IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 31 May 2024, accepted 24 June 2024, date of publication 3 July 2024, date of current version 12 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3422528

SURVEY

Deep Learning Models for Time Series Forecasting: A Review

WENXIANG LI AND K. L. EDDIE LAW
Faculty of Applied Sciences, Macao Polytechnic University, Macau, SAR, China
Corresponding author: WenXiang Li (wenxiang.li@mpu.edu.mo)

ABSTRACT Time series forecasting involves justifying assertions scientifically regarding potential states or predicting future trends of an event based on historical data recorded at various time intervals. The field of time series forecasting, supported by diverse deep learning models, has made significant advancements, rendering it a prominent research area. The broad spectra of available time series datasets serve as valuable resources for conducting extensive studies in time series analysis with varied objectives. However, the complexity and scale of time series data present challenges in constructing reliable prediction models. In this paper, our objectives are to introduce and review methodologies for modeling time series data, outline the commonly used time series forecasting datasets and different evaluation metrics. We delve into the essential architectures for trending an input dataset and offer a comprehensive assessment of the recently developed deep learning prediction models. In general, different models likely serve different design goals. We boldly examine the performance of these models under the same time series input dataset with an identical hardware computing system. The measured performance may reflect the design flexibility among all the ranked models. And through our experiments, the SCINet model performs the best in accuracy with the ETT energy input dataset. The results we obtain could give a glimpse in understanding the model design and performance relationship. Upon concluding the paper, we shall provide further discussion on future deep learning research directions in the realm of time series forecasting.

INDEX TERMS Dataset, deep learning, evaluation metrics, neural network models, time series forecasting, Transformer models.

I. INTRODUCTION

Time series data typically consists of an orderly sequence of observed or measured outcomes from a process at fixed sampling time intervals. A dataset is designed to capture information and activities within its subject matter. In time series applications, the primary tasks revolve around forecasting future states or data by uncovering underlying patterns based on historical time series data. Indeed, time series forecasting (TSF) finds widespread uses in various fields such as stock market prediction, weather forecasting, and traffic congestion anticipation, etc. Through forecasting, decision-makers obtain the abilities to identify and mitigate risks, and facilitate informed decision-making.

Research on time series forecasting started long ago due to its wide applications across different industries, e.g., energy sector [1], transportation [2] and meteorology [3], etc. Conventional statistical methods are often used effectively in prediction tasks. Traditional modelings such as autoregressive (AR) [4], moving average (MA) [5], autoregressive moving average (ARMA) [6] and autoregressive integrated moving average (ARIMA) [7] are popular for univariate predictions in smoothed time series data. However, assumptions have to be made including data stability, linear correlation, and data independence upon using these models. Hence, it is challenging to use these approaches in practice.

More sophisticated forecasting techniques have been developed, for example, the hierarchical time series forecasting [8], sparse multivariate time series forecasting [9], and asynchronous time series forecasting [10]. Indeed, different machine learning and deep learning (DL) models

The associate editor coordinating the review of this manuscript and approving it for publication was Seifedine Kadry.

are explored to tackle specific issues related to time series forecasting.

Deep learning models, in particular, exhibit promising results following the exceptional performance in the fields of natural language processing (NLP) and computer vision (CV). Each model presents a potential solution for addressing time series forecasting challenges. Certain frameworks have been leveraged for multi-objective and multi-granularity prediction [11], as well as multi-modal time series forecasting [12].

In deep learning, we shall anticipate continuous advancements and accomplishments in developing sophisticated techniques for time series data prediction. Many existing review articles on time series forecasting have delineated mostly on classic parametric models and traditional machine learning algorithms. But they lack an exposition on the latest advancements in the Transformer-based models and empirical comparative analyses on commonly used datasets across different industries. In this paper, we attempt to categorize existing time series prediction methods, delineate datasets, and scrutinize the limitations of current methodologies. Our contributions are as follows:

1) The classical time series prediction approaches are discussed.
2) Typical deep learning models related to time series forecasting are thoroughly investigated, categorized, and summarized.
3) The existing issues in time series forecasting are discussed, and some latest problems should be prioritized.
4) The prediction performance of different latest models would be carried out for comparisons.

The rest of the paper is organized as follows. Section II provides an overview of the popular publicly available datasets and some commonly used evaluation metrics in time series forecasting applications. The classical time series models for time series forecasting can be found in Section III. In Section IV, the designs and implications of deep learning models for TSF are discussed. Section V presents experimental comparisons of predictive performance for different models. Section VI summarizes future research directions in time series methods and the unresolved issues. Finally, the research content of this paper is summarized in Section VII.

II. MODEL SETUP, DATASETS, METRICS

In this section, before delving into the discussions of various state-of-the-art designs, we should review the available time series datasets that can be used for training and testing different potential models. But we shall start by characterizing an algebraic foundation of a forecast process. And the process should predict future values or trends based on past time series data for applications in diverse fields such as finance, economics, meteorology, and sales, etc.

In time series analysis, data is arranged in chronological order. Each data point is associated with a specific timepoint. There are many factors that may cause the fluctuations of variables in time. And these factors can be the *general movement*, *data (non-)stationary*, *trend*, *periodicity*, *seasonal variation*, *volatility*, and *noise*. Upon understanding the patterns of these factors, we may possibly formulate and make proper predictions and decisions.

Time series forecasting model predicts the future values of a given entity over time. In its simplest form, a one-step-ahead forecast model takes the form:

$$\hat{y}_{i,t+1} = f(\boldsymbol{y}_{i,t-k:t}, \boldsymbol{x}_{i,t-k:t}, s_i), \tag{1}$$

where $\hat{y}_{i,t+1}$ denotes the prediction of a model, $\boldsymbol{y}_{i,t-k:t} = \{y_{i,t-k}, \ldots, y_{i,t}\}$ and $\boldsymbol{x}_{i,t-k:t} = \{x_{i,t-k}, \ldots, x_{i,t}\}$ are the observed values of the target and exogenous inputs within a backcasting window of k periods, respectively. The s_i represents the static metadata associated with the entity. The $f(\cdot)$ signifies the learned prediction function of the model, and i represents a given entity. Each entity represents a logical grouping of temporal information – such as the measurements from different weather stations in climatology, or vital signs from different patients in medical study – and can be observed at the same time.

The prediction functions $f(\cdot)$'s can be realized through domain-specific model designs elaborated in subsequent sections. A function and its corresponding model can be trained and validated using application-specific datasets. In the ensuing discussion, various types of datasets will be examined, and the commonly employed evaluation metrics for time series forecasting algorithms will be outlined.

A. DATASETS

A collection of measured observations chronologically in a time series is needed to formulate and validate a time series model. The model, that would be developed, should be able to characterize a relation between data points in a given dataset. Currently, there are publicly available datasets for different applications, e.g., finance, industrial energy consumption, urban transportation, medicine, and meteorology, etc. The commonly used datasets, associated with an array of deep learning models, are extracted and listed together in Table 1. The basic information of the majority of different datasets in different domains is presented below for your reference.

1) FINANCIAL SECTOR

In finance, forecasting is frequently used to predict business cycles and long-term stock market movements. It helps financial firms and traders alike find opportunities and make responsible decisions about revenue and expenditure based on estimates. For instance, forecasting can assist investors in the stock market with developing well-rounded investing strategies, forecasting future trends and volatility in stock prices, and selecting more advantageous investments. Additionally, forecasting can help financial institutions assess loan risk [28] by anticipating future borrower repayment capability and credit hazards, or by projecting future interest rate patterns to improve monetary and interest rate policies. Curated datasets include:

TABLE 1. Datasets used in some deep learning models.

Model Name*	Dataset Names Used in Experiments	Applications
HyDCNN [13]	Traffic, Solar-Energy, Electricity	Traffic,Energy
SCINet [14]	Traffic, Solar-Energy, Electricity, ETT, Exchange-Rate, PEMS	Traffic,Energy,Finance
DA-RNN [15]	SML 2010, NASDAQ 100 Stock	Finance
MQRNN [16]	GEFCom2014	Energy
MTGNN [17]	Traffic, Solar-Energy, Electricity, Exchange-Rate, PEMS-BAY, METR-LA	Traffic,Energy,Finance
AutoSTG [18]	PEMS-BAY, METR-LA	Traffic
DMSTGCN [19]	PeMSD4, PeMSD8	Traffic
TPGNN [20]	Traffic, Solar-Energy, Electricity, Exchange-Rate	Traffic,Energy,Finance
MAGNN [21]	Traffic, Solar-Energy, Electricity, Exchange-Rate, Nasdaq	Traffic,Energy,Finance
TFT [22]	Electricity, Traffic, Retail	Traffic,Energy
Informer [23]	ECL, Weather, ETT	Weather,Energy
Autoformer [24]	ILI, Weather, ETT, Traffic, Exchange-Rate, Electricity	Medical,Weather,Energy,Traffic,Finance
Pyraformer [25]	ETT, Electricity	Energy
FEDformer [26]	ILI, Weather, ETT, Traffic, Exchange-Rate, Electricity	Medical,Weather,Energy,Traffic,Finance
Non-stationary Transformer [27]	ILI, Weather, ETT, Traffic, Exchange-Rate, Electricity	Medical,Weather,Energy,Traffic,Finance

*Discussion on deep learning models can be found in Section IV.

Gold Prices [29]: The dataset documents the daily gold prices (in US dollars) from January 2014 to April 2018, with a granularity of days. It uses values for minimum, mean, maximum, median, standard deviation, skewness, and kurtosis to characterize the characteristics of the distribution.

Exchange-Rate [30]: It aggregates daily exchange rates across eight currencies spanning from 1990 to 2016: Australia, UK, Canada, Switzerland, China, Japan, New Zealand, and Singapore.

Stock Opening Prices [31]: From the Yahoo Financial, the dataset compiles the daily opening prices of 50 equities across 10 industrial sectors between 2007 and 2016. There are a total of 2,518 data points per stock, and 5 top businesses in each sector.

2) ENERGY SECTOR

Forecasting is often used in the industrial energy sector to support long-term strategic resource planning. It helps businesses and governments forecast future energy needs, such as those for electricity, oil, and natural gas, and enables more efficient energy production and supply planning. The following open-source datasets are relevant for forecasting industrial energy:

Electricity Transformer Temperature (ETT) [32]: The curated data captures load and oil temperature data of power transformers at 15-minute intervals from July 2016 to July 2018. It is composed of hourly granularity data (ETTh1, ETTh2) and 15-minute granularity data (ETTm1).

Solar Energy [33]: With a 5-minute sampling interval, it documents the highest solar energy generation capacity of 137 photovoltaic power stations in Alabama in 2006.

Electricity [33]: The dataset tracks the kilowatt-hours of power used by 321 customers between 2011 and 2014. The intervals between data points are 15 minutes.

3) URBAN TRAFFIC

In order to predict future road traffic flow patterns and congestion, forecasting assists city traffic management departments, streamlining traffic planning and management. Additionally, long-term forecasting helps identify traffic safety issues, predict future traffic accident risks, and enhance management of traffic safety and accident prevention. The following open-source datasets are relevant to predict urban traffic:

Paris Metro Line [34]: The dataset, consists of 1,742 observations with a granularity of 1 hour, tracks the passenger movements on Lines #3 and #13 of the Paris Metro.

PeMS03, PeMS04, PeMS07, PeMS08 [17]: The PeMS system is the source of these datasets. The technology keeps track of traffic flow data that is averaged every five minutes and gathered every 30 seconds in places such as the California bay area. Traffic flow, rate of occupancy, and speed are provided with timestamps for each sensor on the highway.

Birmingham Parking [35]: The dataset encompasses the operational data of 30 parking lots managed by Birmingham National Parking, including parking lot identities, capacity, occupancy rates, and update time. The data spans from October 4, 2016, to December 19, 2016, from 8:00 to 16:30 daily, with a granularity of 30-minute.

4) ENVIRONMENTAL ISSUES

Long-term climatic trend forecasting is one important application of the time series forecasting. Both the meteorology and pollution issues would have significant impacts on industries such as agriculture, marine transportation, and natural meteorological catastrophe warning systems. In the following, freely available datasets related to pollutions and weather forecasting include:

Weather Pollutants [36]: The dataset encompasses pollution data (PM2.5, PM10, NO2, SO2, O3, CO) gleaned every hour from 76 stations in the Beijing-Tianjin-Hebei region between January 1, 2015, and April 1, 2016. It also includes hourly meteorological observations during the same period, incorporating variables such as wind speed, wind direction, temperature, air pressure, and relative humidity.

Beijing PM2.5 [37]: For the city of Beijing in China, it includes hourly PM2.5 data and related meteorological data. The dataset includes other variables such as dew

point, temperature, air pressure, combined wind direction, cumulative wind speed, snowfall hours, and rainfall hours. In total, there are 43,824 multivariate sequences. Curated PM2.5 data serve as the goal of this series.

Hangzhou Temperature [38]: This dataset records daily average temperature in Hangzhou from January 2011 to January 2017, totaling 2,820 data points.

Weather [23]: The dataset contains nearly 1600 local data from the United States Earth climate data, more than 35,000 data records collected from January 1, 2010 to December 31, 2013, at 1-hour intervals. Each record consists of a target ''wet-bulb'' temperature and 11 other climatic features.

5) MEDICAL DATA

Forecasting finds applications at various stages of pharmaceutical development. It aids in predicting drug toxicity, pharmacokinetics, pharmacodynamics, and other parameters, facilitating optimization of the design and screening processes [39]. Furthermore, long-term forecasting contributes to predicting market prospects and sales of specific medications, thereby boosting the formulation of effective marketing strategies. The following open-source datasets are relevant to medical-related studies:

Influenza-Like Illness (ILI) [40]: This dataset contains weekly records of influenza-like illness patients from 2002 to 2021, as documented by the Centers for Disease Control and Prevention (CDC). It provides insights into the proportion of ILI patients relative to the total patient population.

