Time Series Prediction Based on Multi-Scale Feature Extraction

Ruixue Zhang and Yongtao Hao *

CAD Research Center, Tongji University, Shanghai 200092, China; zhangruixue@tongji.edu.cn
* Correspondence: hyongtao492@163.com

Abstract: Time series data are prevalent in the real world, particularly playing a crucial role in key domains such as meteorology, electricity, and finance. Comprising observations at historical time points, these data, when subjected to in-depth analysis and modeling, enable researchers to predict future trends and patterns, providing support for decision making. In current research, especially in the analysis of long time series, effectively extracting and integrating long-term dependencies with short-term features remains a significant challenge. Long-term dependencies refer to the correlation between data points spaced far apart in a time series, while short-term features focus on more recent changes. Understanding and combining these two features correctly are crucial for constructing accurate and reliable predictive models. To efficiently extract and integrate long-term dependencies and short-term features in long time series, this paper proposes a pyramid attention structure model based on multi-scale feature extraction, referred to as the MSFformer model. Initially, a coarser-scale construction module is designed to obtain coarse-grained information. A pyramid data structure is constructed through feature convolution, with the bottom layer representing the original data and each subsequent layer containing feature information extracted across different time step lengths. As a result, nodes higher up in the pyramid integrate information from more time points, such as every Monday or the beginning of each month, while nodes lower down retain their individual information. Additionally, a Skip-PAM is introduced, where a node only calculates attention with its neighboring nodes, parent node, and child nodes, effectively reducing the model’s time complexity to some extent. Notably, the child nodes refer to nodes selected from the next layer by skipping specific time steps. In this study, we not only propose an innovative time series prediction model but also validate the effectiveness of these methods through a series of comprehensive experiments. To comprehensively evaluate the performance of the designed model, we conducted comparative experiments with baseline models, ablation experiments, and hyperparameter studies. The experimental results demonstrate that the MSFformer model improves by 35.87% and 42.6% on the MAE and MSE indicators, respectively, compared to traditional Transformer models. These results highlight the outstanding performance of our proposed deep learning model in handling complex time series data, particularly in capturing long-term dependencies and integrating short-term features.

Keywords: long time series prediction; deep learning; attention mechanism; multi-scale features

MSC: 68T07

1. Introduction

Time series consists of a sequence of data points arranged in chronological order, often collected at equal time intervals, representing any phenomena which change over time. Time series forecasting finds widespread applications across various industries and domains. For instance, in the financial sector, predicting stock prices and exchange rates is crucial for investors and institutions to formulate trading strategies. In supply chain management, the accurate forecasting of product demand helps optimize inventory and reduce costs. In meteorology, time series analysis aids in predicting weather changes and enhancing the early-warning capabilities for natural disasters. These applications underscore the significance of time series forecasting in addressing real-world problems.
By extracting effective features from sequence data and constructing models for predicting future trends, time series forecasting techniques can effectively promote advancements in the field [4].

Numerous scholars have focused on studying time series using deep learning models [5]. Early research predominantly relied on methods based on CNN (convolutional neural networks) [6] and RNN (recurrent neural networks) [7]. These methods excel in capturing nonlinear relationships and handling long-term dependencies, enhancing their modeling capabilities for complex systems like stock market prices and meteorological changes. The data adaptability and processing capacity of these models on a large scale also compensate for the shortcomings of traditional methods [8]. While CNNs effectively capture local features and patterns, their difficulty in capturing global feature information is a limitation [9]. On the other hand, RNNs, particularly with the design of memory units, excel in handling long-term dependency relationships [10], as illustrated in Figure 1a. Notably, LSTM (long short-term memory) networks [11], an improvement based on RNN, successfully address issues such as gradient vanishing, gradient exploding, and the handling of long sequences. However, LSTM tends to lose information from the initial sequence data as they propagate backward. To overcome this, BiLSTM (bidirectional LSTM) [12] emerged, comprising two LSTMs—one backward and one forward—overcoming the traditional RNN challenge of capturing long-term dependencies. However, it has limitations, especially in focusing on critical parts of a sequence, and may exhibit differences in the attention points for different positions, particularly in longer sequences. Meanwhile, methods based on Transformers have gradually become a hotspot in time series research. Initially designed for natural language-processing tasks, the Transformer model [13], with its self-attention mechanism, excels in capturing global relationships within a sequence. Applying Transformers to time series forecasting, especially with the introduction of variants such as the multi-head attention mechanism, enables the model to flexibly handle correlations at different time points [14].

![Figure 1. Existing sequence modeling architectures for time series forecasting.](image)

Addressing the difficulty in capturing long-term dependencies and short-term features in the long time series mentioned above, this paper proposes a multi-scale feature extraction prediction model based on Pyraformer improvement. We optimized the core mechanism of Pyraformer—PAM (pyramid attention module)—by improving the inter-layer connection strategy, enhancing the model’s ability to capture periodic features of time series. This enables more effective multi-scale feature extraction and time series prediction. Additionally, we updated the CSCM (coarser-scale construction module) to align with Skip-PAM, optimizing our data processing flow. These upgrades have significantly boosted the model’s
accuracy and efficiency in handling complex time series data. The main contributions are as follows:

- Introduction of the Skip-PAM aimed at enabling deep learning models to effectively capture long-term and short-term features in long time series. In the encoder, the implementation of a pyramid attention mechanism processes previous feature vectors to obtain long-term dependencies and local features.

