TimeCF: A novel model with adaptive Convolution and Sharpness-Aware Minimization Frequency Domain Loss for long-term time series forecasting

Bin Wang[1], Heming Yang[1] and Jinfang Sheng[1*]
[1]School of Computer Science and Engineering, Central South University, Changsha, China.
Email: {wb_csut, 244712142, jfsheng}@csu.edu.cn
*Corresponding Author: Jinfang Sheng

***Abstract*—Recent studies have shown that by introducing prior knowledge, multi-scale analysis of complex and non-stationary time series in real environments can achieve good results in the field of long-term time series forecasting. However, this kind of models may produce suboptimal prediction results due to the correlation in time series data which in turn affects the generalization performance of the model. To address this challenge, we were inspired by the idea of Sharpness-Aware Minimization and the recently proposed FreDF method and designed a deep learning model TimeCF for long-term time series forecasting based on the decomposition-learning-mixing architecture, combined with adaptive convolution information aggregation module and Sharpness-Aware Minimization Frequency Domain Loss (SAMFre). Specifically, TimeCF first decomposes the original time series into sequences of different scales. Next, the same-sized convolution modules are used to adaptively aggregate information of different scales on sequences of different scales. Then, decomposing each sequence into season and trend parts and the two parts are mixed at different scales through bottom-up and top-down methods respectively. Finally, different scales are aggregated through a Feed-Forward Network. What's more, extensive experimental results on different real-world datasets show that TimeCF has excellent performance in the field of long-term forecasting. Our code is available at https://github.com/iHccob/TimeCF_IJCNN2026.**

***Index Terms*—Time series forecast, Fourier Transform, Convolution, Sharpness-Aware Minimization**

I. Introduction

With the development of information technology in the past decade, time series forecasting, a field of great significance to human life, has been supported by information technology resources such as computing power and algorithms and has played an indispensable role in key areas related to human living standards, such as financial level prediction [1], traffic flow planning [2], [3], weather forecast [4], water treatment [5], energy and power resource allocation [6], [7]. Since the beginning of time series forecasting, there are mainly the following model architectures: Models based on CNN [8], Models based on RNN [9], Models based on Transformer [10] and Models based on MLP [11].

Although researchers have proposed various methods for time series forecasting, it is still challenging to capture features from time series data spanning the past to the future and construct prediction models. This is because time series data exhibits complex and non-stationary characteristics, and noise generated by data acquisition equipment has an adverse effect on the prediction results. In order to solve this problem, the current mainstream research can be divided into two categories: one is based on the Transformer [12] which achieves the fitting of time series through a large number of parameters. However, while these parameters partially tackle complex non-stationary time series, the Transformer still suffers from unsolved issues of overfitting and high memory overhead. Therefore, after fully studying the components of time series, researchers use the prior knowledge of physics and mathematics to decompose time series into simpler components to reduce the difficulty of the prediction process. On this basis, TimeMixer [13] further introduced the idea of multi-scale decomposition, and TimeKAN [14] proposed modeling based on frequencies of different scales. In summary, current researchers hope to simplify the original time series to provide additional prior information for the time series model, thereby improving the prediction accuracy of the time series forecasting model.

It's undeniable that current long-term time series models have achieved good results by Direct Forecast. But time series in the real world are highly autocorrelated. So, when using the Direct Forecast method, autocorrelation in the label sequence isn't fully handled. This might cause the model's results to be somewhat distorted. Fortunately, a method called FreDF [15] has recently been applied to the field of time series forecasting. It uses Fourier or fast Fourier transform to transform the sequence labels from the time domain to the frequency domain without changing the model structure to deal with the autocorrelation in the time series. In addition, we note that an idea called Sharpness-Aware Minimization(SAM) [16] can be combined with FreDF to reduce the sharpness of the loss so that the model has better generalization performance. At the same time, inspired by the idea of attention mechanism in Transformer and the idea of receptive field in CNN field, we found that neighboring and global information can also be used in time series to supplement information in the prediction process. Therefore, we propose to use convolution kernels with the same size at different scales to obtain global information at low-frequency scales and neighboring information at high-frequency scales to achieve the aggregation of global and local

information.

Combining the advantages of the above technologies, we propose a multi-scale hybrid model (TimeCF) based on the decomposition-learning-mixing architecture proposed by TimeMixer to solve the problems of the lack of global and local information, autocorrelation in time series forecasting and generalization performance. In terms of model structure, TimeCF first uses the downsampling method to generate time series at multiple scales. Secondly, through the Past Decomposable Mixing with adaptive Conv (PDMC) module designed by us, we first use the convolution operation of the same convolution kernel size on the sequences of different scales to achieve adaptive information aggregation between different scales. Then, according to prior knowledge, the season and trend parts of the input sequence are decomposed separately. Through our design, PDMC obtains information of different receptive fields according to different input scales and decomposes the sequences of different scales into season and trend parts to achieve more detailed modeling. In the prediction stage, the output prediction layer aggregates the prediction components of different scales to utilize the complementary prediction capabilities between multi-scale sequences to achieve accurate prediction.