COVID-19 [41]: The dataset includes daily information on confirmed and recovered cases gathered from six nations (Italy, Spain, Italy, China, the United States, and Australia) between January 22, 2020, and June 17, 2020.

Medical Information Mart for Intensive Care (MIMIC)-III [9]: The dataset collected more than 58,000 hospital admission records from Beth Israel Deaconess Medical Center (BIDMC) in the United States from 2001 to 2012, including a variety of clinical features. Chief among them are the ICU admission data, blood sugar levels, and heart rates.

MIMIC-IV [42]: This dataset, which incorporates several improvements over MIMIC III, includes data updates and partial table reconstruction, collected clinical data from more than 190,000 patients and 450,000 hospital admissions at BIDMC from 2008 to 2019.

B. EVALUATION METRICS

First order and second order evaluation metrics are commonly used in predictive analysis. These indices offer the advantage of simplicity in computation, and enable comparison across different methods or models with the same dataset. They can be calculated quickly even if the sizes of the datasets are large. Typically, the popular indicators are:

$$MAE = \frac{1}{m}\sum_{i=1}^{m} | y_i - \hat{y}_i | \tag{2}$$

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2 \tag{3}$$

$$RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2} \tag{4}$$

$$MSLE = \frac{1}{m}\sum_{i=1}^{m}\left(\log(y_i + 1) - \log(\hat{y}_i + 1)\right)^2 \tag{5}$$

$$RMSLE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(\log(y_i) - \log(\hat{y}_i)\right)^2} \tag{6}$$

where $\hat{y}_i$ represents the predicted values, y_i is the true values, and m is the sample size.

Among them, the mostly used metrics would be the MAE (mean absolute error), the RMSE (root mean squared error), and the MSE (mean square error). And the main difference between RMSE and MSLE (mean squared logarithmic error) would be on how sensitive each is to the outliers in a particular dataset. The MSLE exhibits greater robustness when assessing datasets containing outliers, while the RMSE demonstrates relatively higher susceptibility to outlying data points. In contrast, the MAE reflects the actual error situation better than RMSE whereas the RMSLE (root mean squared logarithmic error) proves valuable in cases where underestimation is penalized more severely than overestimation.

Although these measures in general can be evaluated quickly disregard the size of a dataset, they do suffer certain limitations: 1) some among them do not reflect the scenarios if the true values approach zeros, and 2) some indices may exhibit skewed distributions if the true values are close to zeros. Then there are some other useful evaluation metrics, i.e.,

$$MAPE = \frac{100}{m}\sum_{i=1}^{m}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{7}$$

$$sMAPE = \frac{1}{m}\sum_{i=1}^{m}\frac{|y_i - \hat{y}_i|}{\left(|y_i| + |\hat{y}_i|\right)/2} \tag{8}$$

$$IA = \frac{\Sigma_{i=1}^{m}(y_i - \hat{y}_i)^2}{\Sigma_{i=1}^{m}(|\hat{y}_i - \bar{y}_i| + |y_i - \bar{y}_i|)^2} \tag{9}$$

$$R^2 = 1 - \frac{\Sigma_{i=1}^{m}(y_i - \hat{y}_i)^2}{\Sigma_{i=1}^{m}(y_i - \bar{y}_i)^2} \tag{10}$$

where $\bar{y}_i$ is the mean value.

The MAPE (Mean Absolute Percentage Error) is a commonly used metric that calculates the absolute percentage of the prediction error for each observation. It is intuitive, easy to understand, and works well for datasets with a small number of outliers. It provides a measure of the average magnitude of the percentage error.

On the other hand, the sMAPE (Symmetric Mean Absolute Percentage Error) is a symmetric percentage error indicator that takes into account the proportional relationship between the actual and predicted values. It provides a balanced

measure of the error between the predicted and actual values. The sMAPE is particularly useful for datasets that require symmetric treatment of errors between the actual and predicted values.

Another important metric is the IA (Interval Accuracy), which measures the accuracy of the confidence interval of time series prediction. It is suitable for forecasting problems that require consideration of uncertainty, such as meteorological prediction and financial risk estimation.

Lastly, the R^2 (coefficient of determination) is a measure of how well a prediction model fits the observational data. It represents the proportion of the variance in the observed data that can be explained by the model. R^2 is commonly used in linear models and provides a comprehensive evaluation of prediction performance.

III. CLASSICAL APPROACHES FOR TSF

A. PREDICTION METHODS FOR STATIONARY DATA

The traditional time series forecasting methodologies, such as autoregressive (AR), moving average (MA), and autoregressive moving average (ARMA), are essentially based on statistical techniques. For stationary time series forecasting, Yule [4] started and proposed a new symbolic system, which played an important role in multivariate time analysis. He later introduced the autoregressive (AR) model, which was the first to treat randomness in time series analysis. Every time series was considered a stochastic process in its own right. One of the key techniques for forecasting stationary time series data would be the ARMA model [43], which was popularized by Box and Jenkins in 1970.

The ARMA(p, q) is a linear combination of two linear AR(p) and MA(q) components, where AR(p) is an autoregressive model of order p, and MA(q) is a moving average model of order q. The ARMA is frequently used to simulate stationary time series. The time series data is treated as random sequences, with the temporal continuity of the original data being reflected in the correlation between random variables. For the analysis and forecasting of shifting trends in stationary time series, these models offer a potent tool.

Then an adaptive ARMA model [44] was introduced to handle the short-term power load forecasting of a power generation system. Through experiment findings, the adaptive ARMA model performed more accurately than the conventional ARMA model for forecasts that were made for one week and 24-hour in advance. To increase the precision of prediction intervals, a prediction interval algorithm [45] was proposed based on a bootstrap distribution for setting (p, q).

Furthermore, the univariate time series models can be mapped to multivariate time series models by drawing inspiration from the Granger causality[1] [46]. The vector autoregressive moving average (VARMA) model is based on the ARMA model, and it creatively conflates the vector autoregressive (VAR) model with the vector moving average (VMA) model. However, stationary time series data are the necessary condition for building a VARMA model. In order to achieve stationary data if a given data is non-stationary, then differencing operation is required. The Vector Error Correction Model (VECM) [47], which takes into account the cointegration relationship between time series, is a solution to this problem. These research endeavors have driven further advancements in time series forecasting and provided new approaches and methods to tackle complex time series analysis issues.

On balance, the ARMA time series forecasting model has made notable progress in addressing the stationary time series forecasting. However, pure stationary time series data is rarely found in practice. This arguably limits the applicability of the ARMA model. Hence, for better handling non-stationary time series data, more adaptive and malleable forecasting models and methods are needed.

B. PREDICTION METHODS FOR NON-STATIONARY DATA

Non-stationary sequences refer to the ordered lists that exhibit characteristics such as trends, seasonality, or periodicity. These sequences demonstrate consistency in local levels or trends, where some data points closely resemble others. To transform non-stationary sequences into stationary ones, differencing operations can be employed for adjustment. The ARIMA(p, d, q) model comprises three components: Autoregressive (AR), Integrated (I), and Moving Average (MA), where p and q are the orders for the AR and MA processes, respectively, and d is the number of differencing operations. The AR component uses past observations to predict the current value. The MA is for capturing noise and random fluctuations within the sequence. The I is to make a non-stationary time series stationary by performing differencing operations on the non-stationary sequence.

The ARIMA model, for non-stationary time series, is capable of capturing variations in different data patterns and its parameter estimations are relatively intuitive. Consequently, the ARIMA model has been widely adopted in practical applications. The regular AR(p) model can be expressed as:

$$y_t = c' + \varepsilon_t + \sum_{i=1}^{p} \varphi(i) y_{t-i} \tag{11}$$

where y_t represents the current value, c' a constant term, p the order of autoregression, φ_i the i-th autoregressive coefficient, and ε_t the white noise error term. It would be more convenience to use the backward shift (or lag) operator B to represent the ARIMA model. The AR(p) can be represented, ignoring the error term at this moment,

$$\left(1 - \sum_{i=1}^{p} \varphi_i B^i\right) y_t = c'. \tag{12}$$

[1]The definition of Granger casuality is that a series x_i is deemed not to be "causal" of another series x_j if leveraging the history of series x_i does not reduce the variance of the prediction of series x_j.

The regular MA(q) expression is for characterizing the noise error term, and we have:

$$y_t = \mu + \varepsilon_t + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} \tag{13}$$

where y_t indicates the current value, μ a constant term, q the order of the moving average, θ_i the i-th moving average coefficient, and ε_t the white noise error term. With the backward shift operators, we have

$$y_t = \mu + \left(1 + \sum_{i=1}^{q} \theta_i B^i\right)\varepsilon_t. \tag{14}$$

The I(d) model is for differencing the non-stationary sequence d times, i.e.,

$$y'_t = (1 - B)^d y_t \tag{15}$$

where y'_t signifies the differenced sequence, and d is the order of differencing. Combining the AR, I, and MA expressions, we have the ARIMA(p, d, q) model, which is

$$\left(1 - \sum_{i=1}^{p} \varphi_i B^i\right)(1 - B)^d y_t = c + \left(1 + \sum_{i=1}^{q} \theta_i B^i\right)\varepsilon_t \tag{16}$$

where c is the aggregate constant term. The Eqn. (16) represents the ARIMA process. By choosing the parameters, p, d, and q appropriately, the ARIMA models are suitable for different time series predictions and analysis.

The ARIMA model has found widespread uses in time series forecasting. For instance, Valipour [7] showed that the ARIMA model predicted well the precipitation in important areas. In [48], it was used to forecast and analyze trends in the PM2.5 concentration in Fuzhou, China. The ARIMA model was used by Malki et al. [49] to forecast the COVID-19 outbreak's second and potential end times in 2020. Shi et al. [50] proposed a model based on tensor decomposition, BHT-ARIMA, which was novel to forecast multiple short time series. Finally the experiment demonstrated good results on classical datasets, such as electricity and traffic. In this design, a time series is split into multiple blocks and the model utilizes the Hankel tensor properties to construct the proper tensor representation. Then, with tensor decomposition method, the decomposed tensors are for feature extractions in the time series.

Conclusively for the ARIMA model, its application on the time series data is to make it a stable series through differential processing. However, this particular limitation may also restrict its ability to capture nonlinear relationships. Therefore, to better handle nonlinear relationships and complex time series patterns, alternative models and methods merit consideration, such as those based on machine learning and deep learning, with a view to ensure increased accuracy and adaptability of time series forecasting.

IV. DEEP LEARNING AND TSF

In this section, we will explore various cutting-edge deep learning architectures for time series forecasting. These models are typically categorized into four main groups:

1) Convolutional Neural Networks (CNNs),
2) Recurrent Neural Networks (RNNs),
3) Graph Neural Networks (GNNs), and
4) Transformers.

Each of these deep learning models possesses distinct advantages and suitability for time series forecasting. Choosing the appropriate model depends on the characteristics of the data, the complexity of the problem, and the performance requirements. Furthermore, effective hyperparameter tuning and data preprocessing are critical for achieving accurate predictions. This section will provide an overview of models belong to classes of CNN, RNN, GNN, and Transformer-based time series forecasts. The design characteristics of these models will be summarized in tables later on.

A. CNN-BASED MODELS

The inception of Convolutional Neural Networks (CNNs) gained significant attention with the pioneering work done by LeCun in 1989. The models he developed focused on processing grid-like topological data such as images and time series data. CNN has emerged as one of the most powerful techniques for understanding image content and has demonstrated exceptional performance in image recognition, segmentation, detection, and retrieval tasks. Over the years, CNN architectures have witnessed numerous advancements, leveraging depth and spatial aspects to achieve innovative designs.

In general, based on specific architectural modifications, CNN can be broadly classified into seven categories: spatial utilization, depth, multi-path, width, channel boosting, feature map utilization, and attention-based CNN. Fig. 1 illustrates the classification of different deep CNN models.

CNN is widely adopted in field of computer vision due to its ability to effectively handle dimensionality reduction and process long sequences in parallel. Though originally designed for picture datasets to extract local associations across spatial dimensions [51], many recent works have been conducted to explore applications in time series forecasting. CNN can be modified for time series datasets by introducing additional layers of causal convolutions [52]. Filters that depict these convolutions ensure that only historical data is used to make predictions. Each causal convolution filter has the following structure for intermediate features extractions contained in the hidden layer l:

$$h_t^{l+1} = A\left((W * h)(l, t)\right) \tag{17}$$

$$(W * h)(l, t) = \sum_{\tau=0}^{k} W(l, \tau) h_{t-\tau}^{l} \tag{18}$$

where $h_t^l \in R^{H_{in}}$ represents the intermediate state at time t in layer l, and $W(l, \tau) \in R^{H_{out} \times H_{in}}$ signifies the fixed filter

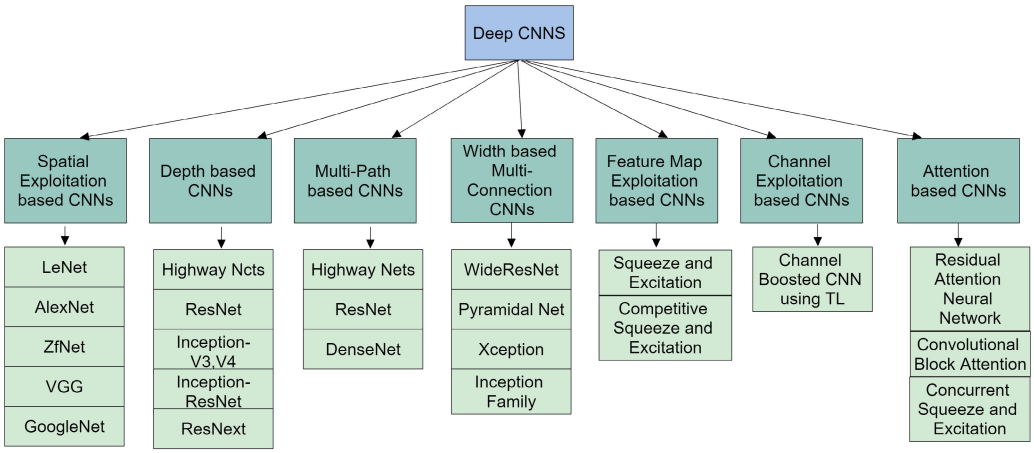


FIGURE 1. Types of CNN models.

weights in layer l, and $A(\cdot)$ denotes the activation function. The H_{out} and H_{in} represent the numbers of output channels and input channels in the convolutional layer, respectively.