- Improvement of the CSCM aimed at establishing a pyramid data structure for the Skip-PAM. After encoding the input data, coarse-grained feature information across different time step lengths is obtained through the proposed feature convolution.

- Based on three time series datasets, our approach is compared with three baseline methods, achieving a favorable performance.

2. Recent Work

2.1. Time Series Prediction Based on CNN and LSTM

When dealing with long time series data, challenges often arise due to long-term dependency and the loss of short-term information. For instance, in the case of LSTM handling long sequences, as data propagate backward, information features gradually decay over time. Similarly, when using CNN to process time series data, its inherent convolutional operations primarily focus on local neighborhoods, potentially overlooking crucial information from past time points, especially when such information is vital for current predictions. Researchers have explored various methods to address these challenges [15,16]. Oord [17] introduced a CNN variant known as Causal CNN, specifically designed for processing sequence data (Figure 1b). In Causal CNN, the output is not influenced by future inputs, ensuring that convolution operations use only current and past data points. This design prevents information leakage during the prediction of future values in time series data, maintaining the causal sequence of time. However, due to its sequential processing of one time point at a time, Causal CNN requires more computational steps and has a higher time complexity when handling long sequences, limiting its effectiveness in capturing long-term dependencies [18]. Building upon this, Bai [19] proposed a TCN (temporal convolutional network), using stacked causal convolutions to capture the temporal dependencies of time series data. It employs dilated convolutions to increase the receptive field, covering a longer input history (Figure 1d). TCN’s causal convolutions and dilation structure enable it to predict the entire output sequence in a single forward pass, significantly reducing the time complexity. TCN also incorporates residual structures to address the issue of gradient vanishing during training, enhancing network training efficiency. Chen [20] introduced a two-stage attention mechanism model called TPM, which divides the original time series into short-term and long-term features. Short-term features are extracted using CNN, while long-term features are obtained through a piecewise linear regression method (PLR) [21]. The encoder and decoder extract short-term and long-term features, respectively, and combining multi-scale temporal feature information leads to improved prediction results. Lai [22] proposed LSTNet (long-and short-term time series network), utilizing CNN and RNN to capture the short-term and long-term features of time series. It addresses the scale insensitivity issue of neural network models by incorporating an autoregressive model. For capturing long-term features, an innovative Skip-RNN is introduced, allowing backward propagation with specified step lengths (Figure 1c). Skip-RNN can obtain global feature information with fewer nodes. Woo [23] introduced CoST to separate trends and seasonality in time series data and used contrastive learning to capture features in the time and frequency domains. CoST’s contrastive loss in the time domain aids the model in learning trend features, while the frequency domain’s contrastive loss focuses on seasonal features. This approach extracts features at multiple scales from a single time series, where these features are independent but collectively contribute to the prediction task, leading to a more comprehensive understanding of data and an improved prediction accuracy.
2.2. Transformer-Based Time Series Prediction Approaches

In recent years, a series of innovative Transformer-based models have emerged, leveraging advanced architectures and algorithms to capture intricate temporal dependencies, thereby enhancing prediction accuracy and efficiency. Informer [24] effectively addresses long-term time series by utilizing the ProbSparse self-attention mechanism. It optimizes the computational complexity of traditional Transformer models, allowing them to handle longer sequences without sacrificing performance. Its design addresses a fundamental issue in long sequence prediction: how to capture deep dynamics in time series while maintaining computational efficiency. Autoformer [25] is a model that combines autoregressive statistical models with deep learning techniques. Its autoregressive correlation structure is specifically designed to identify and leverage the periodic features of time series, making it more suitable for time series prediction tasks with distinct periodic and trend features. Reformer [26] focuses on addressing issues with long sequences by reducing the computational complexity of self-attention using hash techniques and minimizing memory consumption through reversible layers. Pyraformer [27] adopts a different strategy, utilizing a pyramid structure to capture multi-scale features in time series, revealing rich hierarchical and complex dynamics in sequence data (Figure 2a). This design meets the demand for a more refined understanding of time resolution, particularly crucial in areas like financial market analysis. FEDformer [28] introduces a novel perspective on analyzing time series by performing self-attention in the frequency domain. It uses Fourier and wavelet transforms to unveil patterns hidden behind periodic and seasonal variations in time series. Through intelligent transformations of the data, FEDformer captures important features in the frequency domain that may not be easily discernible in the time domain.

![Figure 2. PAM and Skip-PAM. The point of divergence lies in the node connection method.](image)

3. Model

The overall framework of the proposed MSFformer is illustrated in Figure 3. Building upon the Pyraformer architecture, our model enhances the attention mechanism in the encoder by innovatively introducing a pyramid attention mechanism across different time step lengths. To construct multi-scale temporal information, we employ feature convolution to build the CSCM, extracting temporal information at a coarser granularity.
This paper introduces an innovative framework, called the Skip-Pyramidal Attention Module (Skip-PAM), as part of the MSFormer model. The objective is to enhance the capability of deep learning models in handling time series data through a multi-level attention mechanism. The core idea of this mechanism is to process input data at different time step lengths, enabling the model to capture time dependencies at various granularities. At lower levels, the model may focus on short-term, fine-grained patterns. In contrast, at higher levels, it can capture more macroscopic trends and periodicities. Unlike PAM, which focuses on weekly and monthly information, Skip-PAM pays more attention to information such as every Monday or the beginning of each month. This kind of multi-scale processing allows the model to capture diverse time-dependent relationships across different levels.

