electronics

Article

Transformer-Based Time-Series Forecasting for Telemetry Data in an Environmental Control and Life Support System of Spacecraft

Bin Song [1,2,†], Boyu Guo [1,†], Wei Hu [3,*], Zhen Zhang [3,*], Nan Zhang [3], Junpeng Bao [4], Jianji Wang [1] and Jingmin Xin [1]

[1] Institute of Artificial Intelligence and Robotics, College of Artificial Intelligence, Xi'an Jiaotong University, Xi'an 710049, China; wangjianji@mail.xjtu.edu.cn (J.W.)
[2] School of Software Engineering, Xi'an Jiaotong University, Xi'an 710049, China
[3] China Astronaut Research and Training Center, Beijing 100094, China; szznd@126.com
[4] School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an 710049, China; baojp@mail.xjtu.edu.cn
* Correspondence: huweiacc@aliyun.com (W.H.); zhangzhen05@tsinghua.org.cn (Z.Z.)
† These authors contributed equally to this work.

Abstract: Safety and stability are critical in manned space missions, requiring an environmental control and life support system (ECLSS) of spacecraft to operate reliably. This study analyzed the time-series characteristics of telemetry data, including total pressure, temperature, and humidity, to predict the ECLSS's operational state. Existing algorithms for time-series forecasting, including ARIMA, LSTM, TCN, and NBEATS, often struggle with long-sequence forecasting and discrepancies in data distribution, which hinder their ability to deliver accurate predictions. To address these challenges, this study introduces a two-stage normalization method, mean instance normalization (MeanIN), designed to adjust input data distributions and restore output data distributions, thereby significantly enhancing predictive performance. Experimental evaluations on ECLSS telemetry data demonstrate that MeanIN consistently improves model accuracy, with the informer model achieving superior results in long-sequence forecasting tasks. These results underscore the efficacy of MeanIN and its potential to support critical applications in anomaly detection and predictive analysis for spacecraft telemetry data.

Keywords: time-series forecasting; transformer; informer; ECLSS telemetry data

check for updates

Academic Editor: Mahmut Reyhanoglu

Received: 2 December 2024
Revised: 13 January 2025
Accepted: 21 January 2025
Published: 23 January 2025

Citation: Song, B.; Guo, B.; Hu, W.; Zhang, Z.; Zhang, N.; Bao, J.; Wang, J.; Xin, J. Transformer-Based Time-Series Forecasting for Telemetry Data in an Environmental Control and Life Support System of Spacecraft. *Electronics* **2025**, *14*, 459. https://doi.org/10.3390/electronics14030459

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Aerospace technology is the crystallization of modern science and technology involving various disciplines such as electronic technology, computer engineering, and other high-tech areas. Manned space engineering, as a crucial part of aerospace technology, not only pushes the boundaries of innovation but also yields widespread applications across industries. For instance, the heat transfer system of liquid rocket engines has been adapted to address domestic waste issues. Space breeding has been used in food processing and biopharmaceuticals [1]. Space protein crystallization research can be employed to develop new drugs against osteoporosis and muscle atrophy [2]. The application of new materials and processes in the aerospace field to other industries has brought huge economic benefits [3].

Compared with other traditional industries, the aerospace field, especially manned spaceflight, requires higher precision and more refined technology, which are directly

related to the life and health of astronauts. A spacecraft is disturbed not only by various uncontrollable factors (e.g., rays and particles) during its operation but also by its internal problems (e.g., aging equipment and overload), leading to changes in the spacecraft's operating status [4].

The operation status of the spacecraft environmental control and life support system (ECLSS) must be accurately predicted to safeguard astronauts and ensure stable aerospace equipment performance. Predicting the operational status of the ECLSS is critical for anticipating system failures and ensuring the safety and well-being of astronauts. This predictive task involves analyzing temporal patterns in sensor data to forecast future system states. The process typically includes data preprocessing, model development using time-series analysis or machine learning, and generating forecasts to anticipate potential anomalies. Challenges in this process include managing noisy data, addressing missing values, and accommodating dynamic environmental conditions within the spacecraft. Consequently, prediction models must be robust and adaptive, offering reliable forecasts that facilitate timely interventions and maintenance actions. In long-duration space missions, where real-time intervention may not be feasible, accurate predictions are paramount for ensuring the continuous operation of the ECLSS and the safety of the crew. Therefore, the core work of anomaly detection for spacecraft relies on the accuracy of predictions [5].

Spacecraft telemetry data are periodically collected by sensors within the spacecraft at fixed intervals, forming a time series that primarily consists of five parameters: total pressure, partial oxygen pressure, temperature, humidity, and partial pressure of carbon dioxide. Under normal conditions, telemetry data from sensors exhibit stable variations within expected ranges. However, in certain instances, data may fluctuate due to external or internal factors, occasionally exceeding normal thresholds. Additionally, operational requirements or technical issues can cause sensors to be deactivated or malfunction for specific durations. Consequently, data may become discontinuous or entirely missing, complicating the monitoring of the ECLSS. To address these challenges during data preprocessing, the frequency of original data is reduced to meet analysis requirements, thereby minimizing the impact of missing data. Subsequently, linear interpolation is employed to estimate and fill in the missing values, ensuring continuity in the dataset.

In this study, we propose a novel two-stage normalization method, mean instance normalization (MeanIN), specifically designed for transformer-based models to address the challenges posed by non-stationary and high-frequency fluctuations in aerospace telemetry data, which is a direction related to the research direction: information models [6]. MeanIN consists of two key steps: normalization and denormalization. During normalization, the input data distribution is adjusted by leveraging the mean and range of the data to transform them into a consistent and stationary form. This step ensures that temporal patterns are more effectively captured by the model. In the denormalization step, the model's output is restored to its original distribution, preserving the interpretability of the predictions.

The remaining sections of this paper are organized as follows. Section 2 reviews the related works on time-series forecasting and the application of deep learning models in this field. Section 3 outlines the methodology, detailing data preprocessing steps such as downsampling, handling missing data, feature scaling, and the introduction of the proposed mean instance normalization (MeanIN). Section 4 presents the experimental setup and result analysis, comparing various forecasting models and demonstrating the effectiveness of the proposed methods. Section 5 presents the conclusions of the study, summarizing its core contributions and findings, as well as outlining future work directions to address current limitations. Finally, Section 6 discusses broader future research directions, emphasizing practical deployment and potential enhancements to the proposed framework.

2. Related Works

Spacecraft telemetry data are characterized by long-term trend fluctuations, seasonality, and randomness. These features make time series models particularly suitable for predicting future values by extracting potential characteristics and trends from historical data. Traditional statistical models, such as autoregressive (AR), autoregressive moving average (ARMA), and autoregressive integrated moving average (ARIMA) [7], have been widely applied in this domain due to their simplicity and interpretability. By treating time-series prediction as a regression problem, parameter correlations have been effectively utilized to improve the performance of these models [8].

Deep learning models, with their flexibility and ability to learn complex temporal patterns, have emerged as superior alternatives to traditional statistical models [9–12]. Recurrent neural networks (RNNs) and their variants, such as long short-term memory (LSTM) and gated recurrent units (GRU), have been extensively explored for time-series forecasting tasks. For example, comparisons of ARIMA, multi-layer perceptron (MLP), RNN, LSTM, delayed LSTM, and GRU on Egyptsat-1 satellite telemetry data revealed that LSTM and ARIMA achieved the best performance [13]. Notably, the gate mechanisms in LSTM mitigate the gradient vanishing problem inherent in RNNs, enabling the effective modeling of long-term dependencies. However, the recursive structure of LSTM introduces sequential constraints, hindering parallelization and potentially causing information loss when processing long sequences. To address these challenges, attention mechanisms and hybrid architectures have been proposed to enhance deep learning models. Yang [14] applied an attention-enhanced, two-layer LSTM for spacecraft telemetry data prediction, combining nonlinear modeling with multi-scale anomaly detection strategies to improve detection accuracy. Convolutional neural networks (CNNs) and hybrid models that integrate CNN and LSTM have also been explored to address the limitations of sequential processing. For instance, Wang [15] introduced a temporal convolutional network (TCN) to extract telemetry data features for prediction and anomaly detection using a static threshold method. These approaches demonstrate the potential of combining advanced deep learning architectures with domain-specific strategies to improve forecasting and anomaly detection performance. Li and Gou [16] employed improved pattern gradient spectral entropy (IPGSE) and CNN to detect aero-engine sensor faults. Zhao [17] and Zhang [18] proposed fusion models that integrate CNN and LSTM, demonstrating enhanced fault detection accuracy. Despite these successes, such models often struggle to efficiently capture global patterns in time-series data.

