Survey and Taxonomy: The Role of Data-Centric AI in Transformer-Based Time Series Forecasting

Jingjing Xu[1], Caesar Wu[2], Yuan-Fang Li[3], Grégoire Danoy[1,2], Pascal Bouvry[1,2]

[1]FSTM/DCS, University of Luxembourg.
[2]SnT, University of Luxembourg .
[3]Faculty of Information Technology, Monash University.

Abstract

Alongside the continuous process of improving AI performance through the development of more sophisticated models, researchers have also focused their attention to the emerging concept of data-centric AI, which emphasizes the important role of data in a systematic machine learning training process. Nonetheless, the development of models has also continued apace. One result of this progress is the development of the Transformer Architecture, which possesses a high level of capability in multiple domains such as Natural Language Processing (NLP), Computer Vision (CV) and Time Series Forecasting (TSF). Its performance is, however, heavily dependent on input data preprocessing and output data evaluation, justifying a data-centric approach to future research. We argue that data-centric AI is essential for training AI models, particularly for transformer-based TSF models efficiently. However, there is a gap regarding the integration of transformer-based TSF and data-centric AI. This survey aims to pin down this gap via the extensive literature review based on the proposed taxonomy. We review the previous research works from a data-centric AI perspective and we intend to lay the foundation work for the future development of transformer-based architecture and data-centric AI. *

Keywords: Transformer, Time Series Forecasting, Data-Centric AI

*This work was funded by the Luxembourg National Research Fund (Fonds National de la Recherche - FNR), Grant ID 15748747 and Grant ID C21/IS/16221483/CBD. We thank to Dr. Pierrick Pochelu from University of Luxembourg and Jean-Francois Nies from Luxembourg Institute of Science and Technology (LIST) for proofreading. We thank to Lucas Villiere from University of Luxembourg for measuring the carbon footprint of the models. Also, we thank to HPC group of the University of Luxembourg

1 Introduction

Sequential data like languages and time series can be effectively modeled using recurrent neural networks (RNNs) [1], which connect features across different points in a sequence. Traditional fully connected neural networks (FCNs) lack this capability. However, RNNs face challenges such as vanishing and exploding gradients, which impede their ability to capture long-term dependencies. Solutions include using Rectified Linear Unit (ReLU) activation functions, initializing weights with identity matrices, and implementing gates to control information flow. Architectures like Gated Recurrent Units (GRUs) [2] and Long Short-Term Memory (LSTM) [3] networks use gate cells to manage long-term dependencies, but some challenges remain, particularly in training, which often considers only one direction of data leading to capture less detailed context. Bidirectional RNNs address this by processing data in both forward and reverse directions. Despite improvements, these models lack parallelism, leading to inefficiencies.

The Transformer [4] architecture offers an attractive solution to these issues by processing sequence data in parallel through an encoder-decoder architecture and applying multi-head attention mechanisms, thereby significantly enhancing efficiency and performance. Transformers have proven to be highly effective in processing long-term sequence data, such as lengthy sentences and speech in NLP [5] applications, as well as handling image and video data in CV applications [6, 7]. Their capabilities in the realm of long time series forecasting have also been recognized.

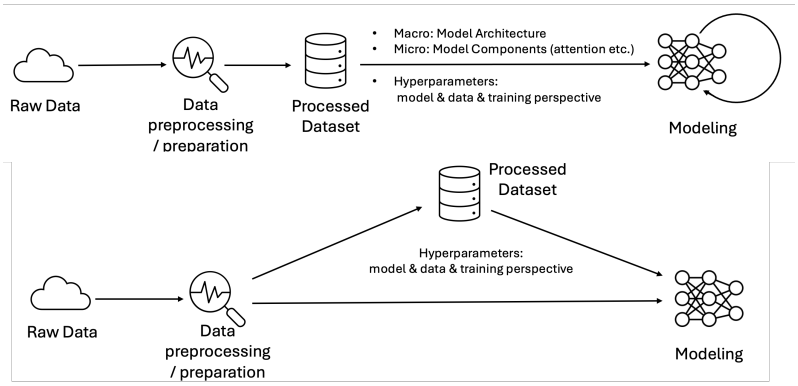

Fig. 1 MCAI (top) and DCAI (bottom) Iteration Life Cycles.

Nevertheless, the improvements made by these model-centric AI approaches (MCAI) can not always overcome insufficient data preparation and preprocessing. These steps, which are essential for developing precise models, were often neglected. Therefore, researchers are increasingly directing their efforts towards data-centric approaches to achieve better model performance, leading to the emergence of Data-Centric AI (DCAI) [8–13]. According to Andrew Ng, data-centric AI is the discipline of systematically engineering the data used to build an AI system [14]. It emphasizes the importance of data in the analysis compared to conventional model-centric AI. We

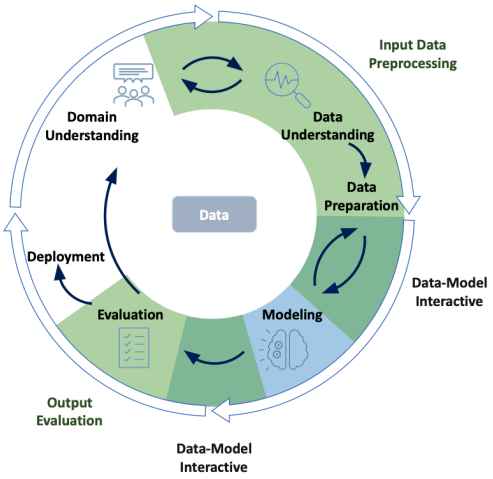

Fig. 2 Different phases of CRISP-DM in Data-Centric AI perspective.

illustrate the processes of data-centric AI (DCAI) and model-centric AI (MCAI) in Fig. 1. However, existing surveys and tutorials focus on explaining the role of MCAI in transformer-based time series models.Consequently, there is a lack of surveys on DCAI in transformer-based time series models. (N.B. In this paper, the terms transformer architecture and transformer model are used interchangeably.)

To address this gap, we apply the CRoss-Industry Standard Process for Data Mining (CRISP-DM) (see Fig. 2) to build a taxonomy (Sec. 3) for data-centric time series forecasting. CRISP-DM is viewed as an AI system-building process, dividing the process into six major phases: domain (business) understanding, data understanding, data preparation (preprocessing), modeling, evaluation, and deployment [15]. We categorize these processes into Input data preprocessing, data-model interaction, and output evaluation in our taxonomy (See Fig. 3). Our contributions encompass several key aspects. We propose a new taxonomy based on data-centric AI and address the following Research Questions (RQs):

RQ 1: *How are datasets preprocessed before being fed into models?*
RQ 2: *How does the data and model interact with each other?*
RQ 3: *How are models evaluated on the data?*

We propose **RQ1** because the initial step of constructing a model involves a thorough analysis of the time series data [16, 17]. Without this fundamental understanding, the justification for model construction becomes uncertain, as models typically do not perform well with random and unstructured datasets. Furthermore, applying the model effectively and efficiently with given datasets in practical forecasting tasks is challenging. Therefore, addressing **RQ2** is crucial for building an effective and efficient model. To successfully implement models in practical forecasting applications, it is necessary to evaluate their performance, making **RQ3** vital. The paper is organized as follows: Section 2 describes related surveys. Section 3 provides the taxonomy.

Section 4 addresses RQ1, Section 5 addresses RQ2, and Section 6 addresses RQ3. Section 7 indicate future research opportunities. Section 8 concludes the paper.