As shown in Figure 2b, Skip-PAM extracts information from the time series at multiple scales through the attention mechanism constructed by the time feature tree. This process involves both intra-scale connections and inter-scale connections. Intra-scale connections involve performing attention calculations between a node and its adjacent nodes within the same scale layer. Inter-scale connections involve attention calculations between a node and its parent node (each parent node having P children) and C child nodes. Specifically, for a node \( n_s^n \), \( s \in [1, S] \) denotes the layer scale from the bottom to the top (S layers in total), and \( l \in [1, l_s] \) represents the \( l \)-th node in that layer.

\[
A_s^i = \left\{ n_s^j : |j - l| \leq \frac{A - 1}{2}, 1 \leq j \leq \frac{L}{C^{s-1}} \right\} \quad (1)
\]
\[
C_s^i = \left\{ n_s^{j-1} : j = l + iC, i = 0, 1, 2 \ldots \text{ if } s \geq 2 \text{ else } \varnothing \right\} \quad (2)
\]
\[
P_s^i = \left\{ n_s^{j+1} : j = l \mod P \text{ if } s \leq S - 1 \text{ else } \varnothing \right\} \quad (3)
\]
\[
N_s^i = A_s^i \cup C_s^i \cup P_s^i \quad (4)
\]
where \( A \) represents the adjacent nodes within the scale, \( A \) denotes the number of adjacent nodes within the scale, \( C \) signifies the child nodes, \( P \) represents the parent node, and \( N \) is the total number of nodes for which the attention mechanism should be computed. So, for the node \( n \), attention can be represented as follows:

\[
\hat{x}_i = \sum_{l \in N_i} \frac{\exp \left( \frac{q_i k_l^T}{\sqrt{d_x}} \right) v_l}{\sum_{l \in N_i} \exp \left( \frac{q_i k_l^T}{\sqrt{d_x}} \right)}
\]

where \( q_i \) is the query matrix corresponding to \( x_i \); \( k_l \) and \( v_l \) are the key and value matrices, respectively; \( d_x \) is the sequence length; and \( \hat{x}_i \) is the output result through attention.

Through this pyramid attention mechanism, combined with the multi-scale feature convolution mentioned later, a powerful feature extraction network is formed. It can adapt to dynamic changes across various time scales, whether they are short-term oscillations or long-term evolutions.

3.2. Coarser-Scale Construction Module

To adapt to cross-stride feature extraction, we designed a FCNN (feature convolutional layer), as shown in Figure 4. Specifically, it first extracts feature vectors through the desired cross-stride step, concatenates them together, and then performs convolutional operations using a convolutional kernel with a size of \( [l/\text{step}] \) and a stride of \( [l/\text{step}] \). The formula is as follows:

\[
\text{Output} = \text{FCNN}(\text{Input})
\]

**Figure 4.** Feature convolution layer. The far left shows the original input time series, which are concatenated together through our specified stride (indicated by the same color) to form the concatenated block, as shown in the middle cube in the image. Then, they each go through convolution operations to produce our output. The checkered rectangle depicted above the arrow on the right represents the convolution kernel.

In order to better apply the aforementioned Skip-PAM structure, we constructed a feature tree and designed a CSCM. As shown in Figure 5, we first passed the input sequence through a linear layer to expand the feature dimension to a fixed dimension. Then, we gradually obtained feature information at different scales through feature convolutional layers and concatenated them to form a pyramid-shaped feature structure. Finally, we used a linear layer to restore the feature dimension.

\[
M_0 = \text{Linear}_1(X)
\]

\[
M_1 = \text{FCNN}_1(M_0)
\]

\[
M_2 = \text{FCNN}_2(M_1)
\]

\[
M_3 = \text{FCNN}_3(M_2)
\]
\[
M_{\text{CSCM}} = \text{Linear}_2([M_0; M_1; M_2; M_3]) \tag{11}
\]

where \( X \) is the input to the CSCM, \text{Linear} represents the linear layer, FCNN is the feature convolution module mentioned in Equation (6), \( M_1, M_2, \) and \( M_3 \) are the consecutive results of feature convolution, \( M_{\text{CSCM}} \) is the output of the CSCM, and the ’;’ operation is used to concatenate \( M_0, M_1, M_2, \) and \( M_3 \) along the time dimension.

By passing our data through such a CSCM, we obtained time feature information at different granularities. By stacking the feature convolutional layers, we built a pyramid-shaped feature tree. This not only enabled us to understand the data at multiple levels but also provided a solid foundation for the implementation of the Skip-PAM.

4. Experiment
4.1. Experimental Data

This paper validates the model using three time series datasets with different levels of stationarity, as detailed below:

- **ETTH1**: The ETTh dataset [24] comprises temperature records collected from transformers and six indicators related to voltage load in the period from July 2016 to July 2018. The ETTH1 data, with their hourly collection frequency, are ideal for evaluating the model’s capability to capture cyclical daily patterns and longer-term seasonal trends in energy usage related to environmental temperature fluctuations.