The advent of deep learning has significantly advanced time series forecasting, with transformer [19], a global deep learning model originally developed for natural language processing, playing a pivotal role in addressing long sequence limitations. The core strength of the transformer lies in its attention mechanism [20,21], which selectively focuses on relevant features, ignoring less critical information. Unlike RNNs, transformers can simultaneously model long-term and short-term dependencies through a multi-head self-attention mechanism, enabling parallel processing and efficient handling of large-scale datasets. Researchers have successfully applied transformer to time series forecasting, achieving breakthroughs in capturing temporal patterns and overcoming traditional model limitations [22–24].

Informer [25], a transformer-based model, further enhances the capabilities of transformer, achieving remarkable advancements in time-series forecasting. As shown in Figure 1, the overall architecture of Informer involves encoding the embedded input using M encoders to extract sample features. These features are then decoded by N decoders, and the final result is produced after passing through a fully connected layer. Informer has been successfully applied to tasks such as forecasting the temperature of electricity transformers,

electricity consumption loads, and weather, demonstrating outstanding performance. In this study, we apply Informer to spaceflight telemetry data.

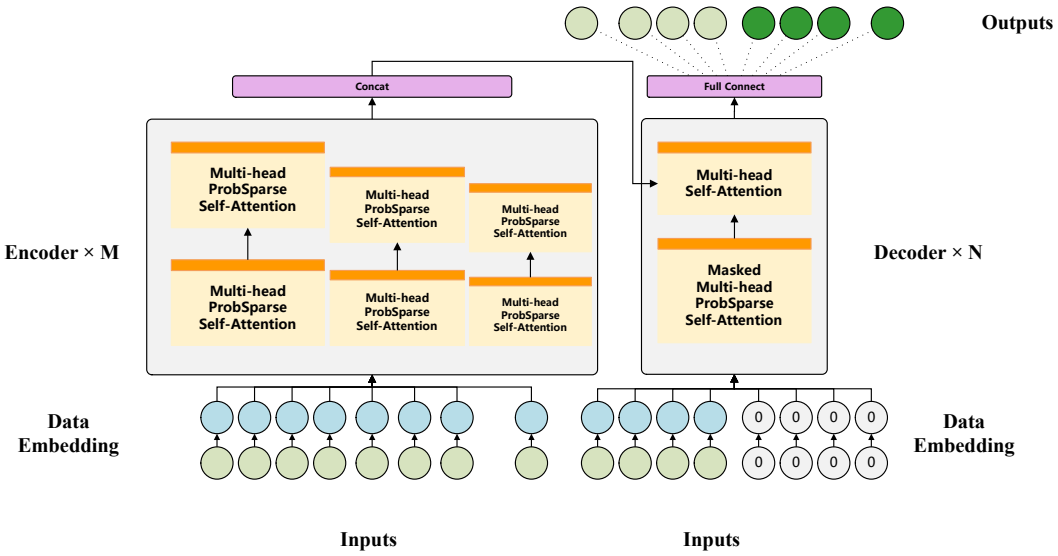


Figure 1. The overall architecture of the informer model.

Compared with the original transformer model, informer has the following characteristics:

1. ProbSparse self-attention is proposed as an alternative to traditional self-attention. It reaches $O(L \log L)$ in terms of time complexity and memory usage and shows good performance in extracting the global information of the sequence. For the random sampling of the query of a certain key, ProbSparse self-attention calculates the sparsity score and selects N queries with the highest sparsity scores to calculate the dot product pairs. Then, the final attention result is obtained. Its equation is as follows:

$$A(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d}})V, \quad (1)$$

   where Q is the selected query, K and V are linear transformations of the original inputs, and d is the input dimension.
2. Self-attention distilling highlights the main attention by halving the cascade input layer and effectively handles long input sequences. As the number of encoder layers increases, the output of each element in the sequence already contains information on other elements in the sequence. Therefore, the dominant feature can be prioritized through the distilling operation, thereby reducing the length of the input sequence that needs to be processed while retaining most of the information.
3. Generative inference adopts a known sequence before the time point to be predicted as a part of the input of the decoder and splices it with the sequence to be predicted. Then, it embeds the position vector and the time features of each timestamp in the spliced sequence. Afterward, the entire sequence is used as input to the decoder. This one-step forward calculation not only considerably improves the inference speed of long sequence prediction but also avoids the problem of error accumulation in the autoregressive structure.

3. Methodology

3.1. Data Downsampling

The dataset in this study is from the record of various parameters of a manned spacecraft during a mission. The parameters mainly include total pressure (kPa), temperature

(°C), and humidity (relative humidity, %). Notably, due to confidentiality requirements, only a part of the data is shown in this paper. Figure 2 presents a part of the original data of the spacecraft telemetry dataset that we use.

The sensors in the manned spacecraft sample data at a high frequency. Spacecraft sensors may shut down or malfunction in some periods due to mission requirements, external interference, sensor aging, and other factors. Thus, many missing and abnormal data points are present.

The original data are too densely sampled, resulting in no significant change in the data over a long period, which causes information redundancy. Therefore, the first step in data pre-processing is to change the sampling frequency of the data. We resample the frequency to 15 min, 30 min, 45 min, and 1 h by the frequency that matches people's working conditions. Specifically, all the values in the sampling interval are merged, and the average is determined.

As shown in Table 1, as the sampling frequency decreases, the proportion of missing values in the data is considerably reduced. No missing data are observed when the sampling frequency is 45 min.

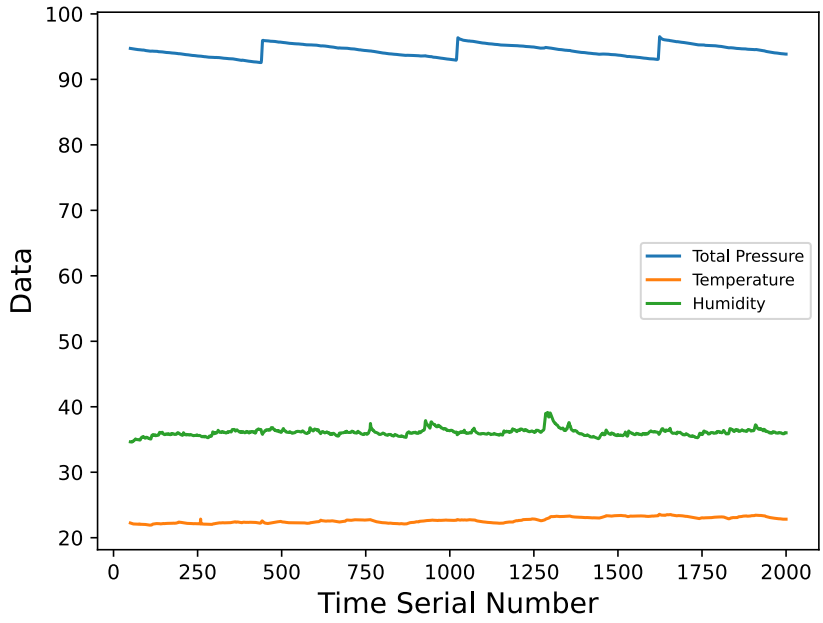


Figure 2. Part original data of original data of manned spacecraft. The units for total pressure, temperature, and humidity are kilopascals, degrees Celsius, and relative humidity, respectively.

Table 1. Missing data with different time-frequency.