In general, our contributions are as follows:

1) We proposed a time series forecasting model TimeCF and introduced the receptive field idea in the CNN field to maximize the use of information aggregation at different scales to supplement global and local information. And based on the ideas of FreDF and SAM, we achieved the decoupling of the autocorrelation in label sequences of the time series and the improvement of the model's generalization ability.
2) Different from previous methods, we use adaptive convolution module to achieve information aggregation of receptive fields of different scales based on sampling results of different scales. And we use the transformation from time domain to frequency domain to solve the challenges brought by direct forecast in time series.
3) TimeCF shows excellent performance in multiple time series forecasting tasks and datasets.

II. Related Work

A. Mainstream Model Architecture

The core of the time series forecasting model is to have efficient and stable pattern extraction and modeling capabilities in different time series, so as to model and predict complex time series. Traditional models such as ARIMA [17] and LSTM [18] can accurately predict time series with simple cycles and trends, but these models are limited by parameters and model structures so the prediction effect for nonlinear and dynamic time series is often unsatisfactory. In recent years, deep learning methods have begun to make great strides in the direction of time series forecasting. For the Transformer, researchers have proposed many methods to apply it to the field of time series forecasting: Autoformer [19] proposed an autocorrelation mechanism to reduce the time complexity of the model to $O(n \cdot lg(n))$. SAMformer [20] solved the instability problem during large model training by using Sharpness-Aware Minimization. Informer [21] used ProbSpare self-attention and Self-attention Distilling to enable it to effectively handle overly long input sequences. iTransformer [22] inverted the time series and then used the Encoder for prediction. Mamba [23] combined the parallelization capability of Transformer and the historical information control capability of RNN, and based on the idea of SSM, it was able to handle the correlation problem between variables at a lower cost. PatchTST [24] regarded the time series as multiple independent time periods of channels, and combined it with Transformer for prediction, achieving good results. And some researchers have found that the use of CNN ideas can better construct the relationship between labels and time steps in time series: MICN [25] introduces the idea of image processing and captures information of different receptive fields through convolution kernels of different sizes. TimesNet [26] performs Fourier transform on the time series and selects its Top-k cycles, then expands each cycle into a two-dimensional image and uses a 2D-kernel convolution kernel for feature extraction. ModernTCN [27] proposes to use large convolution kernels on the time dimension of the time series so that the model can capture dependencies across time and variables at the same time. In addition, researchers have also proposed some models that are not limited to Transformer and CNN: GRU [28] introduces a gating mechanism that allows the model to dynamically adjust the ratio of memory and forgetting according to the current input and previous state, making it more flexible and expressive than traditional RNN. DLinear [29] decomposes time series into seasonal and trend components for separate predictions. FITS [30] uses of basic MLP for prediction in the frequency domain based on DLinear. And SparseTSF [31] obtain information about adjacent time steps through convolution, and then predict future results separately through sparse technology.

Considering the advantages and limitations of the above models, we proposed TimeCF, based on the original idea of scale decomposition, to obtain features of different scales through convolution to achieve multi-scale adaptive information aggregation.

B. Parameters Update

Nowadays, researchers have begun to find that the effects that the models can achieve on the training set are often not achieved on the test set. This is because when the model uses an optimizer to optimize the non-convex loss function on the training set, it may enter a suboptimal or sharp minimum, resulting in insufficient generalization of the model. In response to this situation, researchers have proposed a method called Sharpness-Aware Minimization to update model parameters to improve the generalization ability of deep neural networks: SAM [16] proposed the Sharpness-Aware Minimization method, which first finds the point with the maximum loss in the neighborhood of the current parameter

and then uses gradient descent to update the parameters based on this maximum point, so that the parameters can be moved to a flat area to reduce the sharpness of the loss function. WSAM [32] introduces the concept of weights based on SAM, and adjusts the contribution of different parameters to sharpness according to the importance of the parameters or other indicators, thereby more effectively regularizing. FSAM [33] effectively improves the generalization performance and robustness of the model by improving adversarial perturbations and optimizing the full gradient estimation method.

Considering the generalization degree of the model, we propose a model parameter update module called SAMFre based on the SAM idea to improve the overall generalization ability of the model.