- **ETTm1**: As a more granular subset of the ETTh dataset, ETTm1 provides data at 15 min intervals. This higher-resolution dataset challenges the model to discern subtler short-term variations and abrupt changes in electrical load, which are critical for operational decisions in energy distribution and for responding to rapid demand shifts.

- **Electricity** [29]: This dataset records the hourly electricity consumption from 2012 to 2014 for 321 customers. It serves as a rich source for examining consumer behavior over multiple years, including variations in consumption due to individual lifestyle patterns, societal events, and differing business operations.

As outlined in Table 1, all three datasets are partitioned into training, validation, and test sets with a ratio of 6:2:2. Moreover, the data within these three datasets are all in a floating-point format.
Table 1. Dataset. presents the feature counts, sampling frequencies, data point numbers, as well as the mean, standard deviation, minimum, and maximum for the three datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Feature Counts</th>
<th>Sampling Frequencies</th>
<th>Data Point Numbers</th>
<th>Mean</th>
<th>Std</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETTh1</td>
<td>7</td>
<td>1 h</td>
<td>121,940</td>
<td>4.58</td>
<td>25.59</td>
<td>-25.09</td>
<td>46.01</td>
</tr>
<tr>
<td>ETTm1</td>
<td>7</td>
<td>15 min</td>
<td>487,760</td>
<td>4.59</td>
<td>25.62</td>
<td>-26.37</td>
<td>46.01</td>
</tr>
<tr>
<td>Electricity</td>
<td>321</td>
<td>1 h</td>
<td>7,506,264</td>
<td>2551.88</td>
<td>231,879,417.19</td>
<td>0.00</td>
<td>764,500.00</td>
</tr>
</tbody>
</table>

4.2. Evaluation Metrics

In this study, we employed three widely used performance evaluation metrics to assess the predictive accuracy of the model: mean square error (MSE), mean absolute error (MAE), and mean absolute percentage error (MAPE). Lower values of these three metrics indicate a superior predictive performance of the model [30].

MSE: The MSE measures the average of the squared differences between the predicted values $y_i'$ and the actual values $y_i$. This metric gives higher penalties to larger errors, emphasizing the evaluation of larger prediction discrepancies [31]. The formula is expressed as follows:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - y_i')^2$$  \hspace{1cm} (12)

MAE: Similar to the MSE, this metric calculates the average absolute differences between the predicted values $y_i'$ and the actual observed values $y_i$. It reflects the average magnitude of the deviations of the predicted values from the true values [32]. The formula is expressed as follows:

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |y_i - y_i'|$$  \hspace{1cm} (13)

MAPE: The MAPE is a measure that calculates the average of the percentage differences between the predicted values $y_i'$ and the actual values $y_i$. It provides insight into the accuracy of predictions in terms of their deviation from the actual values as a percentage, which can be particularly useful when comparing the performance of forecasting models across different scales of data. The formula is expressed as follows:

$$\text{MAPE} = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{y_i - y_i'}{y_i} \right| \times 100\%$$  \hspace{1cm} (14)

4.3. Parameter Settings and Experiment Details

During the experimental process in this chapter, four different prediction time point lengths were set for the datasets: 96, 192, 336, and 720. For the ETTh1 and Electricity datasets, these horizons correspond to future predictions of 4 days, 8 days, 2 weeks, and 1 month, respectively. For the ETTm1 dataset, they represent predictions for 1 day, 2 days, 3.5 days, and 7.5 days into the future, respectively. The prediction task aims at multi-feature forecasting. The encoder layers were set to four, and the attention heads were set to six. Using a time series with a length of 96 as the input, a stride length window of [4, 4, 6] was applied to construct the features, resulting in a tree structure with layers [96, 24, 6, 1]. In the construction process of the Skip-PAM, the inter-scale connecting node, i.e., the inner size, was set to five. In the data preprocessing step, we applied different normalization techniques tailored to the characteristics of each dataset. Specifically, for the ETTh dataset, we employed the Z-score normalization method, which transforms the data using the formula $\text{normalized} = (\text{original} - \text{mean}) / \text{std}$. For the Electricity dataset, on the other hand, we utilized a mean-adjusted proportional scaling method, processing the data with the formula $\text{normalized} = \text{original} / (\text{mean} + 1)$.

We adopted Adam as our optimization algorithm, with an initial learning rate of $10^{-3}$, which was decreased to one-tenth of its value at the end of each epoch. We set the
4.4. Comparative Experiments