Frequency	Number of Total Data	Number of Missing Data	Percentage (%)
15 min	2806	70	2.49465
30 min	1403	5	0.35638
45 min	936	0	0
1 h	702	0	0

3.2. Missing Data Treatment

The most commonly used methods for dealing with missing data points include imputation and deletion. Generally, we prefer to keep rather than delete data to avoid information loss. Interpolation is a commonly used prior method with expert experience to determine the value of missing data points. For example, linear interpolation requires a

certain linear fitting relationship between missing data and neighboring points. Considering the characteristics of spacecraft telemetry data, using interpolation to fill in the missing data is appropriate. Figure 3 shows the distribution of data at sampling frequencies of 0.5 s, 5 s, 1 min, 15 min, 30 min, 45 min, and 1 h. As the sampling frequency decreases, the distribution of the data becomes smoother.

In addition, informer can mitigate the impact of missing data and outliers via time-feature encoding and an attention mechanism. The timestamp corresponding to each data point in informer is encoded. The encoded temporal features are later accumulated with the original sequence so that the processed sequence contains the features of each timestamp, thus reducing the impact of data from missing moments. Compared with traditional statistical methods, the attention mechanism in transformer-based deep learning methods can obtain the contribution of each element in the sequence to the whole and allocate less attention to the outliers that deviate from the overall statistical indicators (mean, variance, etc.), that is, the impact of the outliers on the overall data is reduced.

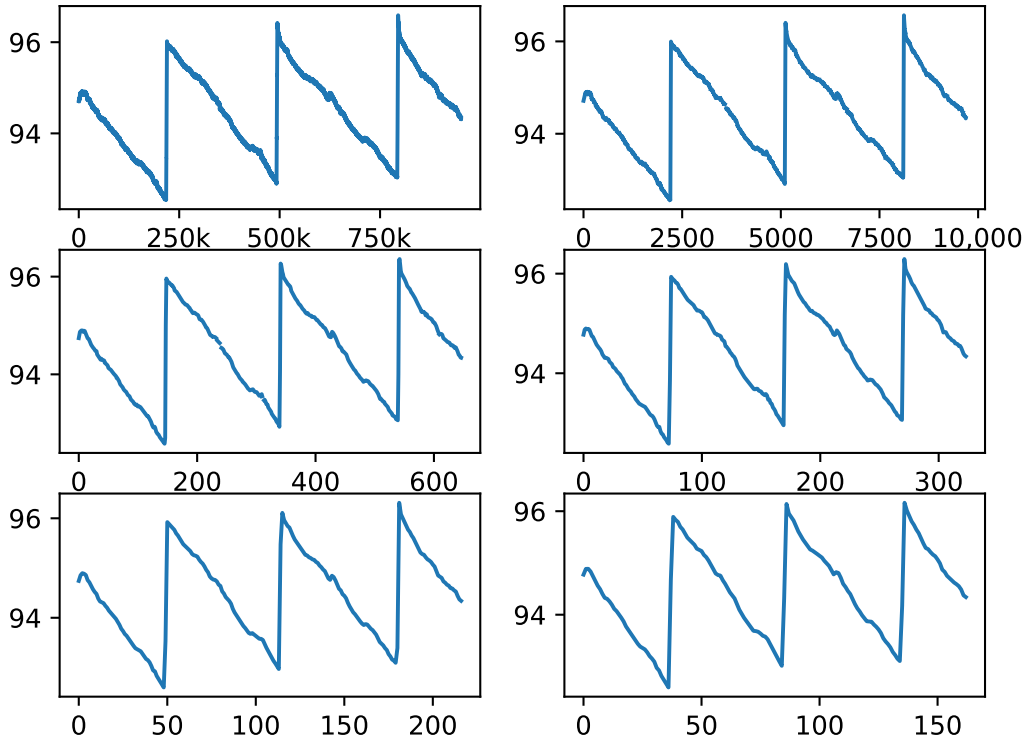


Figure 3. Comparison of total pressure at different resolutions. From left to right and top to bottom, the sampling frequency gradually decreases.

3.3. Feature Scaling

Data are often scaled and processed before training in deep learning to effectively utilize data and accelerate the convergence of the model. After scaling, the multi-dimensional features have similar scales to help algorithms converge rapidly. Standardization (also called Z-score normalization) is the most commonly used feature scaling method. Standardization is essentially a linear transformation. Linear transformations keep linear combinations and linear relational equations unchanged, thereby ensuring that the particular model does not fail. Standardization makes each feature have zero mean and unit variance by subtracting the mean and dividing by the standard deviation. The mean and standard deviation are computed by the whole input sequence as follows:

$$\mathbb{E}[x_l^n] = \frac{1}{L}\sum_{l=1}^{L} x_l^n, \tag{2}$$

$$\mathrm{Var}[x_l^n] = \frac{1}{L}\sum_{l=1}^{L}(x_l^n - \mathbb{E}[x_l^n])^2, \tag{3}$$

where x_l^n refers to the n-th variable of an element of length l, and $\mathbb{E}[x_l^n]$ and $\mathrm{Var}[x_l^n]$ are the mean and standard deviation of the input, respectively.

Using standardization to normalize the input of multivariate time series proceeds as follows:

$$\hat{x}_l^n = \frac{x_l^n - \mathbb{E}[x_l^n]}{\sqrt{\mathrm{Var}[x_l^n] + \epsilon}}, \tag{4}$$

where $\hat{x}_l^n$ represents the result of linear normalization, and ϵ is a constant parameter considering the case of the standard deviation of 0.

The value of spacecraft telemetry data always changes within a small range, but at a high frequency, i.e., the spacecraft telemetry data appear in a jagged data state. Such drastic fluctuations can be alleviated if the value range is limited to a much smaller range. For this purpose, we use min–max normalization to feature-scale the dataset. Min–max normalization makes the data scale fall between 0 and 1, and scaling is determined by the minimum and maximum values. Min–max normalization minimizes the impact on the data distribution while ensuring fast model convergence. Using min–max normalization to normalize the input of multivariate time series proceeds as follows:

$$\hat{x}_l^n = \frac{x_l^n - \min(\boldsymbol{x}^n)}{\max(\boldsymbol{x}^n) - \min(\boldsymbol{x}^n) + \epsilon}, \tag{5}$$

where $\hat{x}_l^n$ represents the result of min–max normalization, $\min(\boldsymbol{x}^n)$ and $\max(\boldsymbol{x}^n)$ represent the maximum and minimum values of the input by feature dimension, respectively, and ϵ is a small constant parameter to avoid the case of division by zero.

3.4. Mean Instance Normalization in Informer

Given a univariate time series $\boldsymbol{x} = \{x_1, \ldots, x_L | x_l \in \mathbb{R}^L\}$ of length L, we consider a multivariate time series to be composed of several univariate time series, where N denotes the number of variables. Thus, a multivariate time series can be formulized as $\mathcal{X} = \{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N | x^n \in \mathbb{R}^{N \times L}\}$.

While feature scaling deflates the whole dataset to a specific range, there are still large differences between the data samples. For example, standardization maps the overall data distribution to a normal distribution with a mean of 0 and a variance of 1, thereby making the data more consistent with statistical assumptions. However, aerospace telemetry data do not precisely conform to a normal distribution, as shown in Figure 4. We divide all the data into four equal segments by time and compare them with the overall data distribution. It can be seen that the data distribution exhibits significant differences between the different time segments. The data distributions of different parameters in aerospace telemetry data vary greatly, even for the same parameter in different periods. The discrepancy between each input series becomes even more apparent for the forecasting models. Such differences in data distributions make it difficult for the model to capture temporal patterns accurately. To address this issue, we propose a normalization method called mean instance normalization (MeanIN). Specifically, MeanIN consists of two distinct steps: normalization and denormalization. The former normalizes the input to fix its distribution based on the mean and maximum–minimum value, while the latter transforms the output to the original distribution. The whole MeanIN proceeds as follows:

$$\mathbb{E}[x_l^n] = \frac{1}{L}\sum_{l=1}^{L} x_l^n, \quad \hat{x}_l^n = \alpha\left(\frac{x_l^n - \mathbb{E}[x_l^n]}{\max(\boldsymbol{x}^n) - \min(\boldsymbol{x}^n) + \epsilon}\right) + \beta, \tag{6}$$

$$\hat{y}_l^n = \frac{\hat{x}_l^n - \beta}{\alpha}(\max(\boldsymbol{x}^n) - \min(\boldsymbol{x}^n) + \epsilon) + \mathbb{E}[x_l^n], \tag{7}$$

where $\hat{x}_l^n$ and $\hat{y}_l^n$ represent the result of normalization and denormalization, respectively, obtained using the MeanIN method that we proposed. $\mathbb{E}[x_l^n]$ represents the mean of the elements in the input, and $\min(\boldsymbol{x}^n)$ and $\max(\boldsymbol{x}^n)$ represent the maximum and minimum values of the input by feature dimension, respectively. α and β are hyperparameters that are trained with the model, and ϵ is a small non-zero constant used to avoid the case of division by zero.