2 Related Works and Background

2.1 Data-Centric Artificial Intelligence (Data-Centric AI)

With the development and movement [18, 19] of Data-Centric AI (DCAI), several surveys have emerged on this topic. However, none have specifically addressed the role of DCAI in Transformer models for time series forecasting. Some researchers [10] have introduced the concepts and principles to outline the foundations of DCAI. Others [9] have defined relevant terms and introduced a framework for DCAI, while another study [12] provides a overview of DCAI missions, detailing definitions, explanations, related tasks, and challenges. These surveys primarily focus on clarifying the definitions and frameworks or providing an overview of DCAI, without delving into detailed methods for each section. One comprehensive survey [13] examines the entire data lifecycle with representative methods but does not discuss specific use cases. Another survey [20] explores use cases with graph learning, and another [21] provides an epidemic forecasting survey from a DCAI perspective. However, none of these involve Transformer models. One paper [22] studies data movement in Transformers' training processes to improve GPU utilization, but it does not consider the data itself. Another study [23] addresses the performance issues of Vision Transformers (ViTs) in face recognition scenarios from a data-centric perspective. Nevertheless, the role of DCAI in Transformer-based time series forecasting remains insufficiently explored.

2.2 Transformer for Time Series Forecasting

2.2.1 Time Series Forecasting

Surveys [24] and tutorials [25] discuss deep learning for time series forecasting from the perspective of model architectures, while another review [26] conducts experimental studies to compare the performance of different deep learning architectures. The Monash time series forecasting archive [27] provides a diverse collection of comprehensive time-series datasets across various domains, along with dataset characteristics analysis. However, these surveys do not conduct comprehensive analyses of Transformers in time series forecasting, and their dataset analysis and analysis pipelines are deficient.

2.2.2 Transformer for Time Series

The use of Transformers for time series tasks arises from their powerful capability in handling sequential data. A survey [28] analyzes the development of time series Transformers from network modifications and application domain perspectives. A tutorial [29] provides details about Transformer architecture and its applications. However, these studies approach the topic from a model-centric AI perspective, leaving the survey of Transformers in time series from a data-centric AI perspective insufficient. Thus, we introduce the role of DCAI in Transformer-based time series forecasting to fill

this gap. We particularly focus on explaining data-model interaction in Transformer architecture. Details on this interaction in time series are introduced in Sec. 5, covering input time series data representation part and the modeling part. Transformer data representation includes input embedding and position encoding, crucial for time series due to the importance of position order. A review [30] provides an overview of position information in Transformers to help choose suitable position encoding solutions. Another paper [31] studies different position encoding and proposes a solution for multivariate time series classification called ConvTran. Nevertheless, the study of input embedding and position encoding for time series forecasting still requires further exploration.

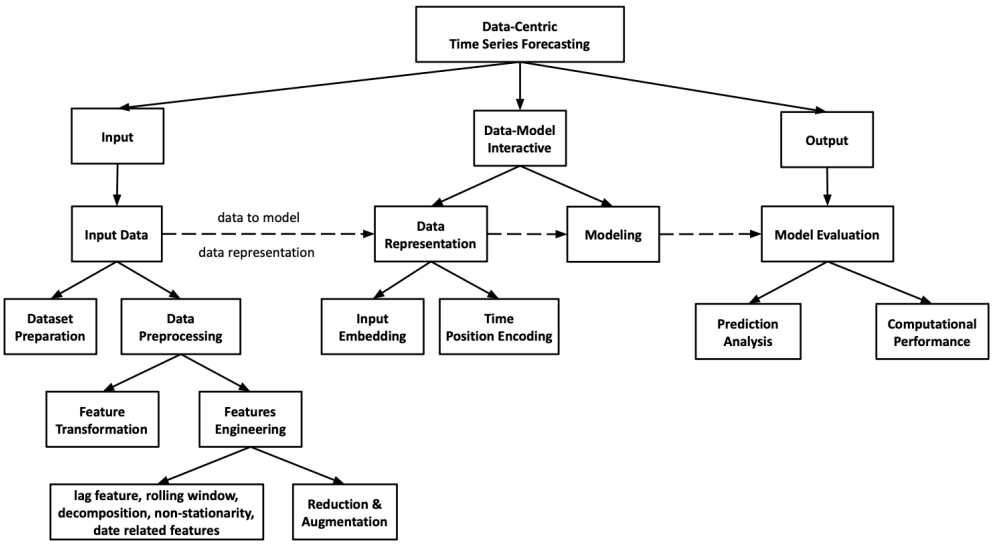

Fig. 3 Taxonomy of Data-Centric Transformer-based Time Series Forecasting

3 Taxonomy

We adopt the CRISP-DM and Transformer model workflow as the foundational framework to systematically structure the different phases of our survey, as illustrated in Figure 3. Input Data, Data-Model Interaction, and Output Evaluation will be detailed in Sections 4, 5, and 6 respectively. In the Input Data section, we cover data preparation and preprocessing to address RQ1: *How are datasets preprocessed before being fed into models?* In the Data-Model Interaction section, we discuss embedding, encoding, and modeling to address RQ2: *How does the data and model interact with each other?* In the Output Evaluation section, we explain model evaluation to address RQ3: *How are models evaluated on the data?*

Table 1 Time Series Forecasting Models and Datasets (Open-Source Transformer model and LLM until 2023. Nov. 01)

Domain	Model	Transformer																				LLM				Frequency
	Abbreviation	GCformer [32]	PDTrans [33]	PatchTST [34]	Crossformer [35]	Scaleformer [36]	Preformer [37]	Client [38]	Taylorformer [39]	iTransformer [40]	TDformer [41]	Non-stationary Trans. [42]	FEDformer [43]	TACTiS [44]	Pyraformer [45]	TCCT [46]	Triformer [47]	Autoformer [48]	Informer [49]	TFT [50]	AST [51]	PromptCast [52]	One Fits All [53]	LLMTime [54]	Lag-Llama [55]	
	Implement	code	code	code	code	code	code	code	code	code	code	code	code	code	code	code	code	code	code	code	code	code	code	code	code	
Energy	Electricity	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	24
Transport	Traffic Hourly	√	√	√	√	√	√	√		√	√	√	√	√		√		√		√	√		√	√		18
Finance	Exchange	√	√			√	√	√	√		√	√	√					√						√	√	12
Energy	ETTm1	√		√	√			√	√		√				√	√	√	√	√				√			12
Nature	Weather_DE	√		√		√	√	√			√	√	√					√					√	√		11
Energy	ETTh1			√	√			√		√					√	√	√	√	√				√			10
Energy	ETTm2	√		√			√	√				√	√			√		√					√	√		10
Health	ILI	√		√	√	√	√	√				√	√					√					√			10
Energy	ETTh2			√				√		√						√		√	√				√			7
Energy	Solar 10 minutes		√							√				√							√			√	√	6
Nature	Weather_US				√												√		√							3
Finance	Fred-MD													√										√	√	3
Multiple	M4 Hourly		√																		√				√	3
Nature	Wind_EU														√						√					2
Health	Covid Deaths																							√	√	2
Tourism	Tourism Monthly																							√	√	2
Energy	Aus. Electricity																							√	√	2
Transport	Pedestrian Counts																							√	√	2
Health	Hospital																							√	√	2
Banking	NN5 weekly																							√	√	2
Nature	Saugeen River Flow																							√	√	2
Finance	CIF 2016																							√	√	2
Nature	KDD Cup 2018													√											√	2
Transport	PEMS									√																1
Sales	Retail																			√						1
Finance	Volatility																			√						1
Web	App Flow														√											1
Nature	City Temperature																				√					1
Nature	SafeGraph Human Mobility																				√					1
Energy	Solar Weekly																						√			1
Tourism	Tourism Yearly																						√			1
Tourism	Tourism Quarterly																						√			1
Nature	US Births																						√			1
Transport	Traffic Weekly																						√			1
Sales	Car Parts																								√	1
Web	Web Traffic																								√	1
Energy	London Smart Meters																								√	1
Transport	Rideshare																								√	1
Transport	Vehicle Trips																								√	1
Nature	Temperature Rain																								√	1
Nature	Weather_AU																								√	1
Multiple	M1 Daily																								√	1
Multiple	M4 Monthly																								√	1
Multiple	M4 Quarterly																								√	1
Multiple	M4 Yearly																								√	1
Energy	Wind_Farms_AU																								√	1
Transport	Taxi																								√	1
Transport	Uber																								√	1
Web	Wikipedia																								√	1
Transport	Air Passengers																								√	1
	Total	7	5	9	6	5	6	9	3	6	5	6	6	5	5	6	4	9	5	4	5	3	8	20	26	