To validate the effectiveness of the proposed MSFformer model in this chapter, we conducted training and compared the results with three baseline models. From Table 2, it can be concluded that, across all the datasets, especially in the case of the ETTh1 and ETTm1 datasets, the MSFformer model demonstrates the best overall performance, followed by Pyraformer. This suggests that MSFformer is particularly effective in long time series prediction tasks, leveraging information from multiple scales. The Skip-PAM allows the model to combine both long-term and short-term features, enhancing its predictive capabilities. For the ETTh1 dataset, MSFformer outperforms all the other comparison methods in terms of the MSE and MAE metrics but performs poorly on the MAPE metric. The higher values of the MAPE, despite the MSE imposing heavier penalties on larger errors, may be attributed to the asymmetry of the MAPE—it penalizes negative errors (when the predicted value exceeds the actual value) more than positive errors. On the ETTm1 dataset, MSFformer significantly surpasses the Transformer and Informer models across all the evaluation metrics and is also slightly better than Pyraformer in most cases. This underscores the superior capability of MSFformer in capturing subtle changes within short intervals, especially at shorter prediction lengths such as 96 and 192. For the Electricity dataset, MSFformer exhibits suboptimal performance in the MSE metric, while achieving the best performance in the MAE and MAPE among all the model methods. This discrepancy might be attributed to larger prediction errors at certain time points, where the MSE amplifies such errors, whereas the MAE and MAPE indicate that the MSFformer model still excels in time series prediction tasks. Overall, the MSFformer model outperforms the three baseline methods. In terms of the MAE, it achieves a maximum improvement of 26.95%, 31.03%, and 35.87% over the Transformer model in the three datasets. Additionally, for the ETTh1 and ETTm1 datasets, the MSFformer model achieves a maximum improvement of 34.84% and 42.60%, respectively, in the MSE metric.

The visual analysis presented in Figure 6 unambiguously substantiates the pronounced superiority of our proposed MSFformer model in terms of precision and the dynamic cap-
turing of time series data. Employing a multi-scale feature extraction strategy, the model demonstrates exemplary performance across diverse forecasting horizons, a fact which becomes evident when juxtaposed with the Pyraformer model’s comparative analysis. Particularly with regard to the identification and prediction of peak phenomena in time series, MSFformer manifests a heightened sensitivity and a superior predictive prowess, attributable to its intricate mechanism for discerning both long- and short-term dependencies. Even as the forecasting window extends, posing increased predictive challenges, the MSFformer maintains laudable accuracy. Furthermore, an augmentation in input length bolsters the model’s performance, highlighting the efficacy of multi-scale feature extraction in harvesting extensive historical insights and pinpointing pivotal temporal dependencies. These findings not only emphasize the robustness of the MSFformer in navigating the complexities of time series prediction but also its adaptability across varying data scales, enhancing its potential for wide-ranging applications in complex forecasting scenarios.

Figure 6. Comparison experiment of the oil temperature feature in the ETTh1 dataset. Each subplot displays the comparison between the actual values, Pyraformer predictions, and MSFformer predictions. The figure consists of a total of sixteen subplots arranged in four rows and four columns, where each row represents a different prediction length of 96, 192, 336, and 720, respectively, and each column corresponds to a different input length of 96, 192, 336, and 720, respectively.
To demonstrate that our approach can integrate information across multiple temporal scales, we utilized a synthetic hourly dataset [27], with multi-range dependencies, for our experiments. This dataset was synthesized by linearly combining three sine functions with different periods: 24, 168, and 720 h, representing daily (short-term), weekly (mid-term), and monthly (long-term) temporal dependencies, respectively. In the experimental setup, both the input length and the prediction length were set to 720, with all the basic settings remaining consistent with previous comparative experiments except for the window size and inner size. Given that the synthetic time series exhibited long-range correlations in both their deterministic and stochastic components, it was crucial for the model to effectively capture such dependencies to accurately forecast the subsequent 720 data points. The experimental results are summarized in Table 3, where MSFformer \(_{24,6,5}\) has a window size of \([24, 6, 5]\), corresponding to a tree structure of \([720, 30, 5, 1]\) and an inner size of 7, and MSFformer \(_{20,6,6}\) has a window size of \([20, 6, 6]\), corresponding to a tree structure of \([720, 36, 6, 1]\), and an inner size of 13. Both configurations of our MSFformer variant achieved the best results among all the methods, with MSFformer \(_{20,6,6}\) showing a 10.37% improvement over the best Pyraformer method in the MSE and a 5.54% improvement in the MAE. Figure 7 visualizes the forecasting results of MSFformer \(_{20,6,6}\). Therefore, in long-term time series forecasting tasks, our approach, which integrates features of multiple granularities and captures characteristics across time steps, is indeed capable of effectively capturing multi-scale information.

**Table 3.** Experiments results on the synthetic dataset. The table describes the MSE and MAE values of different methods on the synthetic dataset, with the best results highlighted in bold and the second-best results underlined.

<table>
<thead>
<tr>
<th>Model</th>
<th>MSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transformer</td>
<td>3.550</td>
<td>1.477</td>
</tr>
<tr>
<td>Informer</td>
<td>7.546</td>
<td>2.092</td>
</tr>
<tr>
<td>Pyraformer</td>
<td>1.176</td>
<td>0.849</td>
</tr>
<tr>
<td>MSFformer(_{24,6,5})</td>
<td>1.103</td>
<td>0.818</td>
</tr>
<tr>
<td>MSFformer(_{20,6,6})</td>
<td><strong>1.054</strong></td>
<td><strong>0.802</strong></td>
</tr>
</tbody>
</table>

**Figure 7.** Visualization results for the synthetic dataset. The figure presents a comparative experiment between the actual values and the predicted values by the MSFformer model, with both the input and prediction lengths set to 720. The horizontal axis represents the time points, while the vertical axis represents the values.
4.5. Ablation Experiments

In the MSFformer model, the key components are Skip-PAM and CSCM. Since CSCM serves Skip-PAM, it is not feasible to remove the CSCM while retaining the Skip-PAM. Therefore, in our ablation experiments, we compared the original model with a model where the Skip-PAM had been removed.