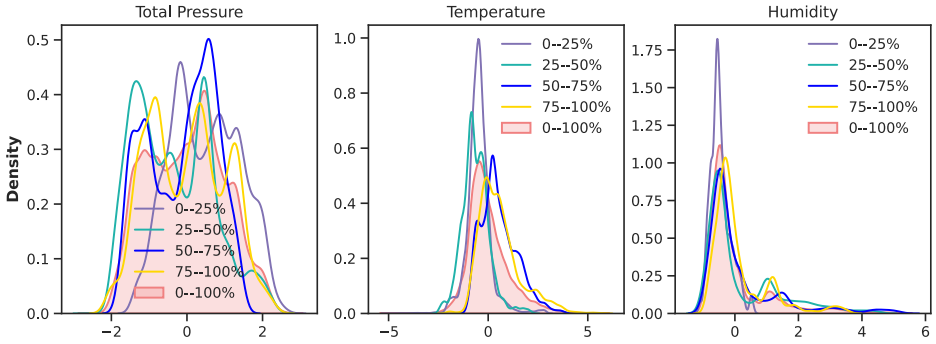


Figure 4. Distribution discrepancy of total pressure, temperature, and humidity data.

The input sequences of different distributions are transformed by MeanIN to intervals with a mean of 0 and a more consistent range of values, making the statistical characteristics of each sequence closer. In other words, MeanIN maps the input sequences of the model to a similar data distribution. The MeanIN-transformed sequences have consistent mean and scale, reducing the non-stationary information in the sequences. The stationary input sequences allow the model to focus more on the temporal patterns of the sequences rather than the short-term trend changes. At the same time, the same scale eliminates the effect of the magnitude of different parameters.

We analyze the distributions of the training and test data and the MeanIN-transformed data for each step of the model separately in Figure 5. It is obvious that there are large differences in the distributions of the training and test data for different periods due to the original non-stationary sequences. In particular, there is a significant distribution drift problem between the training and test data for the temperature and humidity data, i.e., the mean and variance of the data vary with time. After the MeanIN transformation, the distributions of the training and test data converge to the same distribution, and the distribution discrepancy is significantly alleviated.

It shows the overview of informer with MeanIN in Figure 6. Each sample sequence is subtracted from the mean of the sequence and later divided by the difference between the maximum and minimum values in the sequence, and finally mapped to the same vector space by a layer of an affine transformation. Informer only takes the transformed sequence as input and outputs its predicted values, which are obviously not the original data distribution. Therefore, the second step of MeanIN is to map the output sequence back to the original distribution by the inverse operation of Equation (6) to obtain the final output.

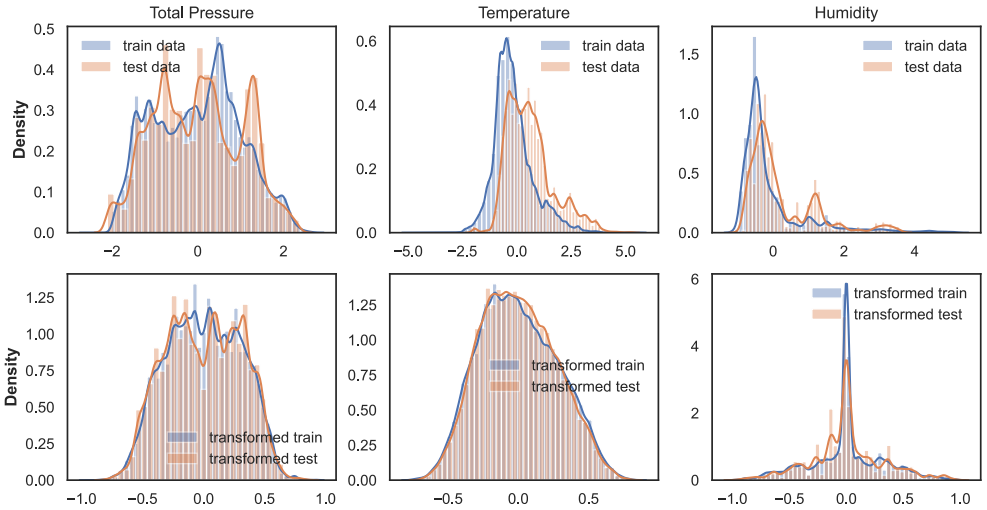


Figure 5. Effect of mean instance normalization on distribution of total pressure, temperature, and humidity data. The first row shows the data distribution of the original training and test data, and the second row shows the distribution transformed by MeanIN.

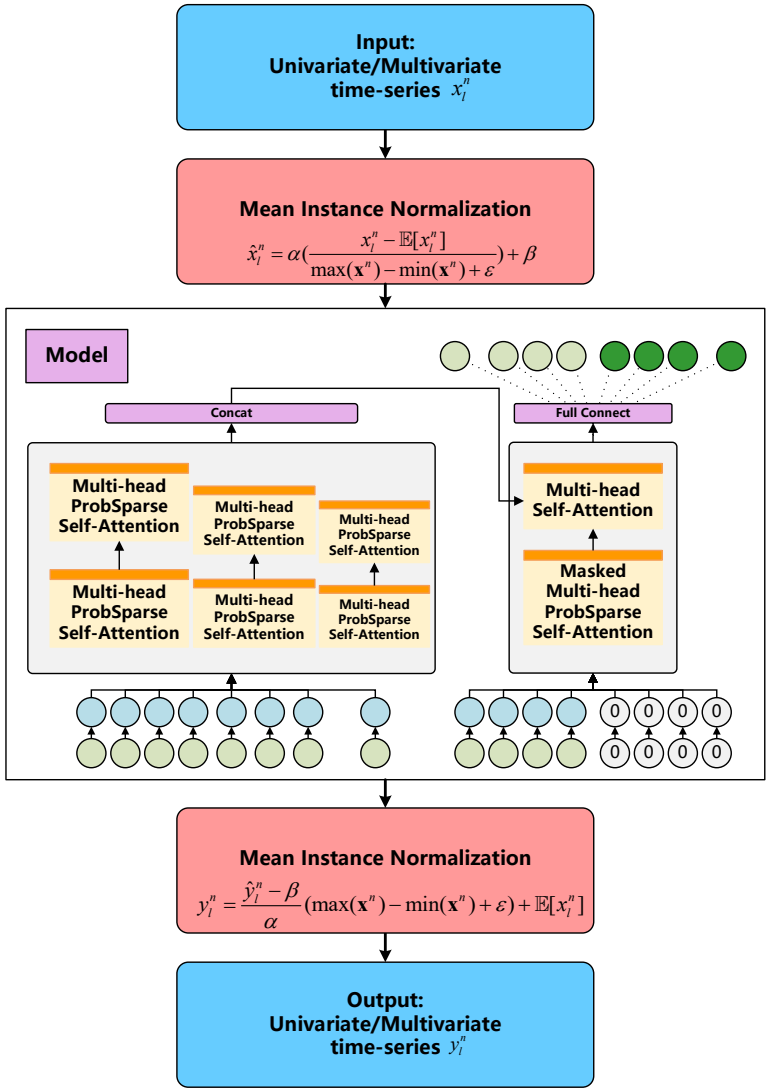


Figure 6. Overview of informer with MeanIN.

4. Experiment

We extensively perform experiments on the telemetry data in environmental control and life support systems of spacecraft, including two datasets with different sampling frequencies. The raw data contain various parameters recorded by sensors on the aircraft. We selected three significant parameters that are closely related to the health of the astronauts as indicators, including total pressure, temperature, and humidity. To explore the effect of temporal granularity on forecasting, we created separate datasets: DBSm1 sampled every minute and DBSm15 sampled every 15 min.