4 Input Data: How are datasets preprocessed before being fed into models?

4.1 Dataset Preparation

We gather datasets from typical models, utilizing open-access papers and open-source codes, resulting in 50 datasets across 24 models [1]. These include 20 transformer models and 4 Large Language Models (LLMs) from 2020 to 2023. The transformer, an encoder-decoder architecture, is applied to various problems [56] such as NLP tasks, CV, and audio/speech processing. LLMs, which utilize transformer architecture, are pre-trained on large text datasets for NLP tasks. A recent survey [57] explores LLMs for time series data, making them relevant for comparison. The datasets and related models are shown in Tab. 1, which indicates that LLM-based time series models use more datasets than transformer-based time series models. These datasets do not have missing or corrupted data. Typically, missing data can be addressed through imputation, and corrupted data can be detected using anomaly detection algorithms. Most datasets for transformers-based TSF models are split into training, validation, and

[1] We include the source of the code and link of dataset in the Tab. 1.

test sets, commonly at a 70/20/10 ratio, although this can vary. Papers [58, 59] discuss optimal data splitting ratios, but these are not well-explored for transformer-based TSF models.

4.2 Data Preprocessing

Once data preparation has been completed, the next stage is data preprocessing. In this section, we follow the sequence of steps from the raw dataset to the model input. These steps are crucial for model performance to avoid "Rubbish in, rubbish out". The preprocessed data (model input) is then passed to the model [60]. In transformer-based time series models, data undergoes sequential preprocessing: organizing features, data reduction or augmentation, and data representation for the model. (N.B. Data representation (input embedding and position encoding) is part of the data-model interaction, discussed in the Sec. 5.)

4.2.1 Data Features

Data Features includes *feature transformations* and *feature engineering*. *Feature transformations* transform a dataset into new distribution base on model's assumption. *Feature Engineering* extracts features from input datasets to improve the performance of the models. A common feature transformation is data normalization, which is frequently used in transformer-based time series forecasting (TSF) models. Models apply data normalization to adjust data to the same common scale or range. This keeps different datasets and models on the same level for comparison. There are several normalization solutions such as Z-normalization, Min-max normalization, Sigmoid normalization etc. [61]. However, further research is needed to analyze the different data normalization methods used in transformer-based time series models. *Feature Engineering* allows us to understand different features of time series data, which is essential for the performance of transformer-based forecasting models. Different features are discussed in [16, 62, 63]. Here, we focus on features applied in transformer-based time series forecasting. Below, we list some common feature engineering methods for transformer-based TSF models specifically.

•`Covariate:` In Dart [64], covariate time series refer to external data or variables that are not the target of forecasting but are useful for improving forecasting accuracy. For example, when modeling participants' heart rates using their weight, additional factors such as environmental temperature and measurement time also influence heart rates. These factors are called covariates. The meaning of covariates can vary depending on the context. In some contexts, input features or explanatory variables are considered covariates [65], similar to their definition in statistical dictionaries [66]. The paper by Davies [67] discusses the role of covariates in forecasting models. The Temporal Fusion Transformer (TFT) model [51] employs static covariate encoders to integrate covariates into the model.

•`Lag Features and Sliding/Rolling Window: Time lag` refers to previous steps in the time series. An example of lag is shown in Fig.4. Lag-Llama[68] apply time lags as covariates to build the forecasting model. The `Rolling Window` technique involves moving a window of specified length across the data sequentially. In traditional

forecasting methods, statistics such as mean, median, and maximum are computed over a fixed-size sliding window. Fig. 4 illustrates this approach. Transformer-based forecasting models like Informer [49] utilize the sliding window method to construct the input dataset. An example demonstrating the use of a sliding window for encoder and decoder inputs is depicted in Fig. 5.

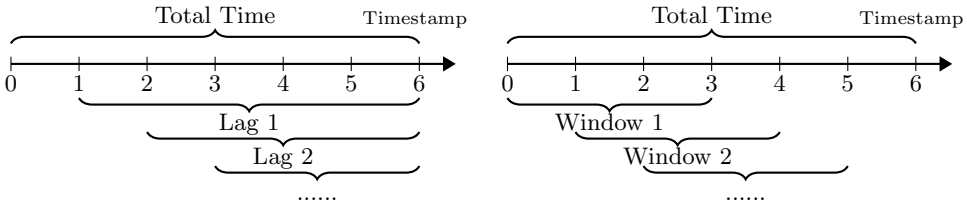

Fig. 4 Left: Example of the Lag in Time Series. Right: Example of the Rolling Window (window size=3).

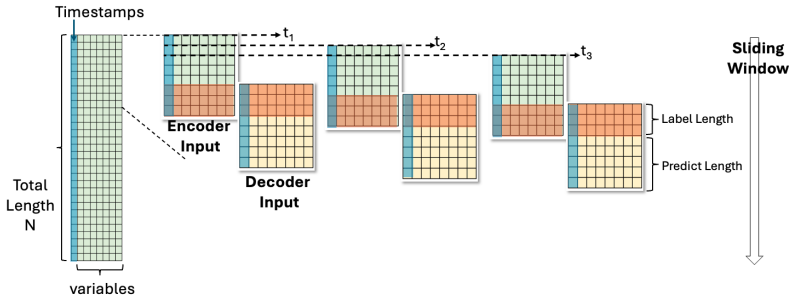

Fig. 5 Using Sliding Window to Prepare Encoder Input and Decoder Input for Transformer-based Time Series Models.

•`Seasonal Decomposition:` Time series can generally be decomposed into three main components: Trend, Seasonal pattern, and Residual part [69]. The Trend component captures long-term increasing or decreasing patterns in the data. Seasonal pattern reflects periodic influences such as those caused by seasonal variations. The Residual component represents the remaining noise after extracting the Trend and Seasonal patterns. Models like FEDformer [43] and TDformer [41] leverage this decomposition approach in their modeling strategies.

•`Stationarity and non-stationarity:` Formally, let $X_t = \{x_{t_1}, \ldots, x_{t_n}\}$ denote a time series, where $D(X_t)$ represents its distribution function. τ denotes the time interval. A time series X_t is stationary if $D(X_{t+\tau})$ does not depend on the observation time t. It means that time series behaves stochastically at any point in time. In contrast, non-stationary time series are influenced by trends or seasonality, exhibiting predictable patterns. Non-stationary Transformers [42] incorporate the concept of non-stationarity in data to inform their modeling approach.

•`Date Related:` In time series analysis, timestamps can range from seconds and minutes to hours, or calendar-based intervals such as days, weeks, and months. Each

interval can reveal distinct patterns within the time series data. Moreover, factors like seasonality, holidays, and weekends introduce cyclical patterns. Date-related features serve as covariates in modeling these patterns.

4.2.2 Data Reduction & Data Augmentation

Data Reduction involves transforming the original data into a corrected and simplified form, often by cleaning up invalid data or generating summaries of the original dataset [70]. Building upon the idea of summarization, PCATransformer [71] introduce Principal Component Analysis (PCA) based transformer TSF models, marking the initial exploration of data reduction in these models. However, research in this area remains limited. In contrast, *Data Augmentation* is a well-explored technique used to enhance the size and quality of training datasets, thereby reducing overfitting in deep learning models such as transformers [72]. The survey [73] reviews various data augmentation methods for different time series tasks to summarize the augmentation methods.