1) VARIANTS OF CNN MODELS

Temporal Convolutional Networks (TCNs) [53] are widely used models for analyzing and modeling time series data. They employ causal convolutions to prevent information leakage. Fig. 2a shows architectural elements in a TCN. A TCN overcomes the challenges of gradient vanishing and explosion by incorporating a different backpropagation path in the temporal direction.

HyDCNN (Hybrid Dilated CNN) [13] is another effective approach that captures nonlinear relationships between sequences using dilated causal convolutions and linear dependencies through autoregressive modules. Fig. 2b shows architectural of HyDCNN. This hybrid framework addresses the limitations of traditional CNNs in capturing long-term correlations. Then the WaveNet-CNN [54] model is introduced by combining the WaveNet speech sequence generation model with CNNs. The model utilizes the ReLU activation function and exhibits superior performance in handling financial analysis tasks.

To address the limitations of CNNs in handling large datasets, a Kmeans-CNN [55] model is introduced, which fuses CNN with the K-means clustering algorithm, enabling the learning of more informative features. By clustering similar samples and segmenting large datasets, Kmeans-CNN achieves excellent performance in processing large-scale power load data.

In 2023, Wu et al. [56] proposed a new framework structure that combines CNN and LSTM for more accurate stock price prediction. This new method is called SACLSTM. An array of sequences of historical data and its leading indicators (options and futures) are constructed. The array is used as an input image to the CNN framework. Certain feature vectors are extracted through the convolutional and pooling layers and are used as input vectors to the LSTM.

2) CNN AND ATTENTION MECHANISM

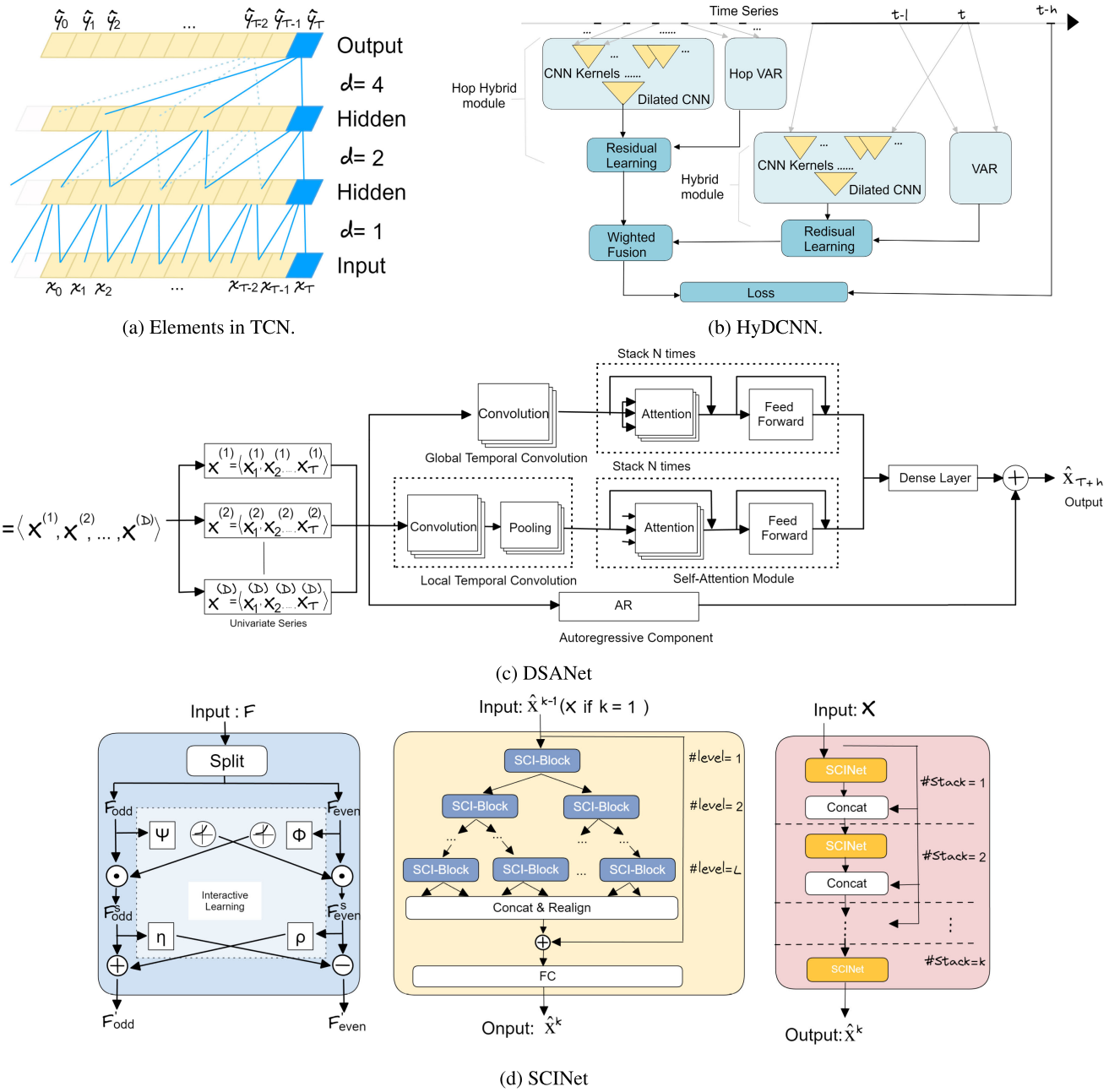
To address the challenge of capturing long-term dependencies in multivariate long-term forecasting, Huang et al. [57] developed the Dual Self-Attention Network (DSANet), which is suitable for dynamic periodic or aperiodic sequences and simultaneously considers both global and local temporal patterns. Fig. 2c shows the architecture of DSANet. Unlike the traditional recurrent neural networks (RNNs) with recursive structures, DSANet employs parallel convolutional layers to capture complex mixed patterns of global and local time. Additionally, the model leverages self-attention mechanisms to learn the correlations among multiple time series. This approach allows DSANet to effectively capture long-term dependencies in time series data and enhance the performance of multivariate long-term forecasting.

3) CNN AND SEQ2SEQ

In recent years, there are interests in integrating both CNN and Sequence-to-Sequence (Seq2Seq) models for optimizing long-term time series forecasting. SCINet, a network model developed by Liu et al. [14], is a technique for extracting useful and distinguishing temporal properties from downsampled subsequences or features. Fig. 2d shows architectural of SCINet. SCINet efficiently models complicated multi-resolution time dynamics and captures local temporal dynamics, trends, and seasonal features by aggregating these rich data from several resolutions. This allows for precise time series forecasting. The application of this comprehensive approach has resulted in notable progress in the field of long-term time series forecasting, as summarized in Table 2, showcasing the recent applications of CNN-based models in the context of TSF.

B. RNN-BASED MODELS

Elman [58] were the first to introduce recurrent neural networks (RNNs). RNNs belong to a type of neural network architecture designed for handling sequential data, such as the time series or text written in natural language. Unlike



(a) Elements in TCN.

(b) HyDCNN.

(c) DSANet

(d) SCINet

FIGURE 2. **Examples of CNNs for TSF.**

traditional feedforward neural networks, RNNs incorporate recurrent connections within the network, enabling information to be passed and persisted over time. The core idea of RNN is to use the output of previous time step as the input at current time step. This establishes dependencies between sequential data, enables RNNs to handle variable-length sequences, and captures temporal correlations and contextual information within the sequences.

The hidden state in RNN serves as the memory unit in network for each time step. Each time step updates the hidden state, which is then passed to the following layer or time step of the network. RNNs can keep track of previous data via this memory propagation technique, and they can use that data to adjust current outputs. Fig. 3 shows the internal organization of a typical RNN.

In the diagram, the x_t represents the input vector at time step t, and h_t the hidden vector at time step t. It can be observed that the traditional RNN neuron receives the previous hidden state h_{t-1} and the current input x_t. The core of an RNN lies in its RNN unit, which includes an internal memory state that summarizes past information. The calculation formula for the internal hidden state of an RNN

TABLE 2. CNN-based models.

Model	Year	Models Applied	Notes	Contributions
TCN [53]	2018	Causal CNN + Dilated CNN	Dilated causal convolutions enable the processing of long sequential inputs and the deepening of residual connections.	1. Proposed TCN for diverse CNN tasks. 2. Analyzed CNN and RNN for sequence modeling.
DSANet [57]	2019	Self-Attention + CNN + AR [a]	DSANet uses parallel convolutions for global and local temporal correlations, improving autoregressive modeling robustness.	1. Introduced DSANet for accurate time series forecasting without exogenous data. 2. Combined DSANet's non-linear attention with autoregressive model for resilience.
HyDCNN [13]	2021	Dilated CNN + AR	AR modeling captures periodic patterns, while dilated causal convolutions excel at capturing trends.	1. Combined linear and non-linear elements to capture sequential and periodic patterns. 2. Enabled adaptive fusion of hybrid modules and end-to-end training.
SCINet [14]	2022	CNN + Encoder + Decoder	SCINet, a hierarchical framework, effectively models time series with complex dynamics.	1. Introduced SCINet for complex temporal dynamics in time series. 2. Developed SCI-Block for SCINet, performing downsampling and feature extraction from input data.
SACLSTM [56]	2023	CNN+LSTM	A neural network combining CNN and LSTM outperforms traditional CNNs and LSTMs in predictions.	Introduced a 2D vector to simulate image input style for stock prediction in the network, enabling accurate stock price forecasting.

[a] AR: auto-regressive.

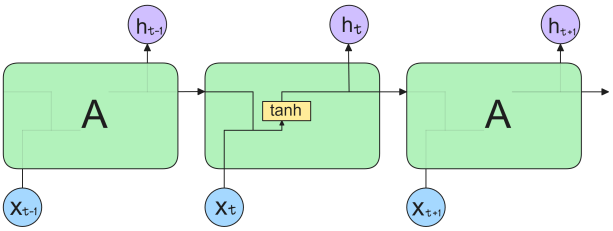


FIGURE 3. Internal structure of RNN.

is as follows:

$$h_t = \tanh(w_h h_{t-1} + w_x x_t + b) \tag{19}$$

where h_t represents the hidden state at time step t, w_h the weight matrix for the hidden state, w_x the weight matrix for the input, b the bias vector, and tanh the hyperbolic tangent activation function. The output of an RNN is:

$$y_t = w_o h_t + b_o \tag{20}$$

where y_t represents the output at time step t, w_o is the weight matrix for the output, and b_o is the bias vector.

Hochreiter and Schmidhuber [59] introduced the Long Short-Term Memory (LSTM) network, which effectively addresses these issues by incorporating gated units and memory mechanisms, in response to the challenges of vanishing and exploding gradients in conventional RNNs when handling long sequences. Fig. 4 shows the internal structure of the LSTM.

The key component of LSTM is the cell state, which acts as a conveyor belt running throughout the chain with minimal linear interactions. LSTM employs carefully designed structures called "gates" to selectively add or remove information from the cell state. Gates serve as a method to allow information to filter through. LSTM utilizes

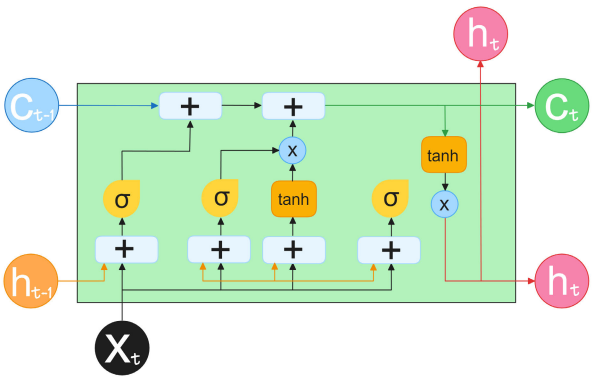


FIGURE 4. Internal structure of LSTM.

a series of gates for modulation, including the input gate, output gate, and forget gate, etc.

In the analysis of flight data, Wang et al. [60] proposed a univariate fault time series forecasting algorithm based on LSTM. They compared multiple baseline models and observed that the LSTM model exhibited superior performance. Additionally, Graves and Schmidhube [61] introduced a bidirectional Long Short-Term Memory (Bi-LSTM) network, which consists of two independent LSTMs that are concatenated. Bi-LSTM showcases stronger capabilities in handling data with longer time delays by leveraging additional contextual information without the necessity of retaining previous inputs.

The Gated Recurrent Unit (GRU) [62] model was introduced as a more efficient variation of LSTM, addressing the high computational complexity associated with LSTM models. This design simplifies the model structure, reduces computational load, and facilitates faster convergence during training.