III. TimeCF

A. Overview

The basic definition of time series forecasting is to input the historical data of a multivariate time series $X_{input} \in R^{T \times N}$ and after the model calculation, output the future multivariate output sequence $X_{output} \in R^{F \times N}$, where T represents the look-back length of the historical data defined by the model, F represents the future time length to be predicted and N represents the number of labels in the time series.

In TimeCF, we make independent predictions for each label in the time series. Therefore, the original input can be regarded as $\left\{X_{input^1}, X_{input^2}, \ldots, X_{input^N}\right\}$, where $X_{input^i} \in R^T$ can be regarded as the input instance of TimeCF. The overall structure of TimeCF is shown in Figure 1, which consists of three components: Input Preprocessing layer, PDMC layer, and Output Predicting layer. At the same time, as a module to solve the autocorrelation problem and improve the generalization ability of the model, SAMFre indirectly participates in model training in the stages of calculating model loss and updating model parameters. In summary, the overall process of TimeCF consists of three explicit modules and one implicit module.

B. Input Preprocessing layer

Since we treat the time series $X_{input^i} \in R^T$ in each label as a separate input instance, for each instance X_{input^i},we first use the pooling layer to generate multi-level sequences of different scales $\{X_1, X_2, \ldots, X_k\}$, where $X_i \in R^{\frac{T}{d^{i-1}}}$ $(i \in \{1, \ldots k\})$. The output X_i is the result of $i-1$ times of downsampling of the original input X_{input^i}. X_1 is equal to the original input sequence X_{input^i} and d represents the length of the moving window in the pooling layer. The specific multi-scale sequence generation formula is as follows:

$$X_i = Pool\left(Padding\left(X_{(i-1)}\right)\right) \tag{1}$$

After generating the multi-scale sequences, each sequence will have a time-related mask X_{mask^i}. Each sequence is first normalized by the RevIn normalization layer, and then the mask and sequence are embedded by the Embedding layer. The specific process is as follows:

$$\begin{aligned} X_i = &TemporalEmbedding\left(X_{\text{mask}^i}\right) \\ &+TokenEmbedding\left(RevIN(X_i)\right) \end{aligned} \tag{2}$$

In formula (2), the sequence of each scale is $X_i \in R^{\frac{T}{d^i-1} \times D}$, D is the output dimension of embedding. At this point, the preprocessing part of the input data is completed, and this stage is only performed once during the model training process.

C. Past Decomposable Mixing with adaptive Conv

Recent studies have found that most time series are the fusion of different components of different periods at most scales. Therefore, we propose the PDMC module, which uses long-term and short-term changes to analyze various periodic and non-periodic properties of the entire time series, while obtaining information of different receptive fields at different scales through convolution. Specifically, in the PDMC, we first add global or local information to the sequence through the idea of convolution and adaptation:

$$X_i = \alpha \times ConvBlocks_{[i]}\left(X_i^T\right)^T + X_i \tag{3}$$

The X_i is the output of the input preprocessing layer and the formula for $ConvBlocks_{[i]}\left(X\right)$ is as follows:

$$ConvBlocks_{[i]}\left(X\right) = MultiConv(Norm(X)) \tag{4}$$

Next, we will explain why using the same size of convolution layers can obtain information of different receptive fields. First, the look-back window length selected by this model is 96, which is consistent with the mainstream model, and the number of downsampling is set to 3. So the input of PDMC is $\left\{X_1 \in R^{96 \times D}, X_2 \in R^{48 \times D}, X_3 \in R^{24 \times D}, X_4 \in R^{12 \times D}\right\}$, where 96,48,24,12 are the time windows after downsampling. And in $ConvBlocks_{[i]}\left(X\right)$,we use three layers of convolution and the convolution kernel size of each layer is 3, and the padding is 1. This means that after three convolutions, each time point in $\{X_1, X_2, X_3, X_4\}$ contains information from at least $2 + 2 + 2$, or 6 neighboring time points. From the perspective of PDMC stacking, the number of PDMC stackings is $L(L \geq 2)$. So for X_4, the window of length $6 \times L$ will eventually cover the entire sequence length, which can be considered as obtaining global information. But for X_1,X_2 and X_3,the window of length $6 \times L$ only occupies a part of the sequence length, which can be considered as obtaining local information of different proportions. Meanwhile, convolution layers operate by maintaining the same number of input and output channels, with each output channel containing information from all input channels. This allows us to incorporate information between variables, thereby providing supplementary information for the subsequent prediction process. In summary, TimeCF leverages convolution blocks and PDMC stacking to capture information from various receptive fields at different scales.