5 Data-Model Interaction: How does the data and model interact with each other?

In this section, our aim is to address the interaction between data and the model. Specifically, we will focus on answering the question: *How was the data prepared for the model, particularly for the encoder and decoder components?* In transformer-based forecasting architectures, the encoder-decoder framework takes a given time series as input and produces a predicted time series as output. While the general flow of data through a transformer model is described in the paper [22], emphasizing hardware-level data movement during training, our paper primarily focuses on the data representation process within the transformer model.

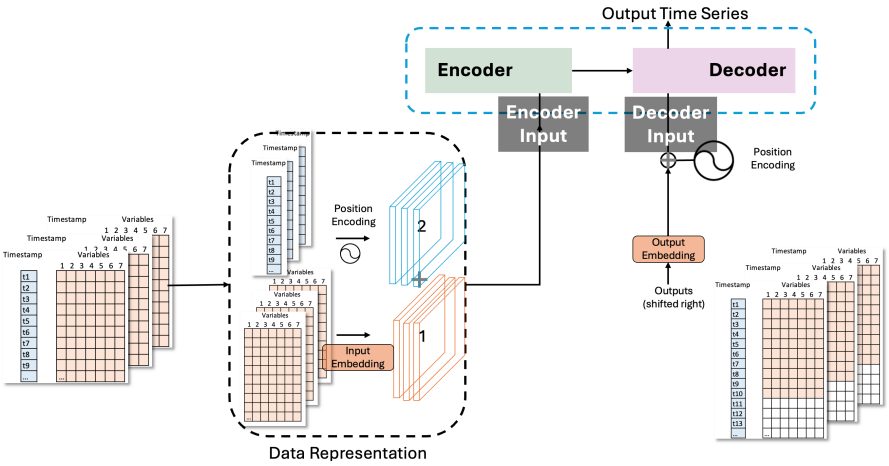

Fig. 6 Input Embedding and Time Position Encoding

5.1 Data Representation: Input Embedding & Time Position Encoding

Fig. 6 illustrates the input embedding and time position encoding components in the transformer-based time series model. The inputs shown in the figure are preprocessed data prepared using sliding windows from Fig. 5. This preprocessed data includes a matrix of variables (features) and their corresponding timestamps. The variables (features) are represented using input embedding (the output is "1" in the Fig. 6), while the timestamps are represented using time position encoding (the output is "2" in the Fig. 6). Both representation processes result in matrices, which are then merged to form the input for the encoder. The input for the decoder is generated in the same manner. Input embedding techniques commonly include 1-D convolutional filters (with a kernel width of 3) on the input data, as used in models like Informer [49] and Autoformer [48], and linear projection methods, as seen in PatchTST [74]. Time position encoding typically applies temporal position encoding to represent timestamp information. The survey [30] provides a comprehensive overview of existing position information methods in Transformer models, while the paper[31] discuss various position encoding solutions for transformer-based time series models. The common sinusoidal position encoding is presented as follows, where t is the $t-th$ timestamp, j is the $j-th$ dimension of the model, d is the total dimension of the model.

$$P_{tj} = \begin{cases} sin(10000^{-\frac{j}{d}}t), & j \in 2n : n \in \mathbb{Z} \\ cos(10000^{-\frac{j-1}{d}}t), & j \in 2n+1 : n \in \mathbb{Z} \end{cases} \tag{1}$$

5.2 Modeling

After data representation, the resulting matrices are fed into the model's first layer, known as the attention layer. This marks the end of the data-model interaction. The main components of the transformer model (Attention, Add & Norm, and Feed Forward) are illustrated in Fig. 7. In the context of MCAI, transformer-based time series models are classified into different types based on modifications to these components and the overall architecture [40, 71].

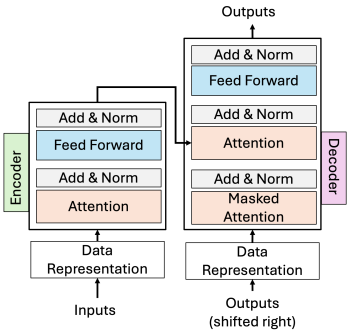

Fig. 7 Main Components (Attention, Add&Norm, Feed Forward) of the Transformer Model

6 Output Evaluation: How are models evaluated on the data?

Transformer-based time series models are generally evaluated from two perspectives: predictive performance and computational performance. Predictive performance is represented by metrics such as Mean Squared Error (MSE) and Mean Absolute Error (MAE). Computational performance is quantified by memory usage and computation time. These aspects are summarized in Table 2. In this part, we summarize metrics used in transformer-based forecasting models and we discuss the computational performance measurement solution.

Table 2 Predictive Performance metrics and efficiency measurement solutions of transformer-based time series models

Model Abbrev.	Predictive Performance	Efficiency Measurements
GCformer	MSE, MAE	training speed
PDTrans	Quantile Loss	
PatchTST	MSE, MAE	runtime
Crossformer	MSE, MAE	memeory, runtime
Scaleformer	MSE, MAE	memeory, runtime
Preformer	MSE, MAE	memeory, runtime
Client	MSE, MAE	memeory, runtime, parameter quantity
Taylorformer	Likelihood, MSE	
iTransformer	MSE, MAE	memory
TDformer	MSE, MAE	
Non-stationary Trans.	MSE, MAE	
FEDformer	MSE, MAE	
TACTiS	CRPS, Energy Score	
Pyraformer	MSE, MAE	memeory, runtime, Q-K pairs
TCCT	MSE, MAE	memeory, runtime
Triformer	MSE, MAE	memeory
Autoformer	MSE, MAE	memeory, runtime
Informer	MSE, MAE	runtime
TST	Quantile Loss	
AST	Quantile Loss	

6.1 Predictive Performance

6.1.1 Mean Squared Error (MSE) and Mean Absolute Error (MAE)

MSE and MAE are used to measure the difference between predicted values and actual values. Lower values of MSE and MAE indicate higher accuracy of the model. Considering $\hat{y}_i$ is predicted values and y_i is the ground truth. The MSE and MAE is denoted as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2, MAE = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i| \tag{2}$$

6.1.2 Quantile Loss (ρ-risk)

Quantiles divide a dataset into equal parts. Quantile is defined as $Q(\rho) = inf\{x : F(x) \geq \rho\}, 0 < \rho < 1$, where $F(x)$ is the distribution function [75]. Normalized Quantile Loss, commonly used in transformer-based models, is discussed in [76, 77].

The ρ-quantile loss for $\rho \in (0,1)$ is defined as follow, where $\hat{x}_i$ is predicted values and x_i is the ground truth. In the equation, if $x \geq \hat{x}$, then $P_\rho = \rho(x - \hat{x})$. This implies that larger values of ρ assign more weight to the loss when $x \geq \hat{x}$.

$$L_\rho(x, \hat{x}) = 2\frac{\sum_{i=1}^{n} P_\rho(x_i, \hat{x}_i)}{\sum_{i=1}^{n} x_i}, P_\rho = \begin{cases} \rho(x - \hat{x}), x \geq \hat{x} \\ (1-\rho)(\hat{x} - x), x < \hat{x} \end{cases} \tag{3}$$

6.1.3 Likelihood

The likelihood function is denoted as $L(\theta|X) = L(\theta|x_1, ..., x_n) = f(x_1, ..., x_n|\theta)$ when $x_1, ..., x_n$ has density function $f(x_1, ..., x_n|\theta)$ [78]. Likelihood function $L(\theta|X)$ is the function of the parameters θ of the model. Taylorformer [39]'s likelihood function is denoted as $L_\theta(Y_T|Y_C, X_C, X_T)$, where $\{X_C, Y_C\}$ is context set and $\{X_T, Y_T\}$ is target set. It evaluates the model based on likelihood method and LocalTaylor function and neural network MHA-X-Net.