1) VARIANTS OF RNN

Given the notable effectiveness of LSTM in capturing long-term dependencies, a model introduced by Jung et al. [63] integrates LSTM into an RNN framework for predicting photovoltaic solar energy generation in new locations. This model employs LSTM layers to learn the temporal and topographical variations in solar radiation and weather conditions, enabling the capture of temporal patterns observed across diverse locations.

Another approach addressing the challenges of training RNNs on extended time series data, including intricate dependencies, vanishing and exploding gradients, and effective parallelization, is the DilatedRNN [64]. This model combines recursive expansion layers with hierarchical expansions to address these issues inherent in the RNN network structure for sequence time forecasting. Fig. 5a shows an example of three-layer DilatedRNN with dilation 1, 2 and 4. In order to improve the accuracy of traffic flow prediction. In 2023, Bharti et al [65] proposed PSO-Bi-LSTM short-term traffic flow prediction model based on the combination of Particle Swarm Optimization (PSO) and Bidirectional Long Short-Term Memory (Bi-LSTM) neural networks.

2) RNN AND ATTENTION MECHANISM

In order to extract more pertinent correlations from data and enable predictions further into the future, some recent works has been on integrating the attention mechanisms into RNNs. The DA-RNN model [15], a two-stage RNN model with attention processes, is proposed to enhance the capacity to recognize long-term correlations in data. Fig. 5b shows graphical illustration of DA-RNN. Then the RETAIN model, proposed by Choi et al. [66], works on predicting electronic health record (EHR) data. This model utilizes two sets of attention mechanisms, visit-level and variable-level. The visit-level attention assigns higher weights to patient visits that provide more helpful information for disease prediction, while the variable-level attention assigns weights to variables that have a greater impact on disease prediction within a patient visit. The STAM (spatio-temporal attention mechanism) [67] is designed to combine LSTM with attention processes. Fig. 5c shows design model of STAM. The STAM uses LSTM layers in the encoder to find the temporal correlations in the input data. The most important time step and variable are chosen by the model for each output time step, and predictions are then made using the derived spatial and temporal context vectors.

3) RNN AND SEQ2SEQ

Because the Seq2Seq models considerably improve the capacity to handle large time series in multi-step predictions upon comparing to traditional RNNs, the Seq2Seq and LSTM are then combined [68] to fill in the encoder-decoder framework. The model makes use of a bidirectional LSTM decoder, which effectively takes into account dynamic future information by propagating future information in both forward and backward directions.

Another model is called Multi-Quantile RNN (MQRNN) model [16] which is capable of making predictions for several future time steps at once. Fig. 5d shows architectural structure of MQRNN. MQRNN uses an encoder-decoder architecture and substitutes a number of fully linked layers for the RNN in the decoder. By directly concatenating the context from the last time step of the encoder with all future features through fully connected layers, the model generates context at each time step and global context for the entire sequence, thereby enhancing the accuracy of predicting distant future values.

The MTSMFF model [69] incorporates attention mechanisms, encoder-decoder structure, and LSTM. For the encoding network, a BiLSTM with a temporal attention mechanism is used to adaptively capture implicit features and long-term correlations in multivariate time series data. Fig. 5e shows graphical illustration of MTSMFF. This model improves the ability to handle lengthy sequences and capture long-term dependencies by using layered Vanilla LSTMs for decoding. Recent research works based on RNN models for forecasting is summarized in Table 3.

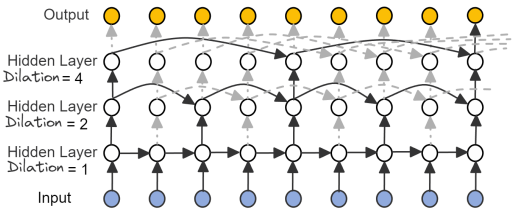
C. GNN-BASED MODELS

The Graph Neural Networks (GNNs) are deep learning models used for handling graph-structured data. In recent years, GNNs have gained significant attention in the field of time series forecasting. Time series data often exhibits complex correlations and temporal dependencies, which are difficult to capture using traditional sequence-based models. By modeling time series data as graphs, GNNs can better handle the relationships and dynamic changes between nodes. In GNNs, each node represents a time series data sample while edges represent the relationships or temporal dependencies between nodes. By encoding and propagating features on nodes and edges, GNNs can learn the interactions between nodes and the temporal evolution patterns, thus facilitating time series forecasting tasks.

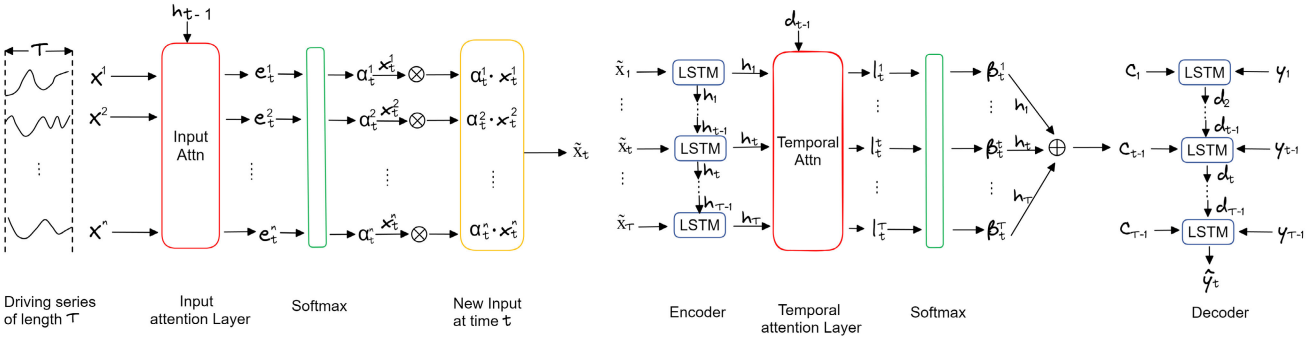
As the prediction horizon increases, the mutual influences between nodes often vary with the fluctuation of the flow, making the dynamic nature of graph structures particularly important in long-term traffic prediction. To address this challenge, researchers have to propose innovative approaches.

One design is the DMSTGCN model based on Dynamic Graph Convolutional Networks (DGNNs), introduced by Han et al. [19]. It utilizes dynamic graph convolutional modules to predict traffic speed while considering the periodicity and dynamic features of traffic. Fig. 6a shows architectural of DMSTGCN. This model captures the spatial correlations between different road segments and applies them to traffic prediction.

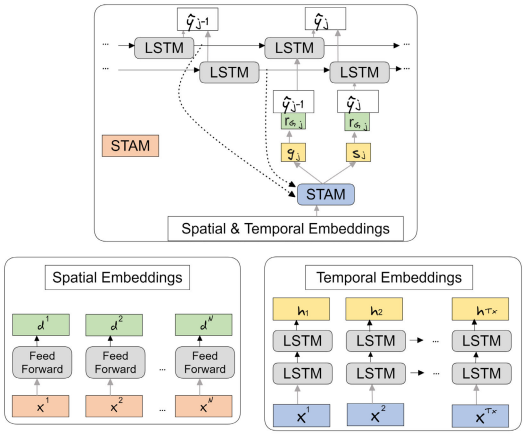
AutoSTG [18], which is a novel spatio-temporal graph automatic prediction framework. Fig. 6b shows architectural of AutoSTG. To capture intricate spatio-temporal relationships, it uses spatial graph convolution and temporal



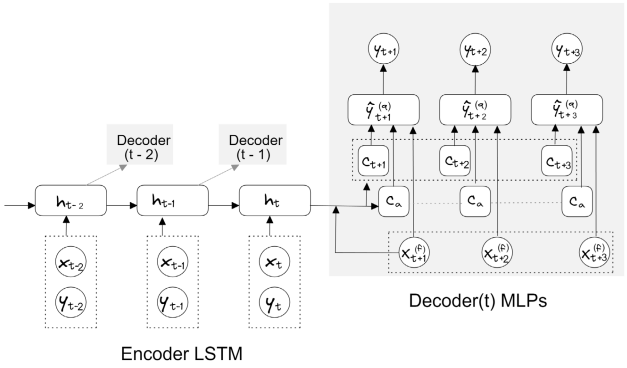
(a) An example of a three-layer dilated RNN.



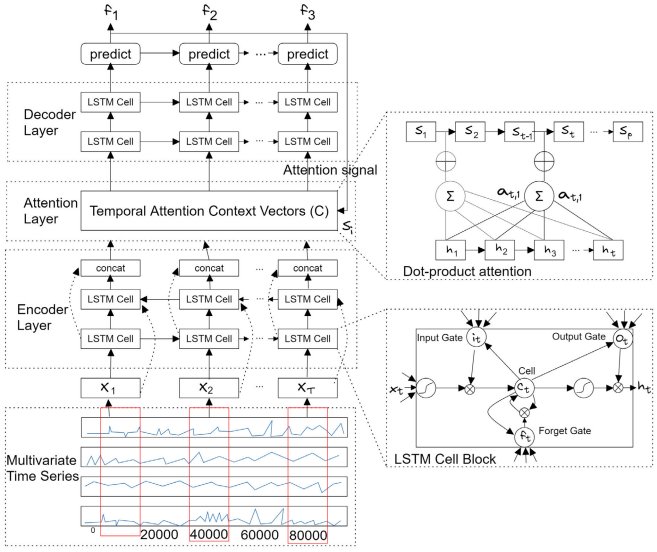
(b) Design of DA-RNN.



(c) STAM.



(d) MQRNN.



(e) MTSMFF.

FIGURE 5. Different RNN-based models for TSF.

TABLE 3. RNN-based models.

Model	Year	Models Applied	Notes	Contributions
DA-RNN [15]	2017	Attention + LSTM	Select the most relevant feature variables and hidden states in the LSTM for both stages of the Attention network.	1. DA-RNN model was proposed based on a dual-stage attention mechanism. 2. Captured long-term dependencies in sequences and select relevant input feature sequences for prediction.
MQRNN [16]	2017	LSTM + Encoder-Decoder + MLP	A multi-horizon LSTM with an encoder-decoder architecture can be used for cold- starting Single Target Forecasting (STF) tasks.	1. An efficient training approach combined sequence neural networks with multi-view prediction. 2. A network substructure was designed to address a previously overlooked problem: how to interpret known future information.
Dilated RNN [64]	2017	Dilated connections + RNN	Dilations skip connections by: 1. capturing long-term and complicated dependencies; 2. addressing the gradient disappearance and explosion issue.	1. Introduced a new dilated recurrent skip connection as the core component of the architecture. 2. By incorporating multiple dilated recurrent layers with hierarchical dilations, the DilatedRNN effectively captured temporal dependencies across various dimensions and layers.
MTSMFF [69]	2020	LSTM + Encoder-Decoder + Attention CNN	The Attention network selected the hidden states of BiLSTM.	1. A novel time attention-based encoder-decoder model was applied to multi-step prediction tasks for multivariate time series. 2. A time attention mechanism was introduced between the encoder and decoder networks.
STAM [67]	2021	LSTM + Attention	STAM captured relevant variables at each time step.	1. A new STAM architecture introduced a unique method for multi-step prediction in the realm of interpretability for multivariate time series issues. 2. By combining spatial and temporal attention mechanisms in a unified structure, it enabled an understanding of the impacts of both time and space. 3. Analysis of STAM complexity offered additional insights to enrich understanding.
PSO-Bi-LSTM [65]	2023	PSO+BI-LSTM	Bi-LSTM prediction model was fine-tuned with PSO (known for its quick convergence, resilience, and wide search capacity).	1. Employed nonlinear change weights to enhance the convergence speed of the particle swarm algorithm. 2. Integrated the particle swarm algorithm with the Bi-LSTM algorithm, addressing the limitations of manual parameter selection in Bi-LSTM.

convolution. Through this approach, the model automatically learns the associations between spatio-temporal data and utilizes them for prediction tasks.

Additionally, MTGNN [17] is a general graph neural network structure that does not require a predefined explicit graph structure. Fig. 6c shows architectural of MTGNN. The model combines graph convolution modules with temporal convolution modules to capture the spatio-temporal correlations in the data. It employs a graph learning module to adaptively extract sparse graph adjacency matrices from input data, allowing for flexible and efficient modeling of complex relationships.

Another approach is REST [70], which automatically mines inferred multimodal directed weighted graphs, is another approach. Fig. 6d shows the architecture of REST. This model combines Edge Inference Networks (EINs) and Graph Convolutional Neural Networks (GCNs). By projecting the spatial correlations between time series nodes from the temporal side to the spatial side, EINS builds multimodal directed weighted graphs for GCNs, enabling the model to effectively capture and utilize multimodal information for predictions.