From Table 4, it can be concluded that the models without the Skip-PAM exhibit a poorer performance across all three datasets. This suggests that the pyramid feature model we constructed effectively connects the required feature information, disregarding nodes with a low correlation. This not only enhances the accuracy of the model but also, to some extent, reduces the computational complexity, accelerating the training iterations of the model. In the ablation experiments, it is observed that the MSFformer, compared to the model without the Skip-PAM, achieves an average improvement of 10.34%, 5.44%, and 5.24% in MAE across the three datasets. Additionally, in terms of the MSE, there is an average improvement of 11.97%, 8.58%, and 1.34%. This indicates that the Skip-PAM component involved in MSFformer performs better on stationary time series, more accurately capturing the embedded temporal features.

Table 4. MSFformer Ablation Experiments. This table describes the results of the ablation experiments conducted on the MSFformer model for three datasets. The MSE and MAE are used as the evaluation metrics, with bold formatting indicating lower values, which are desirable.

<table>
<thead>
<tr>
<th>Model</th>
<th>MSE</th>
<th>MAE</th>
<th>MSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETTh1</td>
<td>0.6598</td>
<td>0.5774</td>
<td>0.7530</td>
<td>0.6487</td>
</tr>
<tr>
<td></td>
<td>0.6861</td>
<td>0.6009</td>
<td>0.8308</td>
<td>0.6957</td>
</tr>
<tr>
<td></td>
<td>0.8359</td>
<td>0.6898</td>
<td>0.9078</td>
<td>0.7418</td>
</tr>
<tr>
<td></td>
<td>0.9312</td>
<td>0.7424</td>
<td>1.0365</td>
<td>0.8223</td>
</tr>
<tr>
<td>ETTm1</td>
<td>0.4926</td>
<td>0.4831</td>
<td>0.5541</td>
<td>0.5082</td>
</tr>
<tr>
<td></td>
<td>0.4955</td>
<td>0.4923</td>
<td>0.5365</td>
<td>0.5071</td>
</tr>
<tr>
<td></td>
<td>0.7266</td>
<td>0.6175</td>
<td>0.7585</td>
<td>0.6262</td>
</tr>
<tr>
<td></td>
<td>0.7825</td>
<td>0.6410</td>
<td>0.8831</td>
<td>0.7326</td>
</tr>
<tr>
<td>Electricity</td>
<td>0.4698</td>
<td>0.2636</td>
<td>0.4816</td>
<td>0.2791</td>
</tr>
<tr>
<td></td>
<td>0.7943</td>
<td>0.2619</td>
<td>0.8023</td>
<td>0.2667</td>
</tr>
<tr>
<td></td>
<td>1.4602</td>
<td>0.2843</td>
<td>1.4744</td>
<td>0.3011</td>
</tr>
<tr>
<td></td>
<td>4.1038</td>
<td>0.3290</td>
<td>4.1433</td>
<td>0.3576</td>
</tr>
</tbody>
</table>

Figure 8 shows the results of the Electricity dataset at different prediction lengths under the same input conditions. The introduction of the Skip-PAM significantly enhances the model’s capability to capture long-term trends. By integrating multi-scale temporal information, the Skip-PAM enables the model to grasp long-term features more accurately, as evidenced by the closer alignment between the trend tracking of the MSFformer with Skip-PAM and the actual values. Moreover, the addition of the Skip-PAM not only improves the model’s ability to capture long-term trends but also enhances its precision in detecting local features, particularly in identifying peak values. This is vividly demonstrated in all sub-figures of Figure 8, especially the peaks near position 59 (a) and at positions 150 (b), 325 (c), and 600 (d), highlighting our method’s efficacy in preserving fine-grained features while extracting large-scale characteristics. In conclusion, the ablation study vividly illustrates the critical importance of the Skip-PAM component for the MSFformer model. It not only substantially improves the model’s proficiency in forecasting long-term trends and accurately detecting local features but also demonstrates the efficiency and accuracy of our approach in tackling time series forecasting tasks. This outcome emphasizes the significance of incorporating multi-scale feature integration when designing time series prediction models.
Model MSFformer Without Skip-PAM

{3, 4, 8}, forming tree structures of {96, 32, 6, 1} and {96, 32, 8, 1}, respectively.

{4, 4, 6}, resulting in a feature tree structure of {96, 24, 6, 1} when using an input length of 96.

window parameters, inner size parameters, as well as the number of attention heads and
Additionally, for an input length of 96, we designed two other window sizes: {3, 6, 6} and

{96, 192, 336, 720}, respectively.

(1) Study on Stride Window Size Parameter

Thus, larger stride lengths can better capture these long-term temporal dependencies,
parameter was set to five. To investigate the impact of this parameter, experiments were

(2) Inner Size Parameter Study

Within the Skip-PAM, another crucial parameter determining the attention structure is
the inner size parameter. This parameter signifies how many neighboring nodes a particular
node should connect with within a given scale. In the previous experiments, the inner size
parameter was set to five. To investigate the impact of this parameter, experiments were
also conducted with inner size parameters set to 3 and 11.