6.1.4 Uncertainty Estimation

Unlike scoring function such as MSE and MAE, which measure the prediction quality, uncertainty estimation express measure the degree of the model's confidence in its own prediction. CRPS (Continuous Ranked Probability Score) is widely used, as it offer the evaluation across the entire predictive distribution. The CRPS [79–81] is defined as follows:

$$CRPS(F, x) = \int (F(\hat{x}) - H(\hat{x} - x))^2 d\hat{x}, H(\hat{x} - x) = \begin{cases} 0, x \geq \hat{x} \\ 1, x < \hat{x} \end{cases} \tag{4}$$

where $x \in \mathbb{R}$ is the observation, F is the cumulative distribution function of the forecast distribution. As one variation of CRPS, TACTiS [44] employs the CRPS calculation solution from GluonTS [82], which exploits the fact that CRPS is equal to twice the mean quantile loss (as per [81]). Another related metric is the Energy Score [80] which characterizes the equality of distributions [83].

6.2 Computational Performance

Most papers record memory usage, runtime, or speed to measure a model's efficiency. Pyraformer [45] uses the number of query-key dot products (Q-K pairs) to describe the time and space complexity of the model. However, with the growing importance of sustainability in the face of climate change, it is crucial to develop models with lower environmental impact. Therefore, we also measure the carbon footprint of models (See Appendix. A) to raise awareness and encourage the development of sustainable models.

7 Future Research Opportunities

7.1 Input Dataset

The Monash Time Series Forecasting Archive [27] offers a diverse range of comprehensive time-series datasets across various domains, accompanied by dataset characteristic analyses. However, there is still a lack of investigation of dataset exploration pipelines tailored specifically for transformer-based time series models. Additionally, research on determining optimal split ratios (training, validation, and testing) remains insufficient, with only a few papers such as [58, 59] discussing the subject. Moreover, while researchers have begun focusing on dataset augmentation in transformer-based time series models [73], dataset reduction for transformer models has so far been neglected. Dataset reduction is crucial due to the complexity (long timestamps with large multiple variables) of time series datasets. Furthermore, most transformer models primarily operate on common datasets such as electricity consumption, traffic, and exchange rates (as shown in Table 1). Some recent papers [84, 85] have explored transformer-based time series models in the financial domain. However, further investigation into the application of transformer-based time series models on diverse real-world datasets is desirable. Real-world datasets are often a mix of various data types, such as medical records including patients' images or speech data alongside numerical measurements [86]. As researchers are also delving into spatio-temporal data [57, 87], this signals a growing interest in transformer-based multimodal forecasting for using various data types.

7.2 Input Data Representation

In this paper, the input data representation includes the input embedding and time position encoding. Another survey paper [30] has explored position information methods in transformer models, highlighting the importance in capturing sequential relationships effectively. Recent research in particular [31] has studied these methods to suit temporal data under the context of transformer-based time series models. Despite advancements in position encoding techniques, research on optimizing input embedding strategies tailored for transformer-based time series models remains insufficient.

7.3 Evaluation

Most transformer-based models use Mean Squared Error (MSE) and Mean Absolute Error (MAE) to assess forecast errors between predicted and observed values. Some models employ quantile loss to evaluate prediction intervals, while others utilize probability-based metrics such as likelihood and Continuous Ranked Probability Score (CRPS) to assess the alignment of predicted probability distributions with observed values. However, understanding the reliability and trustworthiness of these models remains challenging due to the opacity of neural networks within transformers. Methods for identifying, quantifying, and communicating uncertainties in model outputs are discussed in the book [88]. Surveys [89] explore uncertainty management techniques in NLP from both data and model perspectives. However, uncertainty management in

transformer-based time series haven't been investigated enough. A recent paper [90] quantified uncertainty in transformer-based TSF models using glucose datasets. Robformer [91] innovates with a robust decomposition module to address trend shifting. However, generalizing these results to other datasets and modules is decisive.

8 Conclusion

This paper has explored the role of data-centric AI in transformer-based time series forecasting by addressing three key research questions and proposing a taxonomy. Firstly, in the Input Data section, we addressed RQ1 *How datasets are preprocessed before being fed into models?* by discussing the data preparation and preprocessing in transformer-based time series forecasting. Secondly, in the Data-Model Interaction section, we answered RQ2 *How does the data and model interact with each other?* by delving into the data representation within transformer-based time series forecasting models. Finally, in the Output Evaluation section, we addressed RQ3 *How are models evaluated based on the data?* Furthermore, we highlight future research opportunities based on these three reseach questions.

References

[1] Schuster, M., Paliwal, K.K.: Bidirectional recurrent neural networks. IEEE transactions on Signal Processing (1997)

[2] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using rnn encoder-decoder for statistical machine translation. EMNLP (2014)

[3] Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation (1997)

[4] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. NeurIPS (2017)

[5] Gillioz, A., Casas, J., Mugellini, E., Abou Khaled, O.: Overview of the transformer-based models for nlp tasks. In: FedCSIS (2020). IEEE

[6] Han, K., Wang, Y., Chen, H., Chen, X., Guo, J., Liu, Z., Tang, Y., Xiao, A., Xu, C., Xu, Y., et al.: A survey on vision transformer. PAMI (2022)

[7] Nies, J.-F., Rizvi, S.T.R., Munir, M., Elst, L., Dengel, A.: Knowledge-Aware Object Detection in Traffic Scenes. In: ICAART (2024)

[8] Polyzotis, N., Zaharia, M.: What can data-centric ai learn from data and ml engineering? arXiv preprint arXiv:2112.06439 (2021)

[9] Jakubik, J., Vössing, M., Kühl, N., Walk, J., Satzger, G.: Data-centric artificial intelligence. Business & Information Systems Engineering (2024)

[10] Jarrahi, M.H., Memariani, A., Guha, S.: The principles of data-centric ai (dcai). arXiv preprint arXiv:2211.14611 (2022)

[11] Whang, S.E., Roh, Y., Song, H., Lee, J.-G.: Data collection and quality challenges in deep learning: A data-centric ai perspective. The VLDB Journal (2023)

[12] Zha, D., Bhat, Z.P., Lai, K.-H., Yang, F., Hu, X.: Data-centric ai: Perspectives and challenges. In: SDM (2023). SIAM

[13] Zha, D., Bhat, Z.P., Lai, K.-H., Yang, F., Jiang, Z., Zhong, S., Hu, X.: Data-centric artificial intelligence: A survey. arXiv preprint arXiv:2303.10158 (2023)

[14] Data-Centric, A.: Data-Centric AI Resource Hub. URL: https://datacentricai.org (2021)

[15] Wirth, R., Hipp, J.: Crisp-dm: Towards a standard process model for data mining. In: PAKDD (2000)

[16] Hyndman, R.J., Athanasopoulos, G.: Forecasting: Principles and Practice. OTexts, Australia (2018)

[17] Abedjan, Z., Golab, L., Naumann, F.: Data profiling: A tutorial. In: SIGMOD (2017)

[18] Ng, A., Laird, D., He, L.: Data-centric ai competition. URL: https://https-deeplearning-ai.github.io/data-centric-comp/ (2021)

[19] Ng, A.: A Chat with Andrew on MLOps: From model-centric to data-centric AI. URL: https://youtu.be/06-AZXmwHjo (2021)

[20] Yang, C., Bo, D., Liu, J., Peng, Y., Chen, B., Dai, H., Sun, A., Yu, Y., Xiao, Y., Zhang, Q., et al.: Data-centric Graph Learning: A Survey. arXiv preprint arXiv:2310.04987 (2023)

[21] Rodríguez, A., Kamarthi, H., Agarwal, P., Ho, J., Patel, M., Sapre, S., Prakash, B.A.: Data-centric epidemic forecasting: A survey. arXiv preprint arXiv:2207.09370 (2022)

[22] Ivanov, A., Dryden, N., Ben-Nun, T., Li, S., Hoefler, T.: Data movement is all you need: A case study on optimizing transformers. MLSys (2021)

[23] Dan, J., Liu, Y., Xie, H., Deng, J., Xie, H., Xie, X., Sun, B.: Transface: Calibrating transformer training for face recognition from a data-centric perspective. In: ICCV (2023)