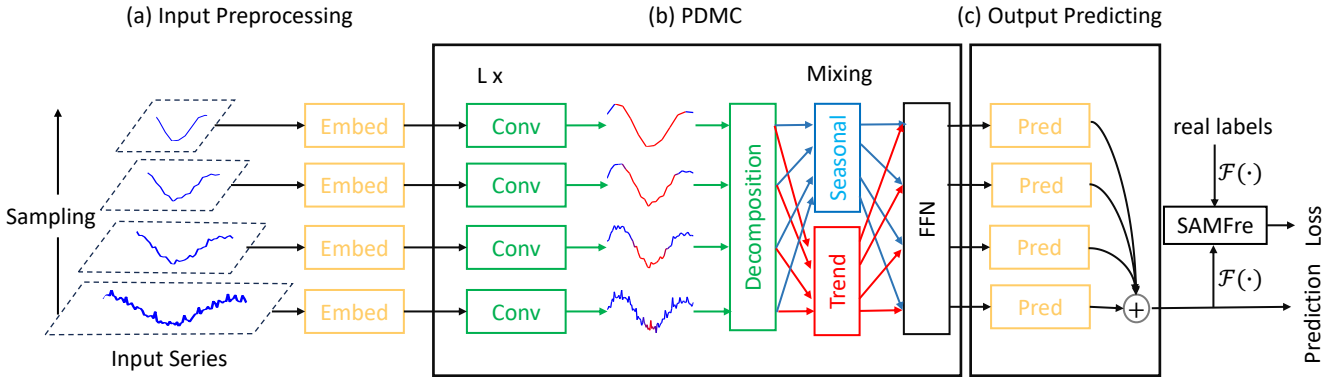

Fig. 1. TimeCF Architecture

Then, we decompose the sequence of each scale into season and trend parts:

$$Season_i, Trend_i = Decomp\left(X_i\right) \tag{5}$$

$Season_i$,$Trend_i$ refer to the season and trend parts decomposed from the i-th scale respectively. We put all the season and trend components into the lists $Season$ and $Trend$ respectively, and based on the idea of TimeMixer, we perform scale fusion on the season and trend respectively:

$$\begin{aligned} Season, Trend = SeasonMix\left(Season\right), \\ TrendMix\left(Trend\right) \end{aligned} \tag{6}$$

The fusion of the season term is a bottom-up sequence fusion and the trend term is a top-down sequence fusion which makes full use of the information inherent in both parts. Finally, PDMC passes the season part, trend part and original sequence through the Feed-Forward Network to achieve the fusion between different components:

$$X_i = X_i + FFN\left(Season_i + Trend_i\right) \tag{7}$$

So far, the PDMC block has finally realized the core tasks of feature extraction and multi-scale mixing process through the adaptive information aggregation by convolution, the decomposition and mixing of the season term and trend term.

D. Output Predicting layer

In the prediction output stage, the output of PDMC we obtain is $\{X_1, X_2, \ldots, X_k\}$, where $X_i \in R^{\frac{T}{d^i-1} \times D}\ (i \in \{1, \ldots k\})$. So if we need to make a prediction for X_i in the time dimension, we need to change the dimension of X_i at least twice. Specifically, first align the time dimension of X_i with the predicted future length according to different scales. Then adjust the dimension of the sequence so that the model vector dimension D can be reduced back to the initial value:

$$X_i = Linear_2\left(Linear_1\left(X_i^T\right)^T\right) \tag{8}$$

The input dimension of the $Linear_1$ is $\frac{T}{d^i-1}$, and the output dimension is the prediction length F. As a result, time series of different scales generate predictions of corresponding time lengths. Then, the input dimension of the $Linear_2$ is D and the output dimension 1. This is to make the sequence dimension match the target output dimension, or let $X_i \in R^F$.

It is not difficult to see that each scale sequence eventually generates a prediction sequence. Then we sum all the prediction sequences and use the RevIN layer of the preprocessing layer to perform inverse normalization:

$$X_O = iRevIN\left(\sum X_i\right) \tag{9}$$

At this point, the prediction of a single label is completed, and the sequences of different labels are finally fused together to generate the result.

E. Sharpness-Aware Minimization Frequency Domain Loss

The loss function of traditional time series forecasting models is usually Mean Square Error (MSE) loss, which has shown its superiority in the training process of a large number of time series forecasting models. However, FreDF's researchers have noticed that the MSE loss hardly takes into account the autocorrelation of the time series in the model using the direct forecast. Therefore, it is not the best choice to calculate the loss by MSE in the training process of the time series forecasting model using the direct forecast method. But according to the idea of Fourier transform, if labels are projected into the frequency domain, unrelated features can be obtained in the frequency domain so that the model based on the idea can obtain better results than the traditional MSE loss when calculating the loss. At the same time, we noticed that the overall generalization performance of the model can be improved by adjusting the sharpness of the loss through the SAM method. Based on these two ideas, the TimeCF we proposed introduces the SAMFre module to decouple the autocorrelation in label sequence and improve

the generalization ability. Specifically, SAMFre projects the model's prediction results and the actual label values into the frequency domain through Fourier transform, then calculates the loss using the L1 norm, and finally obtains the complete loss function by weighted summation with the original loss (MSE):

$$loss = \alpha \times |FFT\,(pred) - FFT\,(real)|_1 + (1-\alpha) \times MSE \tag{10}$$

Here, α is the weight of the frequency loss. Considering the subsequent improvement in the model's generalization brought by SAM, we don't adopt the scheme of setting α to 1 in FreDF. Instead, we set α to 0.85, thereby achieving a better balance between the model's accuracy and generalization.