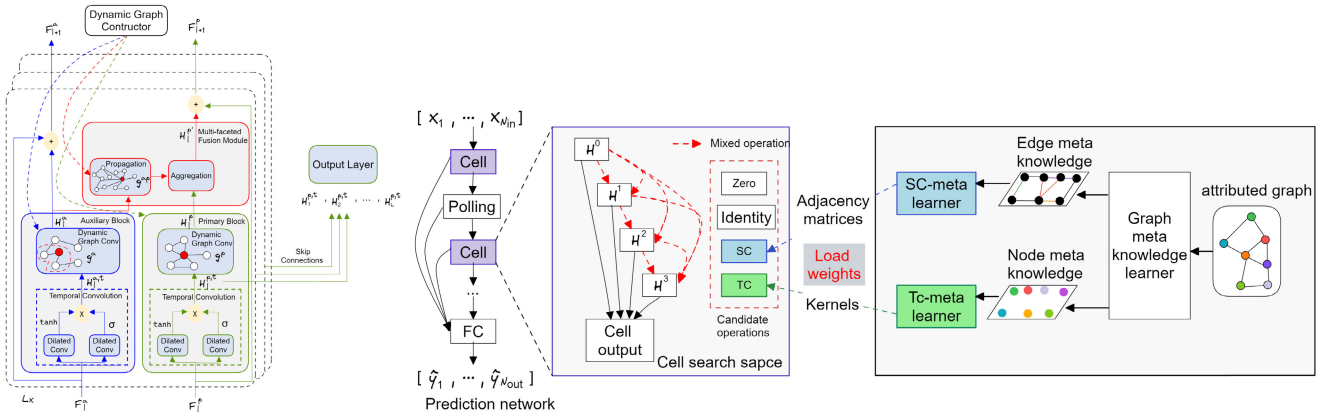
Addressing dynamic graph problems, Liu et al. [20] proposed the TPGNN. Fig. 6e shows architecture of TPGNN. This network uses time matrix polynomials to express the correlations between dynamic variables as a two-step process. First, the overall correlations are captured using a static matrix basis. Then, matrix polynomials are built for each time step using a set of time-varying coefficients and matrix bases.

In order to solve the problem in the field of time-series prediction, where temporal patterns can only be learned through dependencies between individual variables, Chen et al. [21] proposed a multiscale adaptive graph neural network (MAGNN) for MTS prediction. By utilizing a multiscale pyramidal network to model temporal hierarchies, an adaptive graph learning module to automatically infer dependencies between variables, a multiscale temporal graph neural network to model intra- and inter-variable dependencies, and a scale fusion module to facilitate collaboration across different temporal scales, the MAGNN outperforms state-of-the-art methods on six datasets.

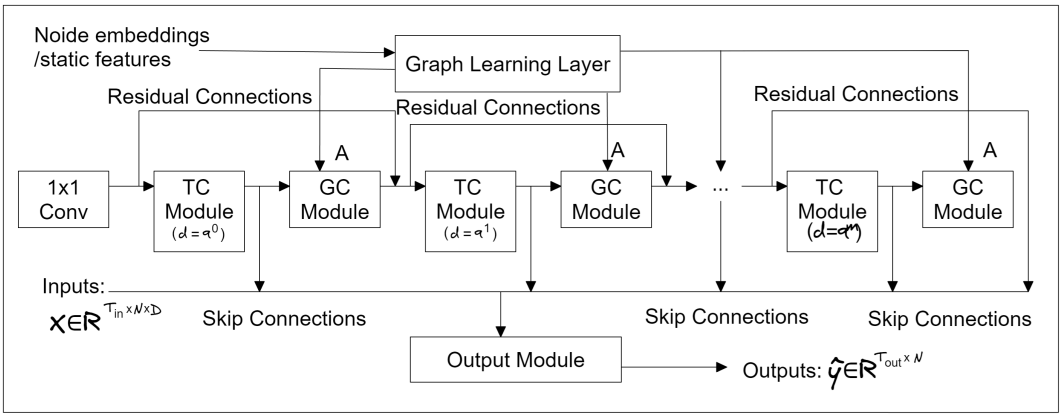
These methods share the common goal of considering dynamic graph structures in time series forecasting and utilizing GNN models to capture the spatio-temporal relationships between nodes. These innovative approaches contribute to improving the accuracy and reliability of long-term prediction and drive the research development in the field of time series forecasting. Table 4 summarizes recent works for forecasting based on GNN models.

D. TRANSFORMER-BASED MODELS

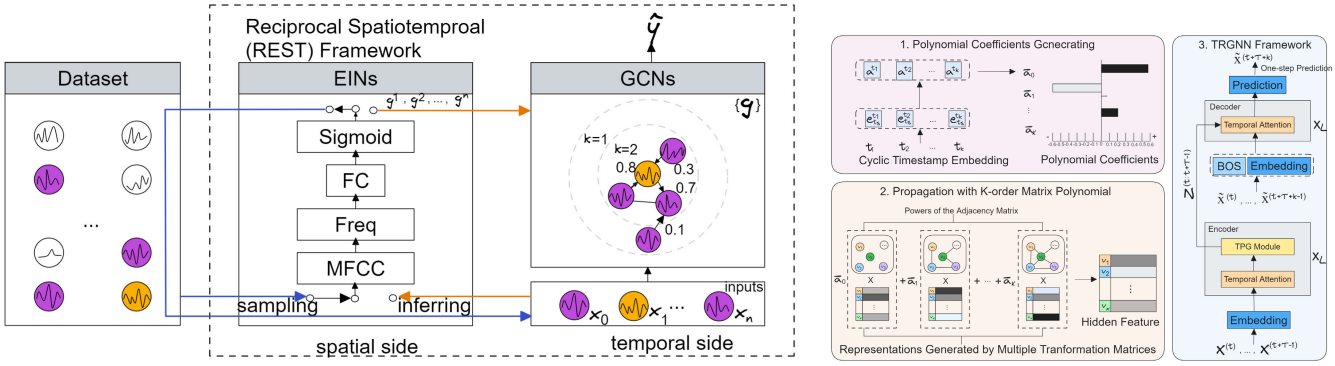
The original Transformer [71] was introduced in 2017 by Vaswani et al. This model completely departs from traditional



(a) DMSTGCN model. (b) AutoSTG model.



(c) MTGNN model.



(d) REST model. (e) TPGNN model.

FIGURE 6. **Different RNN-based models for TSF.**

CNN and RNN architectures for sequence tasks. In contrast to RNNs, the Transformer has superior parallelism, and allows interaction with global information, thereby effectively capturing correlations within sequence data through its self-attention mechanism. The self-attention mechanism of the Transformer is more interpretable and signifies a significant advancement over many conventional deep learning models. The Transformer relies entirely on attention mechanisms to characterize the global dependencies between the inputs and outputs of the model, as illustrated in Fig. 7.

The core of Transformer is the self-attention mechanism. The input is represented as (Query, Key) pairs, and the computation mechanism is:

$$A(Q, K, V) = \texttt{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V \tag{21}$$

where $Q \in R^{L_Q \times d}$, $K \in R^{L_K \times d}$, $V \in R^{L_V \times d}$ and d represents the input dimensions. The L_Q, L_K, and L_V are the lengths of the three dimensions, Q, K, and V, respectively. The probability formula for the i-th attention coefficient of the query is:

$$A(q_i, K, V) = \Sigma_j \frac{k(q_i, k_i)}{\Sigma_l k(q_l, k_l)} v_j = \mathbb{E}_{p(k_i|q_i)}[V_j] \tag{22}$$

TABLE 4. GNN-based models.

Model	Year	Models Applied	Notes	Contributions
MTGNN [17]	2020	GCN + TCN + ResNet	MTGNN graph learning module adaptively extracted sparse graph adjacency matrices.	1. Pioneered the exploration of multivariate time series data from a graph-based perspective using graph neural networks. 2. Introduced a novel graph learning module to capture spatial dependencies among variables. 3. Presented a unified framework combining multivariable time series modeling with graph structure learning.
AutoSTG [18]	2021	GCN + Meta-Learning	Meta-learning was used to learn adjacency matrices and convolution kernel widths for dynamic graph learning.	1. Introduced the innovative AutoSTG framework. 2. Utilized meta-learning to capture spatiotemporal correlations in spatial graph convolution layers and temporal convolution layers.
DMSTGCN [19]	2021	Dynamic Graph Constructor + DGCN + TCN	DMSTGCN utilized tensor decomposition for dynamic graph construction.	1. Proposed an innovative method for learning dynamic spatial dependencies. 2. Introduced a fusion module to integrate auxiliary and primary hidden states in a spatial-temporal manner.
REST [70]	2021	LSTM + DCRNN + GCN	EINS inferred a multimodal directed weighted graph.	1. Introduced the REST framework for spatiotemporal predictions in scenarios with incomplete structural information. 2. Proposed a phased heuristic strategy and a spatiotemporal coupled learning paradigm to enhance training stability in the REST framework.
TPGNN [20]	2022	GNN + Encoder-Decoder + Attention	The TPG module represented correlations using a time-varying matrix polynomial.	Introduced a novel dependence learning module for capturing dynamic correlations in multivariate time series data using a temporal matrix polynomial.
MAGNN [21]	2023	CNN+GNN+Adaptive Graph Learning	An energy-efficient anomaly detection method based on subgraph was proposed.	1. For learning temporal representations and synthesizing multi-scale temporal patterns and inter-variable dependencies. 2. Designed an adaptive graph learning module to explore inter-variable dependencies at different time scales.

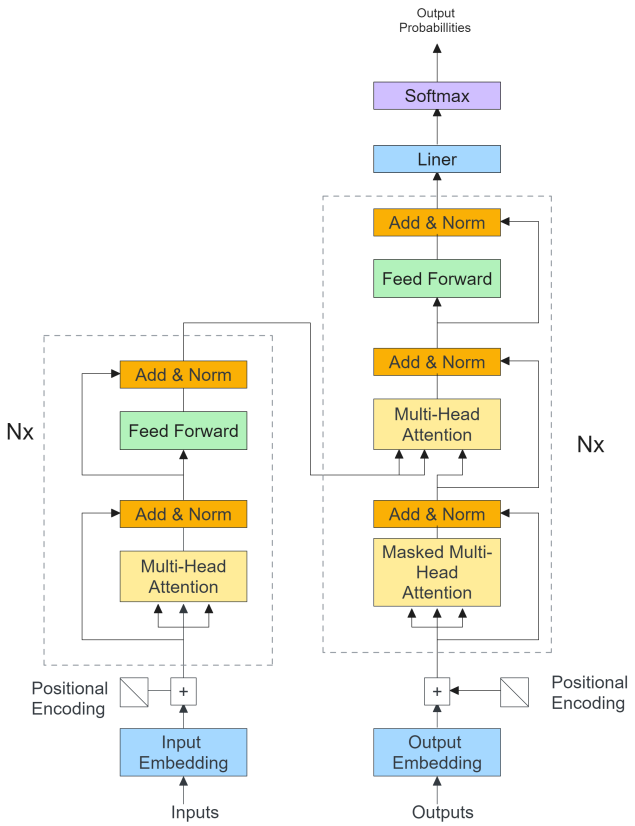


FIGURE 7. Structure of a Transformer.

for $p(k_i, q_i) = \frac{k(q_i,k_i)}{\Sigma_l k(q_l,k_l)}$, and $k(q_i, k_i)$ selects asymmetric exponent $\exp\left(\frac{q_i k_i^T}{\sqrt{d}}\right)$.

1) VARIANTS OF TRANSFORMER MODELS

Wu et al. [40] introduced the vanilla Transformer to the field of flu disease time prediction. The Transformer analyzes the complete sequence of data that has been presented and, using a self-attention process, learns the dependencies in the temporal data. The overall structure is to directly transfer Transformer Encoder-Decoder, use the self-attention mechanism to learn complex patterns and dynamics from time series data to work, and take influenza-like disease (ILI) prediction as a case study to achieve good results. However, the model often fails to adequately encode long sequences when dealing with long sequences of inputs and loses long-term dependencies. Subsequently, in order to solve transformer's inability to adequately encode long sequences and other problems, other transformer models have emerged one after another, which can be roughly divided into the following three categories:

a: TRANSFORMER AND ATTENTION MECHANISM

Lin et al. [72] proposed a Transformer-based SpringNet that uses a Dynamic Time Warping (DTW) attention layer to gather local correlations in time series data in order to predict solar energy. Autoformer [24] is another upgraded version of Transformer that optimizes the original Transformer for time series problems. Fig. 8b shows architecture of Autoformer. Its core includes the Series Decomposition Block module and an upgraded Auto-Correlation Mechanism for multi-head attention.

In order to improve the training speed of transformer, Lee-Thorp et al. [73] proposed a FNet model based on improved Transformer, in which the self-attention sublayer is replaced by the unparameterized Fourier transform, and

the training speed of the model is greatly improved. Pyraformer [25] overcomes the computational complexity limitations of Transformer by using a pyramid attention mechanism to form a multi-resolution representation of time series, effectively capturing the interdependencies in TSF. Fig. 8c shows architecture of Pyraformer.

Liu et al. [27] introduced Non-stationary Transformers which improve the attention mechanism within the Transformer by incorporating non-stationary information, thereby enhancing data predictability and unleashing the excellent time-series modeling capabilities of attention mechanisms. Fig. 8d shows architectural of Non-stationary Transformers. In 2023, Du et al. [74] proposed Preformer, a Transformer-based prediction model that introduces a novel and efficient multiscale segment correlation mechanism that divides a time series into multiple segments and utilizes segment correlation-based attention instead of point correlation. A multiscale structure is established to aggregate dependencies on different time scales, which facilitates the selection of segment lengths.

Informer [23] represents a significant contribution to long-period time series forecasting. It improves on the traditional Transformer model from the perspective of efficiency. For long-period time series forecasting, the time complexity of the regular Transformer model increases exponentially with the length of the input series. Informer creates a sparse attentions with the keys and important queries. Through testings, the attentions scores indicates long-tailed distributions between key and important queries. In fact, most of the scores are small, and only a few of them are large.

Then, Informer focuses on modeling those with important attentions, and the rest would be ignored. This design creates a structure of sparse attentions, and greatly improves the computational efficiency. Informer further introduces self-attention distillation between every two Transformer layers. A convolution operation is used to reduce the length of the sequence by half. This further reduces the training overhead.