4.1. Experiment Environment

The experimental environment includes a CPU with 2*Intel(R) Xeon(R) Gold 622R CPU @ 2.90 GHz, 256 GB DDR4 2933 MHz RAM, Intel S4510 960G 2.5" SATA solid-state hard drive, and RTX 3090 (24 GB video memory). All experiments are performed on RTX 3090.

4.2. Result Analysis

4.2.1. Comparison of Various Time-Series Forecasting Models on Real Spacecraft Data

The univariate forecasting results on DBSm15 and DBSm1 are provided in Table 2 and Table 3, respectively. We set different and gradually increasing forecasting horizons for DBSm15 and DBSm1 to test the effectiveness of the model for short-term prediction and long-term prediction at different time granularity. We compare the performance of transformer, informer, NBEATS [26] and RNN-based models (RNN, LSTM, and GRU), CNN-based model TCN, and the traditional statistical model ARIMA on these two datasets. The length of the input sequence for each model is set to 96, except for ARIMA. Since ARIMA is a single-step model, we use the true values before the predicted point as input to implement multi-step prediction. In addition, we replaced the autoregressive output structure of the RNN-based model by using an RNN encoder to encode a fixed-length input block and a fully connected network to produce a fixed-length output. This approach avoids the problem of error accumulation caused by the autoregressive output structure; thus, the RNN-based model can obtain better prediction performance. Under this setting, each method accomplishes predictions of a single parameter of the dataset. Then, the average of the MSE of the predictions for all parameters is used as an indicator to judge the model performance.

From the results of Table 2, the fully connected NBEATS-based model and the RNN-based model perform slightly better than the other models for short-term prediction. However, as the prediction horizon increases ($L > 48$), the transformer-based model outperforms the other models substantially. This is more evident in Table 3. From Table 3, we can observe that informer has the best performance. Informer outperforms transformer, NBEATS, and RNN on MSE by decreasing 51.3%, 54.1%, and 46.6% on average. ARIMA and TCN perform better in short-term forecasting, but they are weak in long-term forecasting. The computational efficiency and cumulative error of ARIMA become intolerable for long-term forecasting. This is the reason why we did not compare ARIMA in the long-term prediction of DBSm1. TCN performs better with a prediction horizon below 60, and its average performance is only slightly worse than the transformer-based model. However, limited by the structure of CNN, TCN has strict requirements on the input sequence, which affects its performance on long-term prediction.

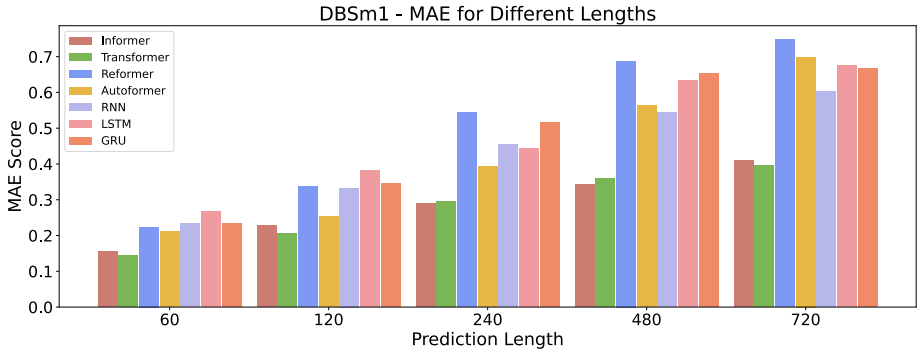
Table 2. Univariate forecasting results on DBSm15 under different prediction lengths $L \in \{12, 24, 36, 48, 60\}$. The input sequence length is set to 96.

Methods	Metric	Prediction Length				
		12	**24**	**36**	**48**	**60**
Informer	MSE	0.497	0.656	0.696	**0.703**	**0.766**
	MAE	0.446	0.529	0.546	**0.591**	**0.624**
Transformer	MSE	0.472	0.647	0.748	0.875	0.926
	MAE	0.364	0.484	0.553	0.605	0.629
NBEATS	MSE	**0.407**	0.620	0.662	0.781	0.935
	MAE	**0.334**	0.453	0.509	0.552	0.611
RNN	MSE	0.453	0.655	0.703	0.883	0.768
	MAE	0.365	0.520	0.573	0.647	0.638
LSTM	MSE	0.476	0.593	0.755	0.761	0.849
	MAE	0.376	0.478	0.595	0.610	0.663
GRU	MSE	0.427	**0.559**	**0.561**	0.757	0.783
	MAE	0.363	**0.464**	**0.495**	0.616	0.610
ARIMA	MSE	0.535	0.896	1.136	1.261	1.289
	MAE	0.439	0.669	0.801	0.863	0.888
TCN	MSE	0.450	0.663	0.761	0.783	0.786
	MAE	0.367	0.503	0.576	0.599	0.613

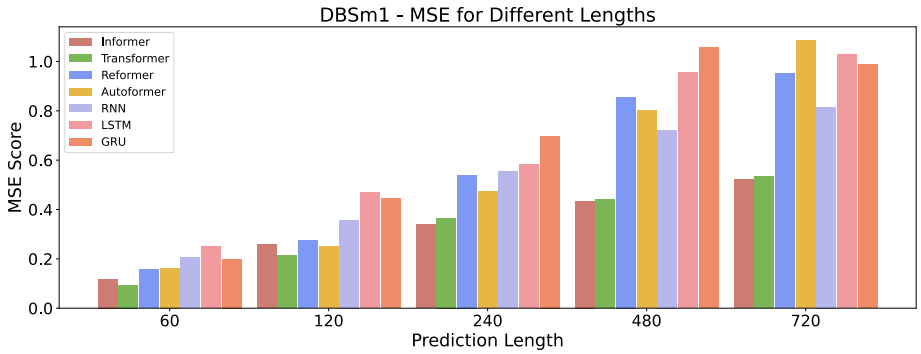
Table 3. Univariate forecasting results on DBSm1 under different prediction lengths $L \in \{60, 120, 240, 480, 720\}$. The input sequence length is set to 96.

Methods	Metric	Prediction Length				
		60	**120**	**240**	**480**	**720**
Informer	MSE	**0.097**	**0.189**	**0.277**	**0.409**	**0.459**
	MAE	**0.129**	**0.214**	**0.266**	**0.350**	**0.382**
Transformer	MSE	0.194	0.393	0.631	0.823	0.879
	MAE	0.203	0.327	0.474	0.589	0.633
NBEATS	MSE	0.193	0.386	0.663	0.894	1.076
	MAE	0.238	0.321	0.482	0.618	0.690
RNN	MSE	0.178	0.354	0.551	0.734	0.866
	MAE	0.191	0.301	0.458	0.599	0.641
LSTM	MSE	0.181	0.369	0.560	0.777	0.914
	MAE	0.190	0.303	0.432	0.590	0.657
GRU	MSE	0.173	0.338	0.511	0.767	0.880
	MAE	0.180	0.288	0.432	0.574	0.640

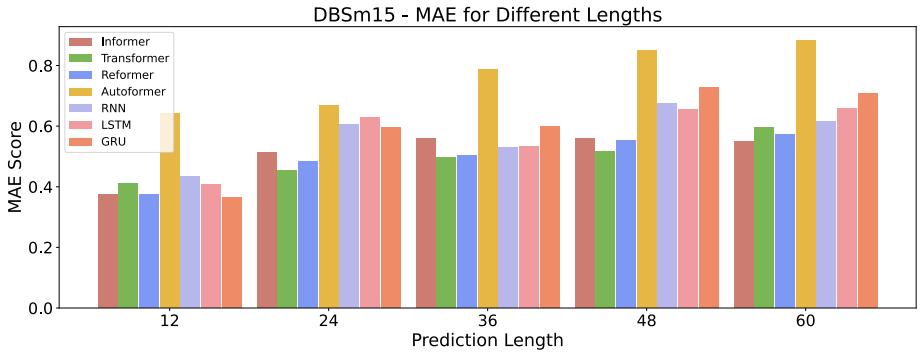
Transformer-based models have demonstrated strong predictive ability in univariate time-series forecasting. On multivariate time-series forecasting, we compare the performance of several state-of-the-art transformer-based models (transformer, informer, reformer [27], and autoformer [28]) and RNN-based models on short-term forecasting and long-term forecasting. From Figure 7, we can observe that transformer-based models still maintain the same excellent performance on multivariate time-series forecasting tasks as that on univariate time-series forecasting tasks. On the forecasting horizon interval $L \in \{60, 120\}$, the vanilla transformer achieves lower prediction errors than several improved transformer models. On the longer forecasting horizon ($L > 120$), informer shows the overwhelming performance in the long-term time-series forecasting.