After calculating the loss, the model uses basic optimization methods to optimize the model parameters before the number of updates reaches the threshold. When the number of updates reaches the threshold, the model uses the SAM method to calculate the point with the largest loss in the neighborhood of the current parameter, and then performs gradient backpropagation based on this point to achieve parameter update:

$$\hat{\epsilon}\,(w) \;=\; \rho \frac{\nabla_w Loss}{\|\nabla_w Loss\|_2} \tag{11}$$

In formula (11), w is the parameters of model. $\nabla_w Loss$ and $\|\nabla_w Loss\|_2$ mean the gradient of the parameters and the L2 norm of the gradient. So $\hat{\epsilon}\,(w)$ means a disturbance whose direction is controlled by a constant ρ.

$$g \;=\; \nabla_w Loss|_{w+\hat{\epsilon}(w)} \tag{12}$$

$$w \;=\; w \;-\; \eta \;\cdot\; g \tag{13}$$

In formula(12) and formula(13), g is the gradient of the disturbed parameters in $w + \hat{\epsilon}\,(w)$. And finally w would be updated by g and a constant η .

So far, we have optimized the model parameter update part through SAMFre so that the model can better deal with the autocorrelation problem in different sequences and improve the generalization ability of the model.

IV. Results

A. Experiment setting

Experimental datasets: In order to verify the prediction accuracy of our model on time series generated in real environments, we selected six commonly used real-world datasets: Weather, ETTh1, ETTh2, ETTm1, ETTm2 and Electricity [19], [21] and conducted sufficient experiments on these six datasets to verify the performance of our model in long-term forecasting.

Benchmark models: Based on timeliness, innovation and prediction effect, we selected 8 time series forecasting models which are widely acclaimed in the field of time series forecasting as our baselines, including: (1) TimeKAN [14] (2) SDE [34] (3) TimeMixer [13] (4) SparseTSF [31] (5) FreTS [35] (6) PatchTST [24] (7) TimesNet [26] (8) DLinear [29]

Experimental environment and related indicators: All experiments were implemented based on the PyTorch framework and executed on a single NVIDIA 3090 24GB GPU. At the same time, in order to ensure fair competition among the models, we set the look-back window, prediction length, and evaluation index to 96, {96, 192, 336, 720}, Mean Square Error (MSE) and Mean Absolute Error (MAE) respectively. What's more, the benchmark model is tested using the scripts provided in the original code, while the test of the TimeCF model we proposed sets different training rounds and early stopping thresholds according to the size of different data sets to improve test efficiency.

B. Experiment results

All experiments are conducted with a fixed random seed for reproducibility. All results in experiments are obtained after local experiments(except for FreTS whose results are obtained from the original paper) and all results are shown in Table 1. We define that the lower the values of MSE and MAE, the better the model prediction effect. It is not difficult to see from Table 1 that the TimeCF we proposed has shown good performance on most datasets. Even if it does not achieve the optimal prediction effect in some datasets, the prediction accuracy of TimeCF is not much different from the results achieved by the optimal model. The average values of MSE and MAE decreased by 2.2% and 1.9% compared with the suboptimal model. And if we look at the number of times the optimal prediction is obtained, TimeCF is far ahead of all models that appeared in the experiments. This proves that TimeCF has accurate and general prediction capabilities on most natural time series data.

C. Ablation experiment

To demonstrate the necessity of our design and addition of modules, we used three forms of TimeCF models in the ablation experiment to compare with our baseline model TimeMixer: (1) TimeCF without the SAMFre module (2) TimeCF without the convolution part and (3) the complete TimeCF. As shown in Table 2, TimeCF without complete modules has a certain improvement over the baseline model in the experiment, but the improvement is not significant. This shows that both the mitigation of the autocorrelation in label sequence and the enhancement of generalization ability based on SAMFre and the adaptive information aggregation between different scales based on convolution can only enhance the partial information extraction and prediction capabilities of the baseline model TimeMixer to a certain extent. However, the good performance of the full TimeCF suggests that the information obtained by convolution at different scales and receptive fields may contain some information with autocorrelation as well as reinforcement of seasonal and trend term differences. By using SAMFre, the autocorrelation within this part of the information can be properly decoupled, which is reflected in the results, which surpass the baseline model in

TABLE I

PERFORMANCE COMPARISON OF DIFFERENT TIME SERIES FORECASTING MODELS ON BENCHMARK DATASETS. THE BEST RESULTS ARE SHOWN IN **BOLD RED** AND THE SECOND BEST RESULTS ARE SHOWN IN **BOLD BLACK**.