At the decoder stage, Informer employs a method of predicting the results of multiple time steps at once, which is able to alleviate the problem of cumulative error. Overall, Informer effectively improves the operational efficiency and performance over Transformer in the long-period time series forecasting tasks. Apart from the sparse attention and self-attention distillation techniques, Informer provides an effective solution for handling large-scale long sequence data in real applications. Fig. 8a shows the model design of an Informer.

b: TRANSFORMER COMBINED MODELS

Lim et al. [22] proposed a method combining LSTM and Transformer, which was called Temporal Fusion Transformer (TFT). Fig. 8e shows the architecture of TFT. The model uses the sequence modeling capability of LSTM to preprocess input sequences first, so that representations considering context and timing information can be generated at different times. Next, the bottom layer representation is input into the upper layer Transformer to make use of Attention's ultra-long period information extraction capability to make up for the problem of sequence model information loss. Madhusudhanan et al. [75] propose a U-Net-inspired Transformer architecture called Yformer, which is based on a unique Y-shaped encoder-decoder architecture that combines coupled scaling mechanisms with sparse attention modules to capture long-term effects across scale levels.

c: TRANSFORMER AND TIMING ANALYSIS

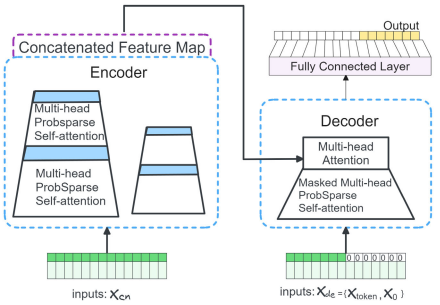
Zhou et al. [26] introduced the FEDformer model, which utilizes trend and seasonal decomposition to incorporate seasonal-trend decomposition within Transformer. Another approach is the introduction of Fourier Transform and using Transformer in the frequency domain to help the model better learn global information. Fig. 8f shows the FEDformer model. The core modules of FEDformer are Fourier transform module and timing decomposition module. The Fourier transform module converts the input time series from the time domain to the frequency domain, and then replaces Q, K and V in Transformer with the frequency domain information after Fourier transform, and carries out Transformer operation in the frequency domain.

Autoformer [24] also used the decomposition of trend items and seasonal items. In order to extract seasonal items, this paper adopted the moving average method. By calculating the average value of each window in the original input time series, the trend items for each window are derived, and then the trend items of the whole series were obtained. At the same time, the seasonal term can be obtained by subtracting the trend term from the original input sequence according to the addition model. It is worth noting that due to the success of Autoformer and FEDformer, exploring self-attention mechanisms in the frequency domain in time series modeling has attracted more and more attention from society. Table 5 shows the recent works based on transformer models for TSF.

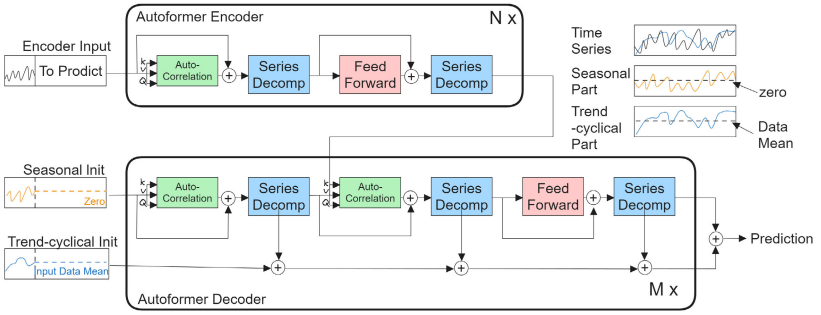
2) COMPLEXITY ANALYSIS OF TRANSFORMER-BASED MODELS

The time complexity of the Transformer [40] model depends mainly on the length of the sequence and the number of hidden layers in the model. Transformer has $O(L^2)$ time and memory complexity, so for longer input sequences and deeper models, the time and memory complexity of the Transformer can become quite high. To solve this problem, subsequent researchers have proposed a number of effective transformer class models to reduce complexity. For example, sparse attention, hierarchical attention and other methods are used.

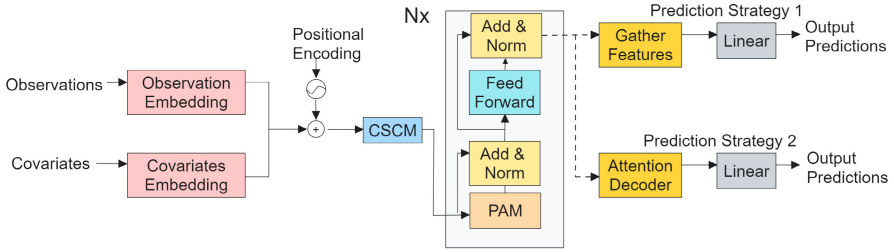
In the self-attention model, Informer introduced the concept of sparse bias and adopted the LogSparse mask



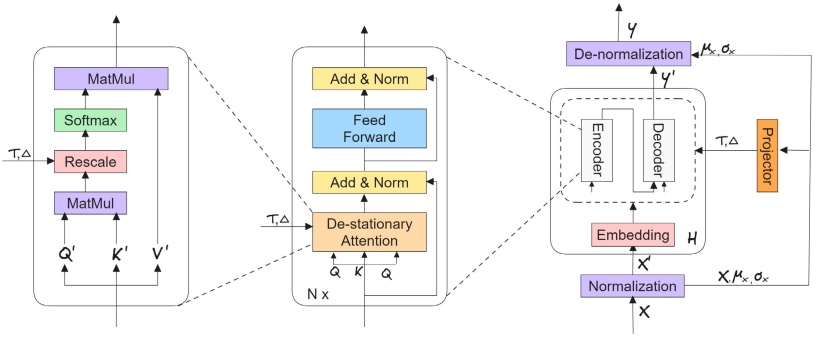
(a) An Informer model.



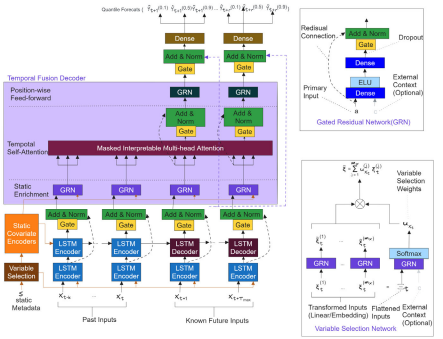
(b) The Autoformer architecture.



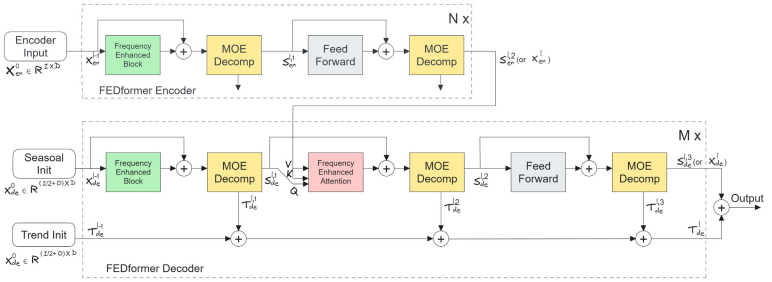
(c) The Pyraformer model.



(d) Architectural designs of non-stationary Transformers.



(e) Temporal Fusion Transformer.



(f) The FEDformer model.

FIGURE 8. **Different Transformer-based models for TSF.**

TABLE 5. Transformer-based models.

Model	Year	Models Applied	Notes	Contributions
TFT [22]	2019	LSTM + Transformer	Sequential modeling combined with feature extraction for time series forecasting.	1. Enhanced multi-horizon forecasting with attention mechanisms for better performance and interpretability. 2. Provided interpretability by identifying key prediction variables, capturing consistent temporal patterns, and detecting significant events.
Informer [23]	2020	Transformer + Sparse Attention + Distillation	Reduced computational complexity for long-term time series forecasting by optimizing Transformer.	1. Introduced Informer for improving long sequence time series forecasting by utilizing Transformer-like models to capture long-range interactions between input and output sequences. 2. Suggested using the ProbSparse self-attention mechanism and self-attention distilling operation. 3. A generative decoder was proposed to predict the next step in the sequence for generating long sequence outputs.
Autoformer [24]	2021	Transformer + Series Decomposition + Auto-correlation	Autoformer was an upgraded version of Transformer that enhanced its performance through sequence decomposition and auto-correlation mechanisms.	1. In order to address intricate time patterns in long-term forecasting, Autoformer was considered as an architectural decomposition solution. 2. An Auto-Correlation mechanism was presented to enable the discovery of dependencies and aggregation of information at the series level.
Pyraformer [25]	2022	Transformer + Pyramidal Attention	A low-complexity pyramid attention model was used for long-term sequence modeling and prediction.	1. Pyraformer was a compact multi-resolution approach to capture temporal dependencies across various ranges. 2. Theoretically by selecting suitable parameters, it was possible to achieve a maximum path length of $O(1)$ and concurrently maintain a time and space complexity of $O(L)$.
FEDformer [26]	2022	Transformer + Series Decomposition + FFT	The utilization of Fourier transform and time series decomposition helped to narrow the gap between the data distribution and the true distribution.	1. Presented a deconstructed Transformer architecture with frequency-boosted experts for seasonal-trend decomposition to capture global aspects in time series data. 2. Fourier and wavelet enhancement blocks within the Transformer structure helped map important structures in time series data through the frequency domain.
Non-stationary Transformer [27]	2022	Transformer + Destationary Attention + Series Stationarization	Non-stationary Transformers handled non-stationarity in time series data through external data denormalization and internal data learning mechanisms.	Developed Non-stationary Transformers, a flexible framework, to enhance time series predictability by reintegrating the underlying non-stationarity of the original series and addressing excessive stationarization.
Preformer [74]	2023	Transformer + Segment-Correlation mechanism	Novel and efficient multi-scale fragment correlation mechanism was introduced to divide time series into multiple fragments and use fragment correlation-based attention instead of point correlation.	Introduced an efficient attention mechanism, MSSC, which leveraged the correlation of segment pairs to uncover dependencies and aggregation information in time series data.

technique. These innovations have significantly reduced the computational complexity of the traditional Transformer model, originally $O(L^2)$, to $O(L \log L)$. It is worth noting that Informer [23] did not explicitly introduce sparse bias when implementing this improvement. Instead, it dynamically selects queries and keys that have significant similarities, effectively prioritizing the dominant $O(L \log L)$ queries. This method greatly improves the computational efficiency.

Autoformer [24] inherited the design idea of Transformer in the encoder-decoder structure. By employing Autoformer's unique internal operators, it is able to effectively separate the overall trend of a variable from the hidden variables predicted. This approach utilizes a unique autocorrelation mechanism to achieve $O(L \log L)$ complexity.

A pyramidal attention module was designed in Pyraformer [25] to transmit information between and within different scales. As the length of the input sequence L increases, Pyraformer is able to achieve lower $O(L)$ computational complexity by adjusting one parameter C and keeping the others fixed. This method has a higher benefit in terms of calculation time and memory cost.

TABLE 6. Complexity analysis of transformer-based models.

Model	Training		Testing
	Time	Memory	(steps needed)
Transformer [40]	$O(L^2)$	$O(L^2)$	L
Informer [23]	$O(L \log L)$	$O(L \log L)$	1
Autoformer [24]	$O(L \log L)$	$O(L \log L)$	1
Pyraformer [25]	$O(L)$	$O(L)$	1
Fedformer [26]	$O(L)$	$O(L)$	1

The Attention mechanism used in traditional Transformer is square complexity, while the $O(L)$ computational complexity can be reached in FEDformer [26] due to the use of low-rank approximation. Table 6 summarizes the algorithm complexity analysis of some transformer models applied to time series forecasting.

3) EFFECTIVENESS OF TRANSFORMERS IN FORECASTING

As the role of Transformer becomes increasingly prominent in the field of deep learning, more publications have begun the debate about its effectiveness in time series

forecasting. In [76], a simple linear model was demonstrated to outperform the complex Transformer models. This raised the doubts about the necessity of using Transformer in time series forecasting.

Later on, there was a mixer structure in the MTS-Mixers model [77] design. The design was adopted from computer vision (CV) field to work on the time series prediction scenario. The function of the mixer is to craete a mix based on the time dimension and channel dimension of factorization. The source time series is divided into multiple sub-sequences, and each sub-sequence is learned from the temporal information respectively, and then pieced together in the original order. By utilizing full connections within and across channels, the MTS-Mixers model replaces the complex Transformer model and achieves superior outcomes.

Recently, the TiDE model [78] was proposed which was completely composed of fully connected MLPs without resorting to any attention mechanism, RNNs or CNNs. In the paper, it reported that the design achieved the state-of-the-art results on multiple datasets, and beat the Transformer once again.

V. EXPERIMENTS

Albeit the deep learning models covered in the paper intake time series input data, each of them may be designed to deal with different primary forecasting objectives. We still like to examine the prediction performances among these models in the situation that the models are tested under the same input datasets running in an identical computing platform. Although the ranked measured results may not reflect the excellence of individual architectural design due to their different design goals, we could get at least a hindsight about the design ideas and the potential forecast accuracy relationship.