From Figure 9, it can be summarized that, for the datasets ETTh1 and ETTm1, setting
the inner size to five produces the best results. This might be attributed to the fact that
setting the inner size to three, during which the model captures a very short time range,
makes it insufficient for capturing long-term dependencies and patterns in the time series.
On the other hand, setting the inner size to 11 may introduce too much non-critical infor-

Figure 8. Ablation study on feature MT_121 in the Electricity dataset. Each image shows a comparison among the true values, the MSFformer-predicted values, and the predicted values after ablation, using the MT_121 feature of the Electricity dataset (a–d). The input time steps are consistently 96, with the row having prediction lengths of [96, 192, 336, 720], respectively.

4.6. Hyperparameter Study

To explore the impact of key parameters in Skip-PAM and crucial encoder parameters
on the overall performance of the model, we will hereby conduct a study on the stride
window parameters, inner size parameters, as well as the number of attention heads and encoder layers.

(1) Study on Stride Window Size Parameter

The most crucial parameter in Skip-PAM and CSCM is the stride window size, determining the scale and dimension of the feature tree we construct and influencing the degree of correlation between nodes. In the previous experiments, we utilized a stride window of {4, 4, 6}, resulting in a feature tree structure of {96, 24, 6, 1} when using an input length of 96. Additionally, for an input length of 96, we designed two other window sizes: {3, 6, 6} and {3, 4, 8}, forming tree structures of {96, 32, 6, 1} and {96, 32, 8, 1}, respectively.

From Table 5, it can be summarized that, for the ETTh1 and ETTm1 datasets, the
configuration of {4, 4, 6} performs the best among the three designed stride window configurations. This configuration may offer a more balanced pyramid shape, providing a stable reference framework for time series prediction. This implies that the model can more effectively capture key patterns in the time series, being less susceptible to random fluctuations. Therefore, in terms of average performance and error rates, this stride configuration is superior compared to the others. For the Electricity dataset, the stride configuration of {3, 4, 8} performs the best. This could be related to the characteristics of the dataset, as electricity usage data typically involve a broader time range and more complex temporal dependencies. In electricity usage scenarios, not only recent usage is important, but also correlations with more distant past time points need consideration. Thus, larger stride lengths can better capture these long-term temporal dependencies, thereby improving the model’s accuracy in predicting electricity usage trends.

(2) Inner Size Parameter Study

Within the Skip-PAM, another crucial parameter determining the attention structure is
the inner size parameter. This parameter signifies how many neighboring nodes a particular
node should connect with within a given scale. In the previous experiments, the inner size
parameter was set to five. To investigate the impact of this parameter, experiments were
also conducted with inner size parameters set to 3 and 11.

From Figure 9, it can be summarized that, for the datasets ETTh1 and ETTm1, setting
the inner size to five produces the best results. This might be attributed to the fact that
setting the inner size to three, during which the model captures a very short time range,
makes it insufficient for capturing long-term dependencies and patterns in the time series.
On the other hand, setting the inner size to 11 may introduce too much non-critical infor-
mation, leading to redundancy or less relevant information in the data processed by the model. In contrast, setting the inner size to five provides a balance, capturing sufficient time information to identify critical patterns while avoiding excessive interference from irrelevant information. For the Electricity dataset, setting the inner size to three yields the best results. This might be closely related to the characteristics of electricity usage data, where usage is often more closely related to adjacent time periods. Therefore, setting the inner size to 5 and 11 may introduce historical information less relevant to the immediate electricity usage trend, thereby reducing prediction accuracy.

Table 5.Stride window size parameter study. This table describes the performance of the MSFformer on three datasets using different sizes of stride window time intervals. The time stride configurations are\{4, 6, 8\}, \{3, 6, 8\}, and \{3, 4, 8\}. The evaluation metrics are the MSE and the MAE, with bold formatting indicating the best performance in these two metrics.

<table>
<thead>
<tr>
<th>Window Size</th>
<th>Metric</th>
<th>[4, 4, 6]</th>
<th>[3, 6, 6]</th>
<th>[3, 4, 8]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MSE</td>
<td>MAE</td>
<td>MSE</td>
<td>MAE</td>
</tr>
<tr>
<td>ETTh1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>0.6598</td>
<td>0.5774</td>
<td>0.7577</td>
<td>0.6531</td>
</tr>
<tr>
<td>192</td>
<td>0.6681</td>
<td>0.6009</td>
<td>0.8374</td>
<td>0.6982</td>
</tr>
<tr>
<td>336</td>
<td>0.8339</td>
<td>0.6898</td>
<td>0.9116</td>
<td>0.7431</td>
</tr>
<tr>
<td>720</td>
<td>0.9312</td>
<td>0.7424</td>
<td>1.0564</td>
<td>0.8337</td>
</tr>
<tr>
<td>ETTm1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>0.4926</td>
<td>0.4831</td>
<td>0.5125</td>
<td>0.4993</td>
</tr>
<tr>
<td>192</td>
<td>0.4955</td>
<td>0.4923</td>
<td>0.5893</td>
<td>0.5589</td>
</tr>
<tr>
<td>336</td>
<td>0.7266</td>
<td>0.6175</td>
<td>0.6934</td>
<td>0.6302</td>
</tr>
<tr>
<td>720</td>
<td>0.7825</td>
<td>0.6410</td>
<td>0.9332</td>
<td>0.7443</td>
</tr>
<tr>
<td>Electricity</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>0.4698</td>
<td>0.2636</td>
<td>0.4509</td>
<td>0.2401</td>
</tr>
<tr>
<td>192</td>
<td>0.7943</td>
<td>0.2619</td>
<td>0.7830</td>
<td>0.2568</td>
</tr>
<tr>
<td>336</td>
<td>1.4602</td>
<td>0.2843</td>
<td>1.4362</td>
<td>0.2740</td>
</tr>
<tr>
<td>720</td>
<td>4.1038</td>
<td>0.3290</td>
<td>4.0574</td>
<td>0.3261</td>
</tr>
</tbody>
</table>