Models		TimeCF Ours		TimeKAN 2025		SDE 2025		TimeMixer 2024		SparseTSF 2024		FreTS 2024		PatchTST 2023		TimesNet 2023		DLinear 2023	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETT h1	96	**0.359**	**0.391**	**0.367**	0.394	0.387	0.402	0.381	0.398	0.385	**0.391**	0.395	0.407	0.376	0.397	0.389	0.411	0.396	0.410
	192	**0.401**	**0.419**	**0.414**	**0.419**	0.444	0.433	0.441	0.430	0.434	0.420	0.490	0.477	0.426	0.432	0.439	0.441	0.445	0.440
	336	**0.440**	**0.436**	**0.445**	**0.434**	0.492	0.457	0.500	0.459	0.476	0.439	0.510	0.480	0.469	0.457	0.493	0.470	0.487	0.465
	720	0.466	**0.462**	**0.451**	0.463	0.504	0.485	0.552	0.507	**0.461**	**0.454**	0.568	0.538	0.518	0.504	0.516	0.494	0.512	0.510
	avg	**0.417**	**0.427**	**0.419**	0.428	0.457	0.444	0.468	0.449	0.439	**0.426**	0.490	0.475	0.447	0.447	0.459	0.454	0.460	0.456
ETT h2	96	**0.282**	**0.333**	0.291	0.340	0.296	0.344	**0.286**	**0.339**	0.302	0.346	0.332	0.387	0.308	0.359	0.337	0.370	0.341	0.395
	192	**0.372**	**0.389**	**0.374**	**0.391**	0.382	0.395	0.391	0.404	0.384	0.395	0.451	0.457	0.380	0.406	0.404	0.414	0.481	0.479
	336	**0.410**	**0.422**	0.423	0.434	0.429	0.434	0.421	0.432	0.421	**0.427**	0.466	0.473	**0.412**	0.429	0.455	0.452	0.592	0.542
	720	**0.416**	**0.436**	0.462	0.461	0.436	0.445	0.468	0.468	**0.420**	**0.437**	0.485	0.471	0.435	0.456	0.434	0.448	0.840	0.661
	avg	**0.370**	**0.395**	0.387	0.406	0.386	0.404	0.391	0.411	**0.382**	**0.401**	0.433	0.447	0.384	0.412	0.407	0.421	0.564	0.519
ETT m1	96	**0.307**	**0.345**	**0.321**	**0.361**	0.322	0.364	0.327	0.364	0.356	0.375	0.337	0.374	0.323	0.364	0.333	0.375	0.345	0.373
	192	**0.353**	**0.372**	**0.356**	**0.382**	0.361	0.386	0.367	0.386	0.394	0.392	0.382	0.398	0.371	0.391	0.407	0.413	0.381	0.391
	336	**0.377**	**0.395**	**0.381**	**0.400**	0.401	0.414	0.393	0.403	0.425	0.413	0.420	0.423	0.398	0.408	0.413	0.421	0.415	0.415
	720	**0.441**	**0.430**	**0.451**	**0.437**	0.453	0.443	0.451	0.442	0.487	0.448	0.490	0.471	0.457	0.444	0.503	0.467	0.472	0.450
	avg	**0.370**	**0.386**	**0.377**	**0.395**	0.384	0.402	0.384	0.399	0.415	0.407	0.407	0.416	0.387	0.402	0.414	0.419	0.403	0.407
ETT m2	96	**0.169**	**0.252**	0.175	**0.257**	0.177	0.264	**0.174**	0.257	0.184	0.267	0.186	0.275	0.184	0.267	0.189	0.266	0.193	0.292
	192	**0.238**	**0.299**	0.239	**0.299**	0.248	0.312	**0.236**	0.299	0.248	0.305	0.259	0.323	0.246	0.304	0.252	0.307	0.284	0.361
	336	**0.296**	**0.335**	**0.301**	0.340	0.314	0.353	0.301	**0.339**	0.307	0.342	0.349	0.386	0.311	0.348	0.321	0.349	0.384	0.429
	720	**0.400**	**0.393**	**0.398**	**0.398**	0.419	0.415	0.400	0.400	0.407	0.398	0.559	0.511	0.418	0.414	0.418	0.404	0.556	0.523
	avg	**0.276**	**0.320**	**0.278**	**0.323**	0.289	0.336	0.278	0.324	0.287	0.328	0.338	0.373	0.290	0.333	0.295	0.331	0.354	0.401
Weather	96	**0.162**	**0.204**	0.162	**0.208**	0.165	0.214	**0.161**	0.208	0.197	0.236	0.171	0.227	0.175	0.217	0.168	0.219	0.196	0.256
	192	0.209	**0.249**	**0.207**	**0.249**	0.214	0.255	**0.207**	0.251	0.243	0.273	0.218	0.280	0.220	0.255	0.225	0.265	0.238	0.299
	336	0.266	**0.293**	**0.263**	**0.290**	0.273	0.298	**0.264**	0.293	0.292	0.308	0.265	0.317	0.279	0.297	0.281	0.303	0.281	0.330
	720	0.345	**0.343**	**0.338**	**0.340**	0.354	0.352	0.345	0.345	0.368	0.357	**0.326**	0.351	0.356	0.348	0.359	0.354	0.345	0.381
	avg	0.246	**0.272**	**0.242**	**0.271**	0.252	0.280	**0.244**	0.274	0.275	0.293	0.245	0.293	0.257	0.279	0.258	0.285	0.265	0.316
ECL	96	**0.153**	**0.245**	0.174	0.266	**0.148**	**0.246**	0.156	0.247	0.209	0.280	0.171	0.260	0.180	0.272	0.168	0.271	0.210	0.301
	192	**0.166**	**0.256**	0.182	0.272	**0.162**	**0.258**	0.170	0.260	0.205	0.281	0.177	0.268	0.187	0.279	0.187	0.289	0.210	0.304
	336	**0.183**	**0.274**	0.196	0.286	**0.177**	**0.274**	0.187	0.278	0.218	0.295	0.190	0.284	0.204	0.295	0.201	0.302	0.223	0.319
	720	**0.221**	**0.305**	0.236	0.320	**0.208**	**0.304**	0.227	0.312	0.260	0.327	0.228	0.316	0.245	0.328	0.229	0.324	0.257	0.349
	avg	**0.181**	**0.270**	0.197	0.286	**0.174**	**0.270**	0.185	0.274	0.223	0.296	0.191	0.282	0.204	0.294	0.196	0.297	0.225	0.318
Total AVG		**0.310**	**0.345**	**0.317**	**0.352**	0.324	0.356	0.325	0.355	0.337	0.359	0.351	0.381	0.328	0.361	0.338	0.368	0.379	0.403
1st Times		**18**	**24**	4	**4**	**5**	1	2	0	0	2	0	0	0	0	0	0	0	0