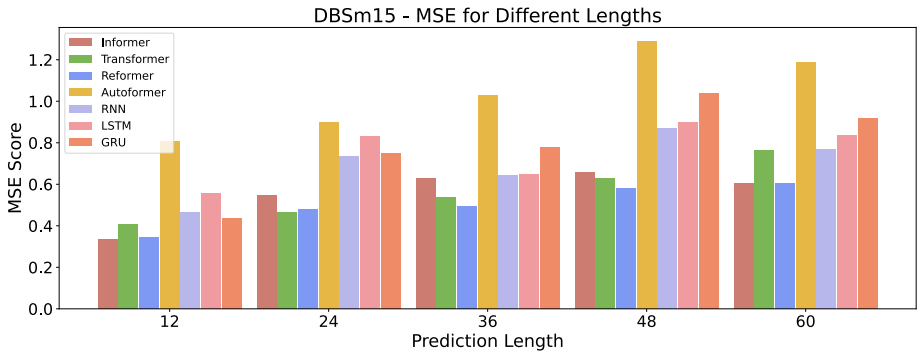
(a)



(b)



(c)



(d)

Figure 7. Performance comparison of different models (informer, transformer, reformer, autoformer, RNN, LSTM, and GRU) on time series prediction tasks. (**a**) MAE scores for different prediction lengths on the DBSm1 dataset. (**b**) MSE scores for different prediction lengths on the DBSm1 dataset. (**c**) MAE scores for different prediction lengths on the DBSm15 dataset. (**d**) MSE scores for different prediction lengths on the DBSm15 dataset.

4.2.2. Effectiveness of Mean Instance Normalization for Transformer-Based Models

The forecasting errors of the four baselines and MeanIN are compared in Table 4. The results show that MeanIN is consistently much better than all baselines, the vanilla transformer, informer, reformer, autoformer, FEDformer [29], and Pyraformer [30]. The effect of MeanIN remains stable when the length of the input sequence is increased from 24 to 720, indicating that MeanIN makes models more robust to different forecasting lengths. For example, transformer with MeanIN outperforms the vanilla transformer on MSE by decreasing 22.5% ($L = 24$), 10.9% ($L = 48$), 22.2% ($L = 168$), 25.3% ($L = 336$), and 12.2% ($L = 720$).

Table 4. Comparison of multivariate forecasting results between transformer-based models and MeanIN on DBSm1 under different prediction lengths $L \in \{24, 48, 168, 336, 720\}$. The input sequence length is set to 96. Better results are highlighted in boldface.

Method	**24**		**48**		**168**		**336**		**720**	
Metric	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**
Transformer	0.031	0.078	0.073	0.127	0.248	0.234	0.384	0.300	0.533	0.395
+MeanIN	**0.024**	**0.060**	**0.065**	**0.096**	**0.193**	**0.191**	**0.287**	**0.253**	**0.473**	**0.375**
Informer	0.045	0.092	0.110	0.144	0.320	0.288	0.431	0.351	0.522	0.409
+MeanIN	**0.034**	**0.084**	**0.076**	**0.128**	**0.269**	**0.249**	**0.364**	**0.301**	**0.518**	**0.405**
Reformer	0.045	0.099	0.117	0.183	0.345	0.393	0.675	0.596	0.950	0.749
+MeanIN	**0.036**	**0.074**	**0.097**	**0.135**	**0.333**	**0.346**	**0.487**	**0.401**	**0.894**	**0.634**
Autoformer	0.062	0.126	0.131	0.185	0.333	0.298	0.618	**0.433**	1.086	0.699
+MeanIN	**0.058**	**0.109**	**0.111**	**0.156**	**0.301**	**0.282**	**0.553**	0.454	**1.028**	**0.672**
FERformer	0.649	0.548	0.941	0.701	0.315	0.283	0.502	0.392	0.926	0.623
+MeanIN	**0.627**	**0.518**	**0.859**	**0.655**	**0.289**	**0.270**	**0.452**	**0.370**	**0.867**	**0.614**
Pyraformer	1.270	0.776	1.636	0.929	0.362	**0.357**	0.483	0.428	0.617	0.530
+MeanIN	**0.990**	**0.760**	**1.306**	**0.894**	**0.323**	0.368	**0.398**	**0.412**	**0.486**	**0.488**

4.2.3. Comparison of Feature Scaling for Time-Series Forecasting of Spacecraft Data

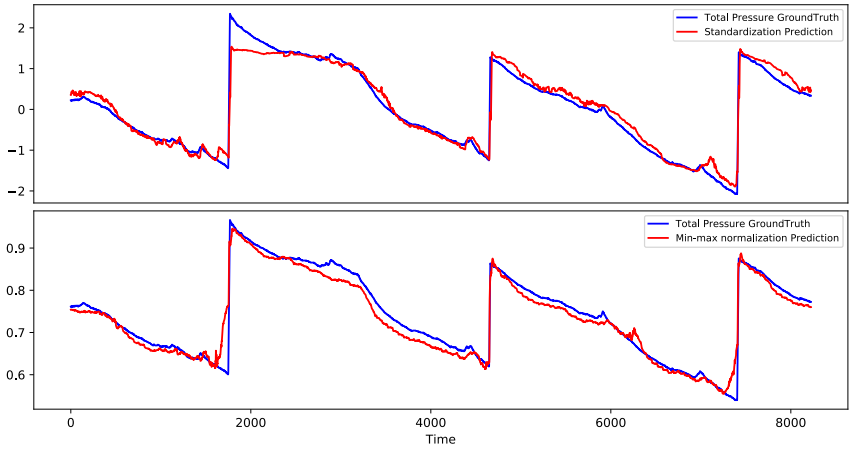
We compare the effect of two feature scaling methods on the prediction effectiveness of the forecasting model. To effectively compare the pros and cons of the two methods, we map the scaled experimental results back to the same scale. Table 5 indicates that min–max normalization performs better than standardization on long-input forecasting ($L > 480$) and long-term forecasting (prediction lengths greater than 168). Standardization has a slight advantage over min–max normalization when both input and predicted sequence lengths are not long ($L = 96$, Pred Length = 24, 48). When the input and output length are increased, min–max normalization significantly outperforms standardization.

Table 5. Comparison of different input and predicted sequence lengths of standardization and min–max normalization. L stands for the length of an input sequence, Std stands for standardization, and MM stands for min–max normalization.