[24] Lim, B., Zohren, S.: Time-series forecasting with deep learning: a survey. Philosophical Transactions of the Royal Society A (2021)

[25] Benidis, K., Rangapuram, S.S., Flunkert, V., Wang, Y., Maddix, D., Turkmen, C., Gasthaus, J., Bohlke-Schneider, M., Salinas, D., Stella, L., et al.: Deep learning for time series forecasting: Tutorial and literature survey. ACM Computing Surveys (2022)

[26] Lara-Benítez, P., Carranza-García, M., Riquelme, J.C.: An experimental review on deep learning architectures for time series forecasting. International journal of neural systems (2021)

[27] Godahewa, R., Bergmeir, C., Webb, G.I., Hyndman, R.J., Montero-Manso, P.: Monash time series forecasting archive. In: NeurIPS (2021)

[28] Wen, Q., Zhou, T., Zhang, C., Chen, W., Ma, Z., Yan, J., Sun, L.: Transformers in time series: a survey. In: IJCAI (2023)

[29] Ahmed, S., Nielsen, I.E., Tripathi, A., Siddiqui, S., Ramachandran, R.P., Rasool, G.: Transformers in time-series analysis: A tutorial. CCSP (2023)

[30] Dufter, P., Schmitt, M., Schütze, H.: Position information in transformers: An overview. Computational Linguistics (2022)

[31] Foumani, N.M., Tan, C.W., Webb, G.I., Salehi, M.: Improving position encoding of transformers for multivariate time series classification. Data Mining and Knowledge Discovery (2024)

[32] Zhao, Y., Ma, Z., Zhou, T., Ye, M., Sun, L., Qian, Y.: Gcformer: an efficient solution for accurate and scalable long-term multivariate time series forecasting. In: CIKM (2023)

[33] Tong, J., Xie, L., Zhang, K.: Probabilistic decomposition transformer for time series forecasting. In: SDM (2023). SIAM

[34] Nie, Y., Nguyen, N.H., Sinthong, P., Kalagnanam, J.: A time series is worth 64 words: Long-term forecasting with transformers. In: ICLR (2022)

[35] Zhang, Y., Yan, J.: Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In: ICLR (2022)

[36] Shabani, M.A., Abdi, A.H., Meng, L., Sylvain, T.: Scaleformer: Iterative multi-scale refining transformers for time series forecasting. In: ICLR (2022)

[37] Du, D., Su, B., Wei, Z.: Preformer: predictive transformer with multi-scale segment-wise correlations for long-term time series forecasting. In: ICASSP (2023)

[38] Gao, J., Hu, W., Chen, Y.: Client: Cross-variable linear integrated enhanced transformer for multivariate long-term time series forecasting. arXiv preprint arXiv:2305.18838 (2023)

[39] Nivron, O., Parthipan, R., Wischik, D.J.: Taylorformer: Probabilistic predictions for time series and other processes. arXiv preprint arXiv:2305.19141 (2023)

[40] Liu, Y., Hu, T., Zhang, H., Wu, H., Wang, S., Ma, L., Long, M.: itransformer: Inverted transformers are effective for time series forecasting. In: ICLR (2023)

[41] Zhang, X., Jin, X., Gopalswamy, K., Gupta, G., Park, Y., Shi, X., Wang, H., Maddix, D.C., Wang, B.: First de-trend then attend: Rethinking attention for time-series forecasting. In: NeurIPS'22 Workshop (2022)

[42] Liu, Y., Wu, H., Wang, J., Long, M.: Non-stationary transformers: Exploring the stationarity in time series forecasting. NeurIPS (2022)

[43] Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., Jin, R.: Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In: ICML

[44] Drouin, A., Marcotte, É., Chapados, N.: Tactis: Transformer-attentional copulas for time series. In: ICML (2022)

[45] Liu, S., Yu, H., Liao, C., Li, J., Lin, W., Liu, A.X., Dustdar, S.: Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In: ICLR (2021)

[46] Shen, L., Wang, Y.: Tcct: Tightly-coupled convolutional transformer on time series forecasting. Neurocomputing (2022)

[47] Cirstea, R.-G., Guo, C., Yang, B., Kieu, T., Dong, X., Pan, S.: Triformer: Triangular, variable-specific attentions for long sequence multivariate time series forecasting. In: IJCAI (2022)

[48] Wu, H., Xu, J., Wang, J., Long, M.: Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. NeurIPS (2021)

[49] Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., Zhang, W.: Informer: Beyond efficient transformer for long sequence time-series forecasting. In: AAAI (2021)

[50] Lim, B., Arık, S.Ö., Loeff, N., Pfister, T.: Temporal fusion transformers for interpretable multi-horizon time series forecasting. International Journal of Forecasting (2021)

[51] Wu, S., Xiao, X., Ding, Q., Zhao, P., Wei, Y., Huang, J.: Adversarial sparse transformer for time series forecasting. NeurIPS (2020)

[52] Xue, H., Salim, F.D.: Promptcast: A new prompt-based learning paradigm for time series forecasting. TKDE (2023)

[53] Zhou, T., Niu, P., Sun, L., Jin, R., et al.: One fits all: Power general time series

analysis by pretrained lm. NeurIPS (2023)

[54] Gruver, N., Finzi, M., Qiu, S., Wilson, A.G.: Large language models are zero-shot time series forecasters. NeurIPS (2024)

[55] Rasul, K., Ashok, A., Williams, A.R., Ghonia, H., Bhagwatkar, R., Khorasani, A., Bayazi, M.J.D., Adamopoulos, G., Riachi, R., Hassen, N., Biloš, M., Garg, S., Schneider, A., Chapados, N., Drouin, A., Zantedeschi, V., Nevmyvaka, Y., Rish, I.: Lag-Llama: Towards Foundation Models for Probabilistic Time Series Forecasting (2024)

[56] Islam, S., Elmekki, H., Elsebai, A., Bentahar, J., Drawel, N., Rjoub, G., Pedrycz, W.: A comprehensive survey on applications of transformers for deep learning tasks. Expert Systems with Applications (2023)

[57] Jin, M., Wen, Q., Liang, Y., Zhang, C., Xue, S., Wang, X., Zhang, J., Wang, Y., Chen, H., Li, X., Pan, S., Tseng, V.S., Zheng, Y., Chen, L., Xiong, H.: Large Models for Time Series and Spatio-Temporal Data: A Survey and Outlook. arXiv preprint arXiv:2310.10196 (2023)

[58] Joseph, V.R.: Optimal ratio for data splitting. Statistical Analysis and Data Mining: The ASA Data Science Journal (2022)

[59] Muraina, I.: Ideal dataset splitting ratios in machine learning algorithms: general concerns for data scientists and data analysts. In: 7th International Mardin Artuklu Scientific Research Conference (2022)

[60] Auffarth, B.: Machine Learning for Time-Series with Python: Forecast, Predict, and Detect Anomalies with State-of-the-art Machine Learning Methods. Packt Publishing Ltd, Birmingham, UK (2021)

[61] Lima, F.T., Souza, V.M.: A large comparison of normalization methods on time series. Big Data Research (2023)

[62] Fulcher, B.D.: Feature-based time-series analysis. In: Feature Engineering for Machine Learning and Data Analytics. CRC press, Florida, USA (2018)

[63] Lazzeri, F.: Machine Learning for Time Series Forecasting with Python. John Wiley & Sons, NJ,USA (2020)

[64] Herzen, J., Lässig, F., Piazzetta, S.G., Neuer, T., Tafti, L., Raille, G., Van Pottelbergh, T., Pasieka, M., Skrodzki, A., Huguenin, N., et al.: Darts: User-friendly modern machine learning for time series. JMLR (2022)