TABLE II

ABLATION STUDY OF TIMECF. THE BEST RESULTS ARE SHOWN IN **BOLD RED**

Model	ETT h1		ETT h2		ECL	
	MSE	MAE	MSE	MAE	MSE	MAE
TimeMixer	0.469	0.449	0.392	0.411	0.185	0.274
TimeCF w/o SAMFre	0.466	0.452	0.392	0.417	0.182	0.273
TimeCF w/o CONV	0.430	0.425	0.372	0.396	0.185	0.272
TimeCF (ours)	**0.417**	**0.427**	**0.371**	**0.396**	**0.181**	**0.270**

terms of evaluation metrics. Finally, it is proved that the adaptive information aggregation module based on convolution and the SAMFre module proposed by us are both indispensable parts of the TimeCF model.

D. Varying look-back window

Based on empirical observations, the length of the look-back window is generally proportional to the amount of historical context it encompasses. However, only time series forecasting models with powerful information extraction capabilities can fully leverage this historical context. As a deep learning model based on a decomposition-learning-mixing architecture, TimeCF possesses a robust ability to capture and utilize historical context, allowing it to achieve more accurate prediction with longer look-back windows. To verify this, we designed several experiments with varying look-back window lengths and prediction horizons: Sequence Length $= \{48, 72, 96, 192\}$ and Prediction Length $= \{96, 192, 336, 720\}$ on the same dataset. As shown in Figure 2(a) and Figure 2(b), the results clearly demonstrate a positive correlation between the forecasting accuracy of TimeCF and the length