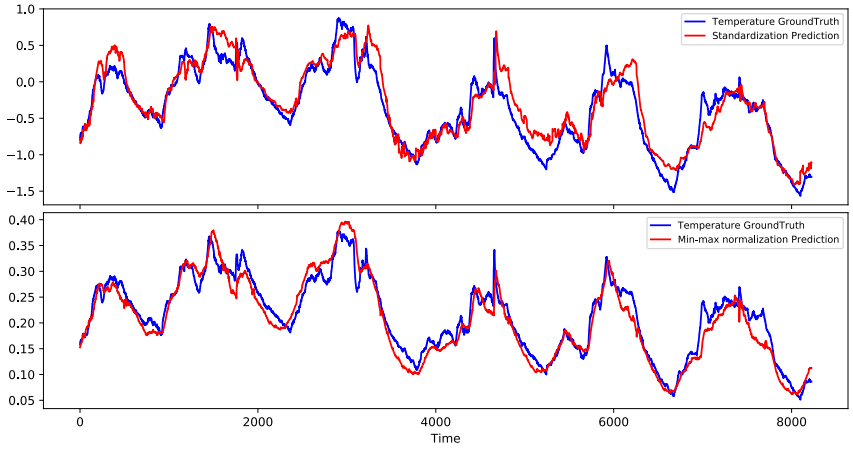
Pred Length	$L = 96$		$L = 168$		$L = 336$		$L = 480$		$L = 720$	
	Std	**MM**	**Std**	**MM**	**Std**	**MM**	**Std**	**MM**	**Std**	**MM**
24	**0.030**	0.054	**0.031**	0.040	**0.041**	0.045	0.055	**0.049**	0.058	**0.043**
48	**0.072**	0.073	0.078	**0.071**	0.086	**0.073**	0.111	**0.085**	0.223	**0.072**
168	0.233	**0.178**	0.206	**0.170**	0.223	**0.221**	0.246	**0.184**	0.254	**0.168**
336	0.271	**0.254**	0.290	**0.243**	0.318	**0.256**	0.357	**0.248**	0.301	**0.229**
720	0.431	**0.267**	0.423	**0.291**	0.417	**0.306**	0.393	**0.326**	0.417	**0.231**

Here, we directly observe the effects of the dataset scaled by different methods. The comparison of using standardization and min–max normalization is shown in Figure 8.

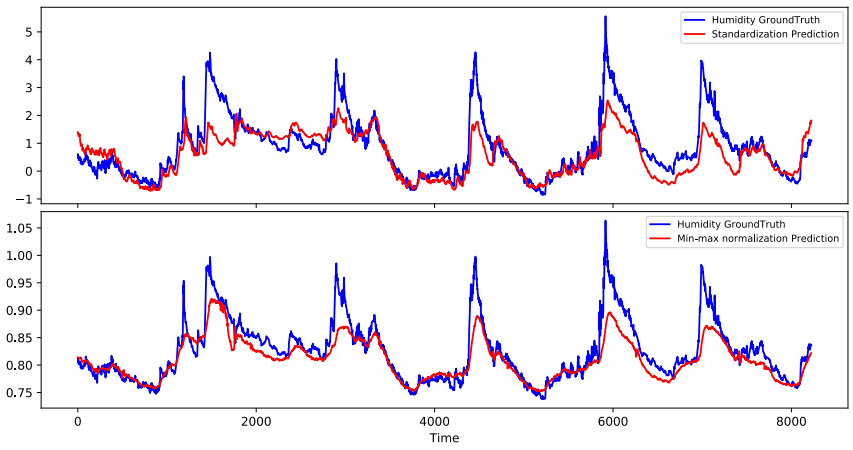
The experimental results indicate that min–max normalization performs slightly better than standardization. As shown in Figure 8a,b, both standardization and min–max normalization fit the curve of the data well on the more regularly varying total pressure data and temperature data. However, the irregular fluctuations of standardization bring about greater errors. This is more clearly reflected in the humidity data.



(a)



(b)



(c)

Figure 8. Comparison of using standardization and min–max normalization on three dataset under sequence length = 720 and prediction length = 168. (**a**) Comparison on total pressure dataset. (**b**) Comparison on temperature dataset. (**c**) Comparison on humidity dataset.

Compared with using min–max normalization, the use of standardization to perform feature scaling on the data produces predictions with more frequent sawtooth fluctuations, whereas the min–max normalization results are smoother. Determining which of the two is more appropriate is difficult because sawtooth fluctuations are widespread in real data.

4.3. Impact of Different Length Start Tokens on Prediction

We find that adopting start tokens of different lengths greatly impacts the model's prediction results. We compare three tokens of different lengths to observe the specific effects when the prediction length is 24 and 48. The result is shown in Figure 9. When the prediction length is 24, the MSE error increases instead with increasing label length, but the opposite result appears when the sequence length is 480. At a prediction length of 48, the MSE error maintains a consistent pattern of variation.

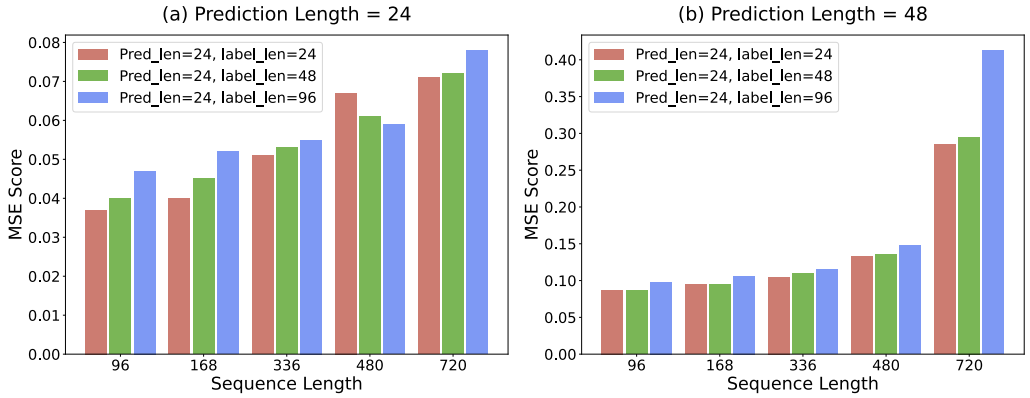


Figure 9. MSE of start token (label length) = 24, 48, or 96 under prediction lengths of 24 and 48. The experiment uses standardization to scale the features of the data.

Empirically, long-start tokens can provide abundant historical information and help the model predict future data well. However, the results are contrary to our intuition. When the output is fixed, increasing the length of the start token increases MSE. Moreover, a long-start token has a great negative impact on a long output.

This result is obtained because increasing the length of the start token brings abundant prior knowledge, and the start token occupies a large proportion of the time series, making the model pay much attention to this part of the data for the current iteration data during training. On the contrary, only a small amount of attention is allocated to this part, leading to model overfitting and reduced generalization ability. Therefore, when training and using the model, we recommend the use of a short-start token.

5. Conclusions and Limitations

5.1. Conclusions

This study proposed a transformer-based framework for time-series forecasting of telemetry data from the environmental control and life support System (ECLSS) in spacecraft. The core contributions and findings of the research are summarized as follows:

- Transformer-based framework: The application of informer, an improved transformer model, demonstrated significant advantages in handling long-sequence time-series forecasting tasks. The model's efficiency was enhanced through ProbSparse self-attention and generative inference mechanisms, which reduced computational complexity and improved forecasting accuracy.
- Mean instance normalization (MeanIN): To address the distribution discrepancies in spacecraft telemetry data, a novel two-stage normalization technique, MeanIN, was

introduced. The results showed that MeanIN effectively improved the performance of transformer-based models by aligning data distributions between the training and testing phases.
- Feature scaling and data preprocessing: The study highlighted the importance of feature scaling methods, with min–max normalization outperforming standardization for spacecraft telemetry data. The preprocessing steps, including downsampling and interpolation for missing data, were crucial in ensuring data quality and consistency.
- Experimental validation: Extensive experiments on real spacecraft telemetry datasets demonstrated that the proposed framework outperformed traditional statistical models and deep learning baselines, particularly for long-term forecasting tasks.

The findings provide valuable insights for leveraging deep learning models in safety-critical aerospace systems, ensuring reliable predictions and anomaly detection in ECLSS telemetry data.

5.2. Limitations

While the current work has demonstrated significant advances, limitations remain to be addressed. Successfully deploying and implementing the algorithms proposed in this paper in the harsh space environment still requires extensive long-term efforts:

- Optimizing computational resources: Developing strategies to further optimize the computational efficiency of the model, ensuring it can be effectively deployed within the resource-constrained environment of spacecraft.
- Compression of transmitted data: Investigating advanced data compression techniques to reduce the volume of telemetry data transmitted to Earth while retaining critical information necessary for accurate analysis [31].