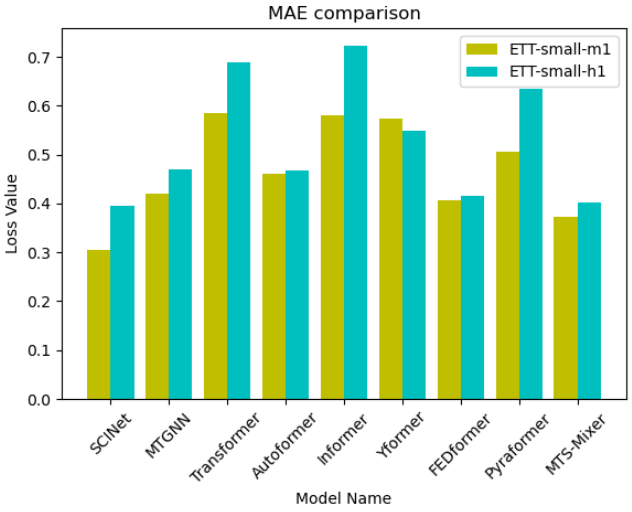
A. THE SETUP

For the experiments, we have selected the datasets, ETT-small-m1 and ETT-small-h1, for multi-step predictions. The ETT dataset [32] consists of information gathered from electricity transformers, including the temperature of the oil, and six power loading parameters. The dataset spanned two years and recorded one data point per minute or per hour, as denoted by m and h, respectively. That is, the datasets, ETT-small-m1 and ETT-small-h1, were used in our experiments.

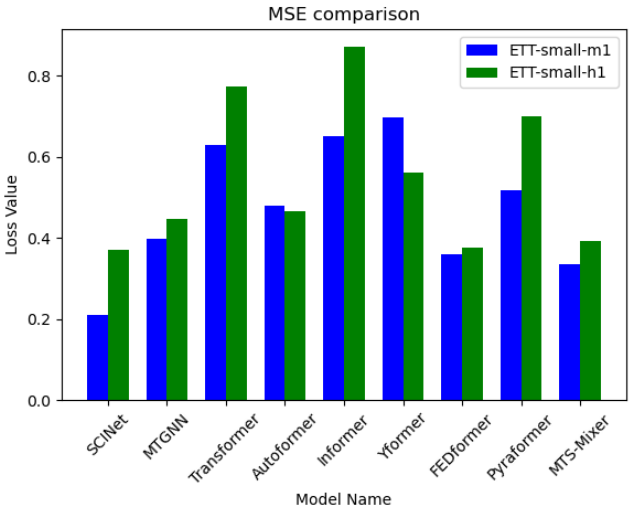
Regarding the setup, the historical horizon length and prediction length for the dataset were both set to 96. The learning rate (L_r) was 0.0001, with a total of 4 epochs and a batch size of 32. The best results obtained are shown in bold in Table 7.

B. RESULT ANALYSIS

In Figs. 9a and 9b, the predictive performance in terms of MAE and MSE among the nine different models under tests are plotted in histograms. In each graph, the error measures against the two datasets, ETT-small-m1 and ETT-small-h1,



(a) .



(b) .

FIGURE 9. Histogram plots of performance of different deep learning models.

are displayed. Among the nine selected models, the SCINet demonstrates a better overall accuracy performance for either MAE or MSE measure. We attribute the results to the fact that the SCINet can better capture the time dependency in both short-term (local temporal dynamics) and long-term (trend, seasonality) scenarios. Moreover, this design is, especially, accurate in predicting long-term forecasting results.

Then, the MTS-Mixers model performs the second best with both the MAE and MSE measures. Likely, the MTS-Mixers realizes information exchanges between the channel and token dimensions through the MLP. The mixer structure was originally designed for computer vision, the MTS-Mixers model was successful in transplanting the design into an effective component for forecasting. Besides, the trivial MLP design proves useful and effective in time series forecasting.

TABLE 7. Forecasting performance of different tpes of models base on ETT-small-m1 and ETT-small-h1 datasets (on NVidia RTX 3080 video card).

Model	ETT-small-m1		ETT-small-h1	
	MSE	MAE	MSE	MAE
SCINet	**0.20980**	**0.30460**	**0.36921**	**0.39584**
MTGNN	0.39820	0.42066	0.44809	0.46907
Transformer	0.62832	0.58537	0.77452	0.68872
Autoformer	0.47886	0.46142	0.46592	0.46682
Informer	0.65231	0.58045	0.87181	0.72298
Yformer	0.69880	0.57389	0.56086	0.54988
FEDformer	0.35978	0.40631	0.37683	0.41468
Pyraformer	0.51749	0.50512	0.70145	0.63532
MTS-Mixers	0.33460	0.37231	0.39133	0.40282

Through the experiments, the performance of FEDformer performs closely to that of the MTS-mixers. Season and trend decomposition methods are adopted in FEDformer model, and Fourier analysis is combined with Transformer based method. The experimental results show that the predictive performance of FEDformer model is much better than those methods based on ordinary Transformers. For MTGNN, the models perform poorly on both MSE and MAE, and their predictive power on this dataset is relatively weak, requiring further tuning of model parameters or the use of more complex structures to improve performance.

VI. DISCUSSION AND FUTURE RESEARCH

In the paper, we have systematically outlined various approaches for carrying out time series forecasting, from fundamental principles to the latest deep learning designs. We have categorized these deep learning models into Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Graph Neural Networks (GNNs), and Transformer-variants classes. Indeed, upon understanding the model architectures and design ideas, each model among them may pursue its own goals in its time-series application fields. As listed in Table 1, the respective application fields of the reviewed deep learning models are listed.

Since time-series forecasting covers a broad range of applications, the availability of datasets in their respective areas could be crucial in training the deep learning models. For example, in finance, some models are designed for pursuing the accuracy of stock price predictions. Models with accurate predictions of market trends can definitely assist investors with informed decisions. This may further have potential to earn investment profits. There are some specific designs for finance predictions, one such reviewed model is the DA-RNN, which can be applied to stock market predictions. In fact, other time-series models, e.g., SCINet, MTGNN, Autoformer, and FEDformer may also work well in financial markets. Other datasets and models in other sectors, such as energy sector, transportation, weather forecast, healthcare issue, etc. are also discussed.

A. FUTURE RESEARCH

A well-crafted time series model should address a spectrum of challenges, such as forecasting accuracy, long-term dependency, noises, nonlinearity, imbalanced and non-smooth data issues. While our study offers insights in comparing deep learning models for time series predictions, their differences in performances warrant further exploration.

Conceptually advanced deep learning design models (e.g., Mamba model) and research paradigms (e.g., stable diffusion) have recently been emerging. They potentially usher new platforms for constructing novel model designs. Lately, the design of Mamba is innovative in the fields of natural language processing, genomics, and audio analysis. It uses a linear time series modeling architecture, combined with a selective state space, to offer superior performance across different modalities. The selective mechanism in Mamba accelerates the speed of inference significantly. It addresses the computation challenges faced by traditional Transformers upon dealing with long sequences, its inference speed is almost five times faster than the regular Transformer model. This will be our expectation that novel deep learning models based on Mamba will possibly be developed in the near future. Another issue would be related to the available time series data, especially, from data-rich sample data to small-sized, irregular spaced time series data for the training of backdone models. Transformer-based models are prone to overfitting problems on small datasets. This requires in-depth thinking and appropriate solutions for time series data. One potential research topic would be associated with the, for example, stable diffusion mechanism, in creating more statistical consistent data samples. This can be an important direction for future research, and for small datasets with irregular interval sampled datapoints. New model architectures and techniques should be investigated to overcome the limitations of the current Transformer model and achieve better overall performance and generalization capabilities.

Another associated research topic could be the "decomposition learning" in time series forecasting. Its goal is to decompose complex time series data into vectors of various dimensions, which may possibly improve the effectiveness of the subsequent downstream tasks. In general for forecasting, there are multiple influencing factors and variables. Complex interactions maong these factors may arise. Thus the goal of decomposition learning is to separate these complex factors and obtain the appropriate vector representations of each dimension, such that each vector may better capture a specific aspect about the characteristics of the data. Our expectation on decomposition learning is about the generation of more accurate, interpretable, and dynamic time series forecasting models for real-world applications.

In summary, despite its challenges and complexity, deep learning holds great promises for the model developments. Emphasis should be placed on enhancing model interpretability, generalization, integration of external factors, real-time

prediction, uncertainty estimation, and other pertinent aspects to advance the field of time series forecasting.

VII. CONCLUSION

Time series forecasting is an excellent tool that provides predictive insights, offers decision-making strategies, and spans across diverse applications through time series data analysis. In this paper, we have undertaken a comprehensive review of publicly available time series datasets and explored various approaches for designing effective time series design models. We starts with an overview of classical statistical methods, such as ARIMA, and then delves into an extensive exploration of different deep learning models, such as Transformer-based architectures. Each architectural paradigm embodies distinct design principles and advantages. It is crucial to recognize that different datasets may necessitate tailored models, and the computational setup can influence model performance. Our goal is to ascertain the optimal deep learning model for a given dataset and computational environment. Through our experiments among different deep learning models, the SCINet demonstrates the best performance in the mean squared errors (MSEs) of 0.21 and 0.37, and mean absolute errors (MAEs) of 0.3 and 0.4 for ETT-small-m1 and ETT-small-h1 datasets, respectively. Although our testing configuration may not reflect that SCINet can perform the best among all time-series applications, it represents a well-crafted model that should be understood on its architectural design. For vigorous investigations, novel conceptual designs such as Mamba should also be considered for future research in time-series forecasting.

REFERENCES

[1] W. Liao and S. Lin, "Prediction of photovoltaic output power based on match degree and entropy weight method," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 37, no. 7, Jun. 2023, Art. no. 2350018.

[2] L. Qu, W. Li, W. Li, D. Ma, and Y. Wang, "Daily long-term traffic flow forecasting based on a deep neural network," *Exp. Syst. Appl.*, vol. 121, pp. 304–312, May 2019.

[3] S. N. Ward, "Area-based tests of long-term seismic hazard predictions," *Bull. Seismological Soc. Amer.*, vol. 85, no. 5, pp. 1285–1298, Oct. 1995.

[4] G. U. Yule, "On a method of investigating periodicities in disturbed series, with special reference to Wolfer's sunspot numbers," *Stat. Papers George Udny Yule*, vol. 226, pp. 389–420, Jan. 1971.

[5] G. T. Walker, "On periodicity in series of related terms," *Proc. Roy. Soc. A, Math., Phys. Eng. Sci.*, vol. 131, no. 818, pp. 518–532, 1931.

[6] I. Rojas, O. Valenzuela, F. Rojas, A. Guillen, L. J. Herrera, H. Pomares, L. Marquez, and M. Pasadas, "Soft-computing techniques and ARMA model for time series prediction," *Neurocomputing*, vol. 71, nos. 4–6, pp. 519–537, Jan. 2008.

[7] M. Valipour, "Critical areas of Iran for agriculture water management according to the annual rainfall," *Eur. J. Sci. Res.*, vol. 84, no. 4, pp. 600–608, 2012.

[8] Z. Liu, Y. Yan, and M. Hauskrecht, "A flexible forecasting framework for hierarchical time series with seasonal patterns: A case study of web traffic," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jun. 2018, pp. 889–892.

[9] Y. Wu, J. Ni, W. Cheng, B. Zong, D. Song, Z. Chen, Y. Liu, X. Zhang, H. Chen, and S. B. Davidson, "Dynamic Gaussian mixture based deep generative model for robust forecasting on sparse multivariate time series," in *Proc. AAAI Conf. Artif. Intell.*, May 2021, vol. 35, no. 1, pp. 651–659.

[10] L. Li, J. Yan, X. Yang, and Y. Jin, "Learning interpretable deep state space model for probabilistic time series forecasting," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 2901–2908.

[11] Z. Chen, Q. Ma, and Z. Lin, "Time-aware multi-scale RNNs for time series modeling," in *Proc. 13th Int. Joint Conf. Artif. Intell.*, Aug. 2021, pp. 2285–2291.

[12] L. Yang, T. L. J. Ng, B. Smyth, and R. Dong, "HTML: Hierarchical transformer-based multi-task learning for volatility prediction," in *Proc. Web Conf.*, Apr. 2020, pp. 441–451.

[13] Y. Li, K. Li, C. Chen, X. Zhou, Z. Zeng, and K. Li, "Modeling temporal patterns with dilated convolutions for time-series forecasting," *ACM Trans. Knowl. Discovery from Data*, vol. 16, no. 1, pp. 1–22, Feb. 2022.

[14] M. Liu, A. Zeng, M. Chen, Z. Xu, Q. Lai, L. Ma, and Q. Xu, "SciNet: Time series modeling and forecasting with sample convolution and interaction," in *Proc. 36th Conf. Neural Inf. Process. Syst.*, 2022, pp. 5816–5828.

[15] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. W. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," in *Proc. 16th Int. Joint Conf. Artif. Intell.*, Melbourne, VIC, Australia, Aug. 2017, pp. 2627–2633.

[16] R. Wen, K. Torkkola, B. Narayanaswamy, and D. Madeka, "A multi-horizon quantile recurrent forecaster," in *Proc. 31st Conf. Neural Inf. Process. Syst. (NIPS)*, Long Beach, CA, USA, 2017, pp. 1–9.

[17] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Connecting the dots: Multivariate time series forecasting with graph neural networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 753–763.

[18] Z. Pan, S. Ke, X. Yang, Y. Liang, Y. Yu, J. Zhang, and Y. Zheng, "AutoSTG: Neural architecture search for predictions of spatio-temporal graph," in *Proc. Web Conf.*, 2021, pp. 1846–1855.

[19] L. Han, B. Du, L. Sun, Y. Fu, Y. Lv, and H. Xiong, "Dynamic and multi-faceted spatio-temporal deep learning for traffic speed forecasting," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Singapore, Aug. 2021, pp. 547–555.

[20] Y. Liu, Q. Liu, J. W. Zhang, H. Feng, Z. Wang, Z. Zhou, and W. Chen, "Multivariate time-series forecasting with temporal polynomial graph neural networks," in *Proc. 36th Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 19414–19426.