[65] Elsayed, S., Thyssens, D., Rashed, A., Jomaa, H.S., Schmidt-Thieme, L.: Do we really need deep learning models for time series forecasting? arXiv preprint arXiv:2101.02118 (2021)

[66] Everitt, B.S., Skrondal, A.: The Cambridge Dictionary of Statistics. Cambridge University Press, Cambridge, UK (2010)

[67] Davies, G.: Evaluating the effectiveness of predicting covariates in lstm networks for time series forecasting. arXiv preprint arXiv:2404.18553 (2024)

[68] Rasul, K., Ashok, A., Williams, A.R., Khorasani, A., Adamopoulos, G., Bhagwatkar, R., Biloš, M., Ghonia, H., Hassen, N.V., Schneider, A., et al.: Lag-llama: Towards foundation models for time series forecasting. arXiv preprint arXiv:2310.08278 (2023)

[69] Hartmann, K., Krois, J., Rudolph, A.: Statistics and Geodata Analysis Using R (SOGA-R); Department of Earth Sciences, Freie Universitaet Berlin (2023)

[70] Turner, S.M., Eisele, W.L., Benz, R.J., Holdener, D.J.: Travel time data collection handbook. Technical report, Federal Highway Administration (1998)

[71] Xu, J., Wu, C., Li, Y.-F., Bouvry, P.: Transformer multivariate forecasting: Less is more? AI4TS@AAAI'24 (2023)

[72] Shorten, C., Khoshgoftaar, T.M.: A survey on image data augmentation for deep learning. Journal of big data (2019)

[73] Wen, Q., Sun, L., Yang, F., Song, X., Gao, J., Wang, X., Xu, H.: Time series data augmentation for deep learning: A survey. In: IJCAI (2021)

[74] Nie, Y., H. Nguyen, N., Sinthong, P., Kalagnanam, J.: A time series is worth 64 words: Long-term forecasting with transformers. In: ICLR (2023)

[75] Hyndman, R.J., Fan, Y.: Sample quantiles in statistical packages. The American Statistician (1996)

[76] Rangapuram, S.S., Seeger, M.W., Gasthaus, J., Stella, L., Wang, Y., Januschowski, T.: Deep state space models for time series forecasting. NeurIPS (2018)

[77] Salinas, D., Flunkert, V., Gasthaus, J., Januschowski, T.: Deepar: Probabilistic forecasting with autoregressive recurrent networks. International Journal of Forecasting (2020)

[78] Lehmann, E.L., Casella, G.: Theory of Point Estimation. Springer, Berlin, Germany (2006)

[79] Grimit, E.P., Gneiting, T., Berrocal, V.J., Johnson, N.A.: The continuous ranked probability score for circular variables and its application to mesoscale forecast ensemble verification. Quarterly Journal of the Royal Meteorological Society (2006)

[80] Gneiting, T., Raftery, A.E.: Strictly proper scoring rules, prediction, and estimation. Journal of the American statistical Association (2007)

[81] Bröcker, J.: Evaluating raw ensembles with the continuous ranked probability score. Quarterly Journal of the Royal Meteorological Society (2012)

[82] Alexandrov, A., Benidis, K., Bohlke-Schneider, M., Flunkert, V., Gasthaus, J., Januschowski, T., Maddix, D.C., Rangapuram, S., Salinas, D., Schulz, J., Stella, L., Türkmen, A.C., Wang, Y.: GluonTS: Probabilistic and Neural Time Series Modeling in Python. JMLR (2020)

[83] Székely, G.J.: E-statistics: The energy of statistical samples. Bowling Green State University, Department of Mathematics and Statistics Technical Report (2003)

[84] WU, C., XU, J., BOUVRY, P., Li, J.: Strategic predictions and explanations by machine learning. In: ICOIN (2024)

[85] WU, C., Li, Y.-F., LI, J., XU, J., BOUVRY, P.: Trustworthy ai: Deciding what to decide. In: Computing Conference (2024)

[86] Waqas, A., Tripathi, A., Ramachandran, R.P., Stewart, P., Rasool, G.: Multimodal data integration for oncology in the era of deep neural networks: a review. arXiv preprint arXiv:2303.06471 (2023)

[87] Zhang, X., Chowdhury, R.R., Gupta, R.K., Shang, J.: Large language models for time series: A survey. arXiv preprint arXiv:2402.01801 (2024)

[88] Loucks, D.P., Van Beek, E.: Water Resource Systems Planning and Management: An Introduction to Methods, Models, and Applications. Springer, Cham, Switzerland (2017)

[89] Hu, M., Zhang, Z., Zhao, S., Huang, M., Wu, B.: Uncertainty in natural language processing: Sources, quantification, and applications. arXiv preprint arXiv:2306.04459 (2023)

[90] Sergazinov, R., Armandpour, M., Gaynanova, I.: Gluformer: Transformer-based personalized glucose forecasting with uncertainty quantification. In: ICASSP (2023)

[91] Yu, Y., Ma, R., Ma, Z.: Robformer: A robust decomposition transformer for long-term time series forecasting. Pattern Recognition (2024)

[92] Budennyy, S., Lazarev, V., Zakharenko, N., Korovin, A., Plosskaya, O., Dimitrov, D., Akhripkin, V., Pavlov, I., Oseledets, I., Barsola, I., *et al.*: Eco2ai: carbon emissions tracking of machine learning models as the first step towards sustainable ai. In: Doklady Mathematics (2023)

Appendix A Experiments on CO_2 emission

The experiments on CO_2 emissions were conducted on the High Performance Computing (HPC) using commonly utilized datasets as listed in Tab. A1 (Also see Tab. 1). On HPC, the GPU setting is {Tesla V100-SXM2-16GB and Tesla V100-SXM2-32GB}. The CPU setting is {Intel(R) Xeon(R) Gold 6132 CPU @ 2.60GHz/2 device(s), TDP:140.0}. The transformer-based forecasting models and their settings follow the Time-Series-Library, with the measurement solution provided by ECO2AI [92]. This experiment was conducted in a long-term forecasting setting with different prediction lengths.

In Table A2, the settings {96, 192, 336, 720} correspond to the prediction lengths. The results indicate that longer training times lead to higher CO2 emissions(See Fig. A1). Notably, the Crossformer model exhibited the highest CO2 emission at 566.09 grams, equivalent to burning 250 milliliters of gasoline [2]. In addition, The CO_2 emission is influenced by training dataset, especially by the number of timestemps and the number of variables. In Fig. A2, Transformer and Autoformer are influenced by the number of timestamps (The more timestemps, the more CO_2 emission. See Weather and ECL dataset on Transformer and Autoformer model). Crossformer is influenced by the number of variables (The more variables, the more CO_2 emission. See Traffic and ECL dataset on Crossformer model).

Table A1 Summary of information about four datasets used in measuring CO_2 emission).