(3) Study on Attention Heads and Encoder Layers Parameters

The proposed model in this study is fundamentally an improvement based on the Transformer model. Therefore, the number of attention heads and encoder layers is crucial for the performance of the MSFformer model. In the previous experiments, the number of attention heads and encoder layers were set to six and four, respectively. To investigate the impact of these two parameters, experiments were conducted, with the number of attention heads being \{4, 6, 8\} and the encoder layers being \{4, 6, 8\}.

From Figure 10, it can be concluded that, for the datasets ETTh1 and ETTm1, the model performs best when the number of attention heads is six and the number of encoder layers is four. In this configuration, the model can better capture complex relationships and features in the data. Increasing the number of encoder layers can lead to overfitting on the dataset, resulting in a poorer overall performance. For the Electricity dataset, the model performs best when the number of attention heads is eight and the number of encoder layers is four. This may be due to the larger feature dimensions of the dataset. By increasing the number of attention heads, the model has a larger parameter capacity, allowing it to fully capture complex relationships and features in the data.
From Figure 9, it can be summarized that, for the datasets ETTh1 and ETTm1, setting the inner size to five produces the best results. This might be attributed to the fact that setting the inner size to three, during which the model captures a very short time range, makes it insufficient for capturing long-term dependencies and patterns in the time series. On the other hand, setting the inner size to 11 may introduce too much non-critical information, leading to redundancy or less relevant information in the data processed by the model. In contrast, setting the inner size to five provides a balance, capturing sufficient time information to identify critical patterns while avoiding excessive interference from irrelevant information. For the Electricity dataset, setting the inner size to three yields the best results. This might be closely related to the characteristics of electricity usage data, where usage is often more closely related to adjacent time periods. Therefore, setting the inner size to 5 and 11 may introduce historical information less relevant to the immediate electricity usage trend, thereby reducing prediction accuracy.

Figure 9. Inner size parameter study This figure illustrates the results of the inner size parameter study on different datasets, where the inner size parameter is set to 3, 5, and 11. The input time step length is 96, and the stride window size parameter is \{4, 4, 6\}.
Figure 10. Study on attention heads and encoder layers parameters. This figure illustrates the MSE and MAE values for three datasets under different combinations of attention heads and encoder layers. The combinations considered are \{6, 4\}, \{4, 4\}, \{8, 4\}, \{4, 8\}, and \{8, 8\}. Other combinations, where the MSE and MAE values for three datasets under different combinations of attention heads and encoder layers were higher than those mentioned, are not explicitly marked in the figure.
5. Conclusions

In this paper, a multi-scale feature extraction model, MSFformer, based on Transformer models, was proposed to address the issue of the insufficient extraction of long-term dependencies and short-term features in long time series prediction tasks. Specifically, a novel feature convolution method was introduced in the CSCM to obtain coarse-grained information, building a pyramid-shaped data structure through convolution operations with specified strides, extracting time feature information continuously. The Skip-PAM, a cross-stride attention mechanism, was then constructed on this basis, extracting the relevance of adjacent nodes, parent nodes, and child nodes with a specified stride to build an attention mechanism. MSFformer demonstrated superior performance, on average, on three datasets compared to three baseline methods, indicating the effectiveness of our model. Additionally, ablation experiments conducted on the MSFformer model highlighted the significant contribution of the proposed key modules toward performance improvement. Lastly, a study on important hyperparameters revealed different optimal parameters for different datasets.

The limitations of this study include the use of designed feature convolution for constructing coarse-grained data in the MSFformer, which may not efficiently extract features. Exploring efficient methods for constructing coarse-grained data is an area for further research. Additionally, the three layers constructed in the experiments may not be optimal for longer prediction horizons. Further investigation into the relationship between the number of layers and model performance for longer prediction horizons is warranted. Moreover, the temporal granularity of the datasets used in this paper is daily and hourly, and future research could explore the model’s performance on datasets with different levels of granularity and from various domains. For the Electricity dataset, the significant MSE loss indicates that our model’s capability to predict larger errors remains limited. In future work, we will delve deeper into enhancing our model’s predictive accuracy for substantial deviations. In the presentation of our results, we did not detail how the features extracted from the historical sequences influenced the predictions. In our future work, we plan to enhance the interpretability of our model by further exploring and elucidating the specific impact of these features.

Author Contributions: Methodology, R.Z.; validation, R.Z.; data curation, R.Z.; writing—original draft, R.Z.; writing—review & editing, R.Z.; supervision, Y.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Publicly available datasets were analyzed in this study.

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

References


22. Chai, T.; Draxler, R.R. Root mean square error (RMSE) or mean absolute error (MAE)?-Arguments against avoiding RMSE in the literature. *Geosci. Model Dev.* 2014, 7, 1247–1250. [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.