[21] L. Chen, D. Chen, Z. Shang, B. Wu, C. Zheng, B. Wen, and W. Zhang, "Multi-scale adaptive graph neural network for multivariate time series forecasting," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 10, pp. 10748–10761, Oct. 2023.

[22] B. Lim, S. Ö. Arik, N. Loeff, and T. Pfister, "Temporal fusion transformers for interpretable multi-horizon time series forecasting," *Int. J. Forecasting*, vol. 37, no. 4, pp. 1748–1764, Oct. 2021.

[23] H. Y. Zhou, S. H. Zhang, J. Q. Peng, S. Zhang, J. X. Li, H. Xiong, and W. C. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 12, pp. 11106–11115.

[24] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting," in *Proc. NIPS*, vol. 34, Dec. 2021, pp. 22419–22430.

[25] S. Liu, H. Yu, C. Liao, J. Li, W. Lin, A. X. Liu, and S. Dustdar, "Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting," in *Proc. Int. Conf. Learn. Represent.*, 2021, pp. 1–20.

[26] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "FedFormer: Frequency enhanced decomposed transformer for long-term series forecasting," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 27268–27286.

[27] Y. Liu, H. X. Wu, J. M. Wang, and M. Long, "Non-stationary transformers: Exploring the stationarity in time series forecasting," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 1–13.

[28] B. Yang, L. X. Li, H. Ji, and J. Xu, "An early warning system for loan risk assessment using artificial neural networks," *Knowl.-Based Syst.*, vol. 14, nos. 5–6, pp. 303–306, Aug. 2001.

[29] I. E. Livieris, E. Pintelas, and P. Pintelas, "A CNN–LSTM model for gold price time-series forecasting," *Neural Comput. Appl.*, vol. 32, no. 23, pp. 17351–17360, Dec. 2020.

[30] J. Yoo and U. Kang, "Attention-based autoregression for accurate and efficient multivariate time series forecasting," in *Proc. SIAM Int. Conf. Data Mining (SDM)*, 2021, pp. 531–539.

[31] Y. Zhao, Y. Shen, Y. Zhu, and J. Yao, "Forecasting wavelet transformed time series with attentive neural networks," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Singapore, Nov. 2018, pp. 1452–1457.

[32] Z. H. Yue, Y. J. Wang, J. Y. Duan, T. M. Yang, C. R. Huang, Y. H. Tong, and B. X. Xu, "Ts2vec: Towards universal representation of time series," in *Proc. AAAI Conf. Artif. Intell.*, 2022, vol. 36, no. 8, pp. 8980–8987.

[33] Y. Li, H. Wang, J. Li, C. Liu, and J. Tan, "ACT: Adversarial convolutional transformer for time series forecasting," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Vancouver, BC, Canada, Jul. 2022, pp. 17105–17115.

[34] Q. Wang, L. Chen, J. Zhao, and W. Wang, "A deep granular network with adaptive unequal-length granulation strategy for long-term time series forecasting and its industrial applications," *Artif. Intell. Rev.*, vol. 53, no. 7, pp. 5353–5381, Oct. 2020.

[35] Z. Yang, W. Yan, X. Huang, and L. Mei, "Adaptive temporal-frequency network for time-series forecasting," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 4, pp. 1576–1587, Apr. 2022.

[36] Y. Qi, Q. Li, H. Karimian, and D. Liu, "A hybrid model for spatiotemporal forecasting of $PM_{2.5}$ based on graph convolutional neural network and long short-term memory," *Sci. Total Environ.*, vol. 664, pp. 1–10, May 2019.

[37] Y.-Y. Chang, F.-Y. Sun, Y.-H. Wu, and S.-D. Lin, "A memory-network based solution for multivariate time-series forecasting," 2018, *arXiv:1809.02105*.

[38] Z. Shen, Y. Zhang, J. Lu, J. Xu, and G. Xiao, "A novel time series forecasting model with deep learning," *Neurocomputing*, vol. 396, pp. 302–313, Jul. 2020.

[39] J. C. Lauffenburger, J. M. Franklin, A. A. Krumme, W. H. Shrank, O. S. Matlin, C. M. Spettell, G. Brill, and N. K. Choudhry, "Predicting adherence to chronic disease medications in patients with long-term initial medication fills using indicators of clinical events and health behaviors," *J. Managed Care Specialty Pharmacy*, vol. 24, no. 5, pp. 469–477, May 2018.

[40] N. Wu, B. Green, X. Ben, and S. O'Banion, "Deep transformer models for time series forecasting: The influenza prevalence case," 2020, *arXiv:2001.08317*.

[41] A. Zeroual, F. Harrou, A. Dairi, and Y. Sun, "Deep learning methods for forecasting COVID-19 time-series data: A comparative study," *Chaos, Solitons Fractals*, vol. 140, Nov. 2020, Art. no. 110121.

[42] K. H. Lee, J. Won, H. J. Hyun, S. C. Hahn, E. Choi, and J. H. Lee, "Self-supervised predictive coding with multimodal fusion for patient deterioration prediction in fine-grained time resolution," in *Proc. Int. Conf. Learn. Represent.*, May 2023, pp. 41–50.

[43] G. Janacek, "Time series analysis forecasting and control," *J. Time Ser. Anal.*, vol. 31, no. 4, p. 303, Jul. 2010.

[44] J.-F. Chen, W.-M. Wang, and C.-M. Huang, "Analysis of an adaptive time-series autoregressive moving-average (ARMA) model for short-term load forecasting," *Electric Power Syst. Res.*, vol. 34, no. 3, pp. 187–196, Sep. 1995.

[45] X. Y. Lu and L. H. Wang, "Bootstrap prediction interval for ARMA models with unknown orders," *Revstat-Stat. J.*, vol. 18, no. 3, pp. 375–396, 2020.

[46] C. W. J. Granger, "Investigating causal relations by econometric models and cross-spectral methods," *Econometrica*, vol. 37, no. 3, pp. 424–438, Aug. 1969.

[47] R. F. Engle and C. W. J. Granger, "Co-integration and error correction: Representation, estimation, and testing," *Econometrica*, vol. 55, no. 2, pp. 251–276, Mar. 1987.

[48] L. Zhang, J. Lin, R. Qiu, X. Hu, H. Zhang, Q. Chen, H. Tan, D. Lin, and J. Wang, "Trend analysis and forecast of $PM_{2.5}$ in Fuzhou, China using the ARIMA model," *Ecological Indicators*, vol. 95, pp. 702–710, Dec. 2018.

[49] Z. Malki, E.-S. Atlam, A. Ewis, G. Dagnew, A. R. Alzighaibi, G. Elmarhomy, M. A. Elhosseini, A. E. Hassanien, and I. Gad, "ARIMA models for predicting the end of COVID-19 pandemic and the risk of second rebound," *Neural Comput. Appl.*, vol. 33, no. 7, pp. 2929–2948, Apr. 2021.

[50] Q. Q. Shi, J. M. Yin, J. J. Cai, A. Cichocki, T. Yokota, L. Chen, M. X. Yuan, and J. Zeng, "Block Hankel tensor ARIMA for multiple short time series forecasting," 2020, *arXiv:2002.12135*.

[51] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 25, 2012, pp. 1097–1105.

[52] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," 2016, *arXiv:1609.03499*.

[53] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018, *arXiv:1803.01271*.

[54] A. Borovykh, S. Bohte, and C. W. Oosterlee, "Conditional time series forecasting with convolutional neural networks," 2017, *arXiv:1703.04691*.

[55] X. Dong, L. Qian, and L. Huang, "Short-term load forecasting in smart grid: A combined CNN and K-means clustering approach," in *Proc. IEEE Int. Conf. Big Data Smart Comput. (BigComp)*, Feb. 2017, pp. 119–125.

[56] J. M.-T. Wu, Z. Li, N. Herencsar, B. Vo, and J. C.-W. Lin, "A graph-based CNN-LSTM stock price prediction algorithm with leading indicators," *Multimedia Syst.*, vol. 29, no. 3, pp. 1751–1770, Jun. 2023.

[57] S. Huang, D. Wang, X. Wu, and A. Tang, "DSANet: Dual self-attention network for multivariate time series forecasting," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manag.*, Nov. 2019, pp. 2129–2132.

[58] J. Elman, "Finding structure in time," *Cognit. Sci.*, vol. 14, no. 2, pp. 179–211, Jun. 1990.

[59] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 15, 1997.

[60] X. Wang, J. Wu, and C. Liu, "Fault time series prediction based on LSTM recurrent neural network," *J. Beijing Univ. Aeronaut. Astronaut.*, vol. 44, no. 4, pp. 772–784, 2018.

[61] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Netw.*, vol. 18, nos. 5–6, pp. 602–610, Jul. 2005.

[62] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, Doha, Qatar, 2014, pp. 1724–1734.

[63] Y. Jung, J. Jung, B. Kim, and S. Han, "Long short-term memory recurrent neural network for modeling temporal patterns in long-term power forecasting for solar PV facilities: Case study of South Korea," *J. Cleaner Prod.*, vol. 250, Mar. 2020, Art. no. 119476.

[64] S. Chang, Y. Zhang, W. Han, M. Yu, X. Guo, W. Tan, X. Cui, M. Witbrock, M. A. Hasegawa-Johnson, and T. S. Huang, "Dilated recurrent neural networks," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst. (NIPS)*, Dec. 2017, pp. 76–86.

[65] P. Redhu and K. Kumar, "Short-term traffic flow prediction based on optimized deep learning neural network: PSO-Bi-LSTM," *Phys. A, Stat. Mech. Appl.*, vol. 625, Sep. 2023, Art. no. 129001.

[66] E. Choi, M. T. Bahadori, J. A. Kulas, A. Schuetz, W. F. Stewart, and J. Sun, "RETAIN: An interpretable predictive model for healthcare using reverse time attention mechanism," in *Proc. 30th Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 3512–3520.

[67] T. Gangopadhyay, S. Y. Tan, Z. Jiang, R. Meng, and S. Sarkar, "Spatiotemporal attention for multivariate time series prediction and interpretation," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Toronto, ONT, Canada, Jun. 2021, pp. 3560–3564.

[68] C. Fan, Y. Zhang, Y. Pan, X. Li, C. Zhang, R. Yuan, D. Wu, W. Wang, J. Pei, and H. Huang, "Multi-horizon time series forecasting with temporal attention learning," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 2527–2535.

[69] S. Du, T. Li, Y. Yang, and S.-J. Horng, "Multivariate time series forecasting via attention-based encoder–decoder framework," *Neurocomputing*, vol. 388, pp. 269–279, May 2020.

[70] H. Lin, Y. Fan, J. Zhang, and B. Bai, "REST: Reciprocal framework for spatiotemporal-coupled predictions," in *Proc. Web Conf.*, Apr. 2021, pp. 3136–3145.

[71] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, Dec. 2017, pp. 6000–6010.

[72] Y. Lin, I. Koprinska, and M. Rana, "SpringNet: Transformer and spring DTW for time series forecasting," in *Proc. Int. Conf. Neural Inf. Process. (ICONIP)*, Bangkok, Thailand, 2020, pp. 616–628.

[73] J. Lee-Thorp, J. Ainslie, I. Eckstein, and S. Ontanon, "FNet: Mixing tokens with Fourier transforms," 2021, *arXiv:2105.03824*.

[74] D. Du, B. Su, and Z. Wei, "Preformer: Predictive transformer with multi-scale segment-wise correlations for long-term time series forecasting," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Rhodes Island, Greece, Jun. 2023, pp. 1–5.

[75] K. Madhusudhanan, J. Burchert, N. Duong-Trung, S. Born, and L. Schmidt-Thieme, "U-Net inspired transformer architecture for far horizon time series forecasting," in *Proc. Joint Euro. Conf. Machine Learning Knowledge Discovery Databases*, 2022, pp. 36–52.

[76] A. L. Zeng, M. X. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?" in *Proc. 37th AAAI Conf. Artif. Intell.*, 2023, vol. 37, no. 9, pp. 11121–11128.
[77] Z. Li, Z. Rao, L. Pan, and Z. Xu, "MTS-mixers: Multivariate time series forecasting via factorized temporal and channel mixing," 2023, *arXiv:2302.04501*.
[78] A. Das, W. Kong, A. Leach, S. Mathur, R. Sen, and R. Yu, "Long-term forecasting with TiDE: Time-series dense encoder," 2023, *arXiv:2304.08424*.

WENXIANG LI is currently pursuing the Ph.D. degree in applied computer technology with the Faculty of Applied Sciences, Macao Polytechnic University, Macau, China. Her current research interests include deep learning modeling and forecast analysis of time series data.

K. L. EDDIE LAW received the B.Sc. (Eng.) degree in electrical and electronic engineering from The University of Hong Kong, the M.S. degree in electrical engineering from the Tandon School of Engineering (Polytechnic University, New York), New York University, and the Ph.D. degree in electrical and computer engineering from the University of Toronto. From 1995 to 1999, he was a Member of Scientific Staff with Nortel Networks, Ottawa, Canada, carrying research for passport switches, protocol designs, and scalable proxy web server system in the Computing Technology Laboratory. From 1999 to 2003, he was an Assistant Professor with The Edward S. Rogers Sr. Department of Electrical and Computer Engineering, University of Toronto. From 2004 to 2010, he was an Associate Professor with Toronto Metropolitan University (Ryerson University), Canada. Since 2019, he has been with the Faculty of Applied Sciences, Macao Polytechnic University, Macau. He has U.S. patents, and published refereed articles in journals, conferences, magazines, and books. Among his latest research interests, he works on data lake designs, distributed computing, computer networking, deep learning model designs, blockchains, and the IoT systems.