6. Future Research Directions

While this study has made notable advancements, several avenues for future research remain open, particularly in the practical deployment of the proposed framework in real-world space missions. The following areas merit further exploration:

- Integration with onboard systems: Future research should focus on integrating the proposed framework into onboard systems for real-time monitoring and forecasting. This will involve optimizing the model for constrained computational resources and ensuring compatibility with spacecraft hardware.
- Application to in-orbit operations: The framework's potential can be extended to support in-orbit operations, such as anomaly detection and predictive maintenance. By continuously monitoring telemetry data, the system can provide early warnings for potential failures and suggest corrective actions to maintain system stability.
- Scalability to multimodal data: Spacecraft telemetry often includes data from diverse sources (e.g., sensors, logs, and images). Expanding the framework to handle multimodal data will enhance its robustness and applicability.
- Robustness to extreme conditions: Spacecraft operate under extreme and unpredictable conditions. Future studies should investigate the framework's resilience to noisy, sparse, or corrupted data, ensuring reliable performance under challenging circumstances.
- Long-Term operational validation: Conducting long-term validation of the framework on in-orbit missions will provide valuable feedback to refine the models and improve their generalizability.

By addressing these challenges, the proposed methods can become a cornerstone for next-generation spacecraft telemetry analysis, enhancing the safety, reliability, and efficiency of manned space missions.

Author Contributions: Conceptualization, B.S.; methodology, B.S. and B.G.; software, B.S. and B.G.; validation, B.S. and B.G.; formal analysis, B.S. and B.G.; investigation, B.S.; writing—original draft preparation, B.S.; writing—review and editing, B.G.; supervision, W.H., Z.Z., N.Z., J.B., J.W. and J.X.; project administration, W.H., Z.Z., N.Z., J.B., J.W. and J.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded in part by NSFC under grant No. 62125305, No. U23A20339, No. 52300236, and Shaanxi Natural Science Basic Research Program Project No. 2024JC-YBQN-0488.

Data Availability Statement: The data are not available.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Jin, J.; Chen, H.; Cai, W. Growth of rice cells in Shenzhou 8 under microgravity and transcriptome analysis. *Manned Spacefl* **2014**, *5*, 481–490.
2. Cang, H.; Zhang, H.; Han, Y.; Li, N.; Shang, G.; Nie, R.; Ma, J. Protein crystallization experiments aboard Shenzhou-8 spacecraft. *Keji Daobao/Sci. Technol. Rev.* **2012**, *30*, 20–25.
3. Zhang, X.; Chen, Y.; Hu, J. Recent advances in the development of aerospace materials. *Prog. Aerosp. Sci.* **2018**, *97*, 22–34. [CrossRef]
4. Hassanien, A.E.; Darwish, A.; Abdelghafar, S. Machine learning in telemetry data mining of space mission: Basics, challenging and future directions. *Artif. Intell. Rev.* **2020**, *53*, 3201–3230. [CrossRef]
5. Wan, P.; Zhan, Y.; Jiang, W. Study on the satellite telemetry data classification based on self-learning. *IEEE Access* **2019**, *8*, 2656–2669. [CrossRef]
6. Gratius, N.; Wang, Z.; Hwang, M.Y.; Hou, Y.; Rollock, A.; George, C.; Bergés, M.; Akinci, B. Digital Twin Technologies for Autonomous Environmental Control and Life Support Systems. *J. Aerosp. Inf. Syst.* **2024**, *21*, 332–347. [CrossRef]
7. Fujimaki, R.; Yairi, T.; Machida, K. Adaptive limit-checking for spacecraft using relevance vector autoregressive model. In Proceedings of the 8th International Symposium on Artificial Intelligence, Robotics and Automation in Space—iSAIRAS, ESA SP, Munich, Germany, 1 August 2005; Volume 603.
8. Wang, J.; Zhang, S.; Liu, Q.; Du, S.; Guo, Y.C.; Zheng, N.; Wang, F.Y. Conditional uncorrelation and efficient subset selection in sparse regression. *IEEE Trans. Cybern.* **2021**, *52*, 10458–10467. [CrossRef]
9. Liu, X.; Wang, X.; Zhang, W.; Wang, J.; Wang, F. Parallel data: From big data to data intelligence. *Pattern Recognit. Artif. Intell.* **2017**, *30*, 673–682.
10. Ismail Fawaz, H.; Forestier, G.; Weber, J.; Idoumghar, L.; Muller, P.A. Deep learning for time series classification: A review. *Data Min. Knowl. Discov.* **2019**, *33*, 917–963. [CrossRef]
11. Lim, B.; Zohren, S. Time-series forecasting with deep learning: A survey. *Philos. Trans. R. Soc. A* **2021**, *379*, 20200209. [CrossRef]
12. Mahmoud, A.; Mohammed, A. A survey on deep learning for time-series forecasting. In *Machine Learning and Big Data Analytics Paradigms: Analysis, Applications and Challenges*; Springer: Cham, Switzerland, 2021; pp. 365–392.
13. Ibrahim, S.K.; Ahmed, A.; Zeidan, M.A.E.; Ziedan, I.E. Machine learning methods for spacecraft telemetry mining. *IEEE Trans. Aerosp. Electron. Syst.* **2018**, *55*, 1816–1827. [CrossRef]
14. Yang, L.; Ma, Y.; Zeng, F.; Peng, X.; Liu, D. Improved deep learning based telemetry data anomaly detection to enhance spacecraft operation reliability. *Microelectron. Reliab.* **2021**, *126*, 114311. [CrossRef]
15. Wang, Y.; Wu, Y.; Yang, Q.; Zhang, J. Anomaly detection of spacecraft telemetry data using temporal convolution network. In Proceedings of the 2021 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), Glasgow, UK, 17–20 May 2021; pp. 1–5.
16. Li, H.; Gou, L.; Zheng, H.; Li, H. Intelligent fault diagnosis of aeroengine sensors using improved pattern gradient spectrum entropy. *Int. J. Aerosp. Eng.* **2021**, *2021*, 8868875. [CrossRef]
17. Zhao, Y.; Zhao, H.; Ai, J.; Dong, Y. Robust Data-Driven Fault Detection: An Application to Aircraft Air Data Sensors. *Int. J. Aerosp. Eng.g* **2022**, *2022*, 2918458. [CrossRef]
18. Zhang, Y.; Zhao, H.; Ma, J.; Zhao, Y.; Dong, Y.; Ai, J. A Deep Neural Network-Based Fault Detection Scheme for Aircraft IMU Sensors. *Int. J. Aerosp. Eng.* **2021**, *2021*, 3936826. [CrossRef]
19. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 5998–6008.
20. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv* **2014**, arXiv:1409.0473.
21. 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. *arXiv* **2014**, arXiv:1406.1078.

22. Wei, S.; Yang, Y.; Liu, D.; Deng, K.; Wang, C. Transformer-Based Spatiotemporal Graph Diffusion Convolution Network for Traffic Flow Forecasting. *Electronics* **2024**, *13*, 3151. [CrossRef]
23. Chen, Y.; Ren, K.; Wang, Y.; Fang, Y.; Sun, W.; Li, D. Contiformer: Continuous-time transformer for irregular time series modeling. *Adv. Neural Inf. Process. Syst.* **2024**, *36*.
24. Liu, Y.; Hu, T.; Zhang, H.; Wu, H.; Wang, S.; Ma, L.; Long, M. itransformer: Inverted transformers are effective for time series forecasting. *arXiv* **2023**, arXiv:2310.06625.
25. 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 Proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, Virtually, 2–9 February 2021; Volume 35, pp. 11106–11115.
26. Oreshkin, B.N.; Carpov, D.; Chapados, N.; Bengio, Y. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. *arXiv* **2019**, arXiv:1905.10437.
27. Kitaev, N.; Kaiser, Ł.; Levskaya, A. Reformer: The efficient transformer. *arXiv* **2020**, arXiv:2001.04451.
28. Wu, H.; Xu, J.; Wang, J.; Long, M. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 22419–22430.
29. Zhou, T.; Ma, Z.; Wen, Q.; Wang, X.; Sun, L.; Jin, R. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In Proceedings of the International Conference on Machine Learning, Baltimore, MD, USA, 17–23 July 2022; pp. 27268–27286.
30. 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 Proceedings of the Tenth International Conference on Learning Representations (ICLR 2022), Virtual Event, 25–29 April 2022.
31. Rakhmanov, A.; Wiseman, Y. Compression of GNSS data with the aim of speeding up communication to autonomous vehicles. *Remote Sens.* **2023**, *15*, 2165. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.