carbon emission management, promoting sustainable development and green transition.

The practical model of this study can help enterprises, governments, and other institutions formulate emission reduction policies, adopt energy-saving and emission-reduction measures, and evaluate emission contributions, thereby reducing carbon emissions and mitigating climate change. In addition, the methods and results of this study can also be applied to time series data forecasting in other fields, providing a reference for future related research.

References

Alibrahim, H. & Ludwig, S. A. (2021, 28 June-1 July 2021). Hyperparameter Optimization: Comparing Genetic Algorithm against Grid Search and Bayesian Optimization. 2021 IEEE Congress on Evolutionary Computation (CEC),

AlKheder, S. & Almusalam, A. (2022). Forecasting of carbon dioxide emissions from power plants in Kuwait using United States Environmental Protection Agency, Intergovernmental panel on climate change, and machine learning methods. *Renewable Energy,* 191, pp. 819-827.

EIA. (2022). Texas State Energy Profile. U.S. Energy Information Administration Retrieved from https://www.eia.gov/state/print.php?sid=TX

Fang, Z., Yang, H., Li, C., Cheng, L., Zhao, M. & Xie, C. (2021). Prediction of PM2. 5 hourly concentrations in Beijing based on machine learning algorithm and ground-based LiDAR. *Archives of Environmental Protection,* 47(3).

Hochreiter, S. & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), pp. 1735-1780.

Hsu, A., Wang, X., Tan, J., Toh, W. & Goyal, N. (2022). Predicting European cities' climate mitigation performance using machine learning. *Nature Communications*, 13(1), 7487. DOI:10.1038/s41467-022-35108-5

Hyndman, R. J. & Athanasopoulos, G. (2018). Forecasting: principles and practice. OTexts.

Mitchell, M. (1998). An introduction to genetic algorithms. MIT press.

Potvin, J.-Y. (1996). Genetic algorithms for the traveling salesman problem. *Annals of Operations Research*, 63, pp. 337-370.

Riekstin, A. C., Langevin, A., Dandres, T., Gagnon, G. & Cheriet, M. (2020). Time Series-Based GHG Emissions Prediction for Smart Homes. *IEEE Transactions on Sustainable Computing,* 5(1), pp. 134-146. DOI:10.1109/TSUSC.2018.2886164

Rumelhart, D. E., Hinton, G. E. & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088), pp. 533-536.

Şahin, U. (2019). Forecasting of Turkey›s greenhouse gas emissions using linear and nonlinear rolling metabolic grey model based on optimization. *Journal of Cleaner Production*, 239, 118079.

Sen, P., Roy, M. & Pal, P. (2016). Application of ARIMA for forecasting energy consumption and GHG emission: A case study of an Indian pig iron manufacturing organization. *Energy*, 116, pp. 1031-1038.

Sonata, I. & Heryadi, Y. (2024, 17-18 July 2024). Comparison of LSTM and Transformer for Time Series Data Forecasting. 2024 7th International Conference on Informatics and Computational Sciences (ICICoS),

Sun, W. & Liu, M. (2016). Prediction and analysis of the three major industries and residential consumption CO_2 emissions based on least squares support vector machine in China. *Journal of Cleaner Production*, 122, pp. 144-153.

Szeląg, B., Bartkiewicz, L., Studziński, J. & Barbusinski, K. (2017). Evaluation of the impact of explanatory variables on the accuracy of prediction of daily inflow to the sewage treatment plant by selected models nonlinear. *Archives of Environmental Protection,* 43. DOI:10.1515/aep-2017-0030

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

Yin, L., Sharifi, A., Liqiao, H. & Jinyu, C. (2022). Urban carbon accounting: An overview. *Urban Climate*, 44, 101195. DOI:10.1016/j.uclim.2022.101195