Datasets	ETTh1	Weather	Electricity (ECL)	Traffic
Variables	7	21	321	862
Timestamps	17420	52696	26304	17544

Table A2 CO_2 Emissions of Training Process in Transformer-based Time Series Forecasting (OOM: Out of Memory)

Models		Autoformer			Crossformer			Transformer		
Metric		duration (s)	power consumption(Wh)	CO2 emissions(g)	duration (s)	power consumption(Wh)	CO2 emissions(g)	duration (s)	power consumption(Wh)	CO2 emissions(g)
ECL	96	841.8177	36.0394	7.0897	4157.4333	299.0629	58.8320	555.0109	26.8174	5.2756
	192	2099.2409	88.1902	17.3489	4859.8069	369.5088	72.6901	857.2692	42.9433	8.4478
	336	1602.8052	69.9166	13.7541	10024.7159	745.7526	146.7052	1088.4216	55.6925	10.9559
	720	2837.3136	131.2501	25.8196	OOM	OOM	OOM	1452.2351	81.4772	16.0283
ETTh1	96	322.3691	12.7535	2.5089	217.5166	8.4103	1.6545	117.5711	5.5337	1.0886
	192	468.7925	18.9128	3.7205	247.1342	10.7373	2.1122	134.2723	6.9900	1.3751
	336	663.4532	29.1745	5.7392	492.4600	24.4043	4.8008	146.9931	7.5727	1.4897
	720	587.0618	24.6777	4.8546	560.1151	34.2809	6.7438	251.7572	14.3321	2.8194
Traffic	96	397.8709	16.9082	3.3262	21642.9455	1746.5806	343.5891	216.2519	9.7177	1.9117
	192	466.2856	22.2510	4.3772	30353.6828	2418.3465	475.7395	283.9747	13.5821	2.6719
	336	600.3932	28.0720	5.5224	36043.9070	2877.6318	566.0906	337.8724	16.3602	3.2184
	720	844.6213	43.6340	8.5837	OOM	OOM	OOM	467.6036	23.9352	4.7086
Weather	96	607.6077	25.2550	4.9682	1884.1530	46.8592	9.2182	313.3538	16.0077	3.1490
	192	2111.8946	91.4837	17.9968	186.7157	5.0592	0.9953	858.2094	47.2053	9.2863
	336	2560.8089	115.5892	22.7388	1544.5321	42.6614	8.3924	548.1598	29.8991	5.8818
	720	2492.1999	117.2282	23.0613	232.2534	7.3457	1.4451	861.5856	54.7474	10.7700

[2] see calculation at link

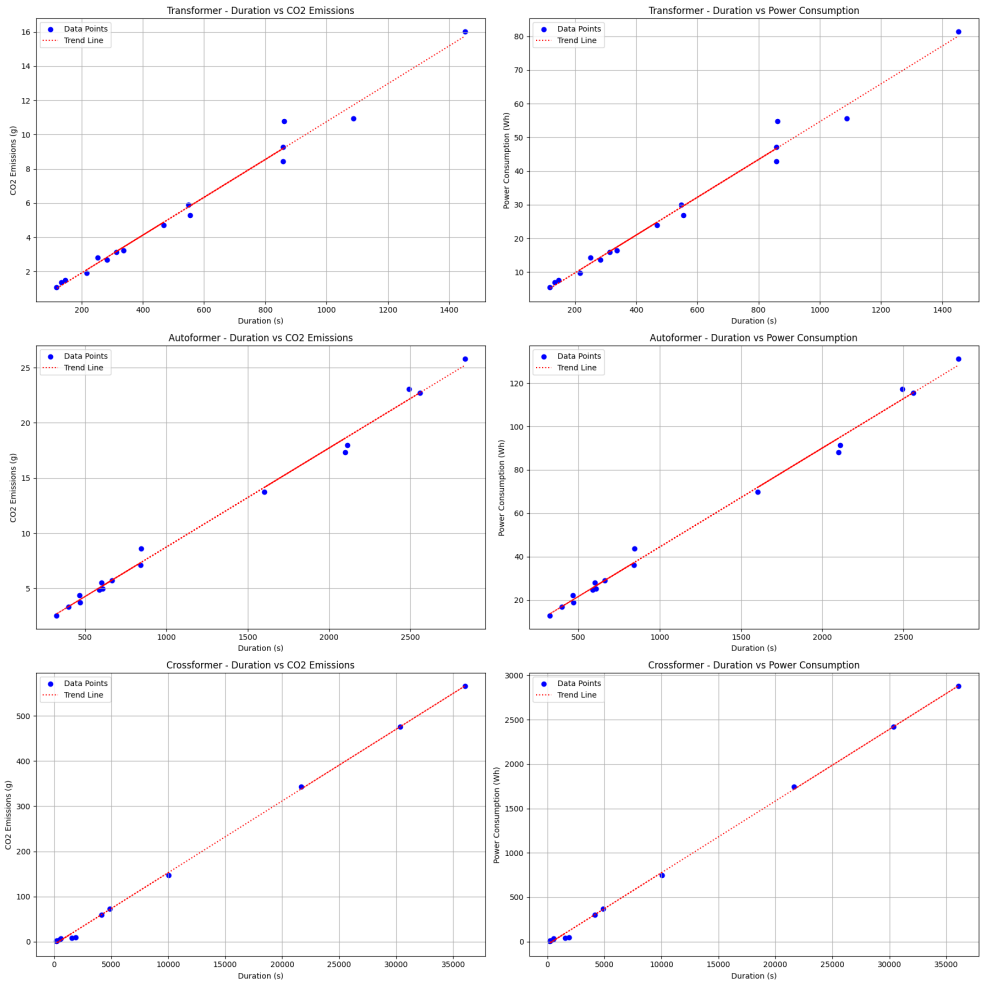

Fig. A1 Scatter Trend Charts of CO_2 Emission and Runtime (duration)/Power Consumption on Transformer-based Long-term Forecasting Models.

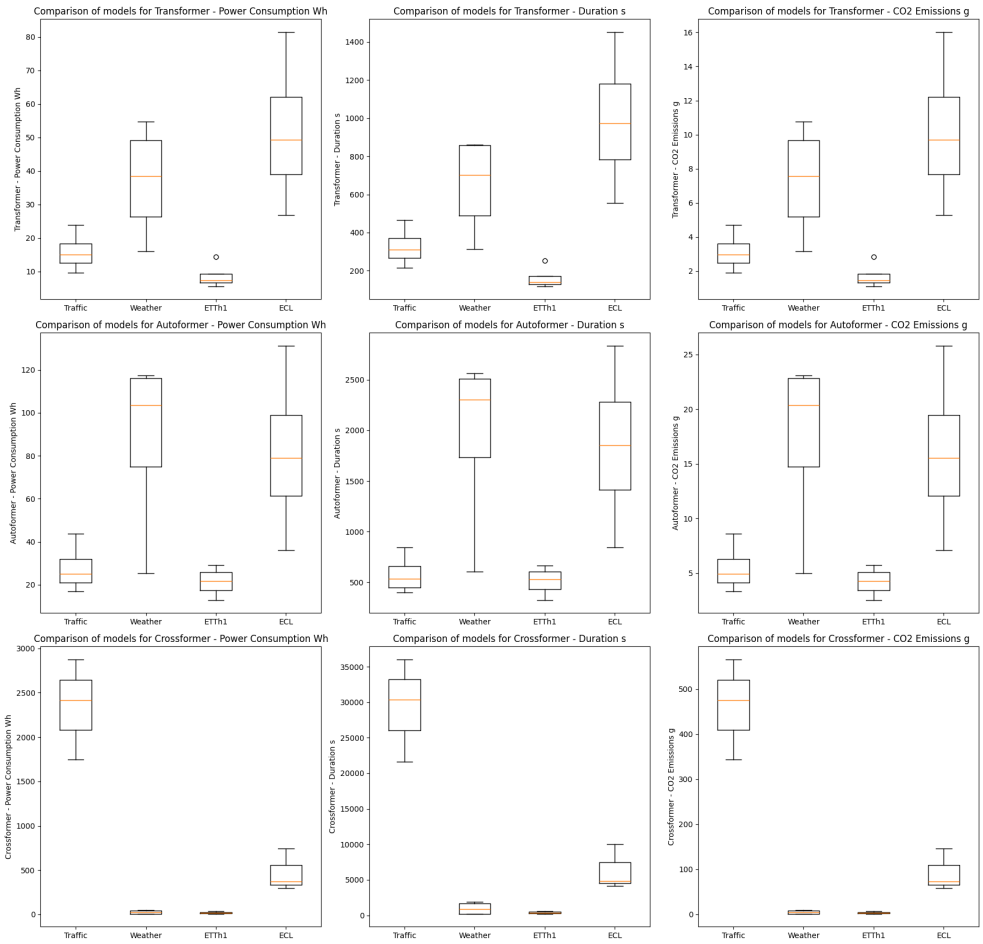

Fig. A2 Box Plot of CO_2 Emission of Different Models on Different Datasets.