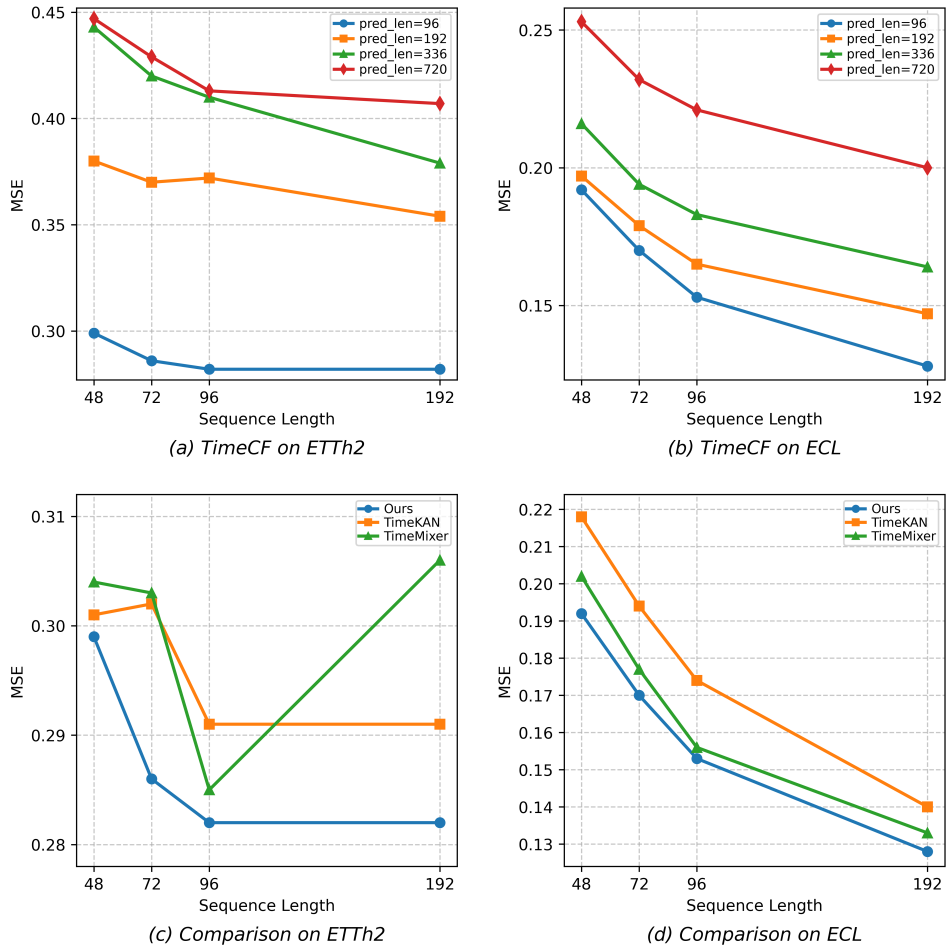

Fig. 2. Experiments of the information extraction performance by extending the sequence length on datasets(a) ETTh2 and (b) ECL. Comparison of forecasting performance between ours and baseline models by varying look-back windows on datasets(c) ETTh2 and (d) ECL.

of the look-back window. To further investigate, we set up a comparative experiment by fixing the prediction length to 96 and varying the look-back window length to evaluate the information extraction performance of TimeCF and baseline models. The results in Figure 2(c) and Figure 2(d) indicate that as the look-back window length increases, TimeCF's forecasting accuracy consistently improves and its information extraction performance significantly outperforms that of the baseline models. These findings strongly validate the powerful ability of TimeCF to extract and utilize historical context.

V. Conclusion

In our paper, we proposed a time series forecasting model TimeCF based on the decomposition-learning-mixing architecture to achieve high-precision time series forecasting. With the support of PDMC, TimeCF can utilize the information of different receptive fields of sequences of different scales, learn and mix the season and trend parts separately and finally combine SAMFre to decouple the autocorrelation in label sequence and reduce the sharpness of the loss function. The performance of our model on real-world datasets also proves that TimeCF can cope with time series forecasting tasks in the real world with good prediction performance.

VI. Acknowledgment

This research was supported by the Key Research and Development Program of Hunan Province(Grant No.2023SK2038).

References

[1] G. Sonkavde, D. S. Dharrao, A. M. Bongale, S. T. Deokate, D. Doreswamy, and S. K. Bhat, "Forecasting stock market prices using machine learning and deep learning models: A systematic review, performance analysis and discussion of implications," *International Journal of Financial Studies*, vol. 11, no. 3, p. 94, 2023, publisher: MDPI.

[2] G. Huo, Y. Zhang, B. Wang, J. Gao, Y. Hu, and B. Yin, "Hierarchical spatio–temporal graph convolutional networks and transformer network for traffic flow forecasting," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 4, pp. 3855–3867, 2023, publisher: IEEE.

[3] X. Huang, B. Zhang, S. Feng, Y. Ye, and X. Li, "Interpretable local flow attention for multi-step traffic flow prediction," *Neural networks*, vol. 161, pp. 25–38, 2023, publisher: Elsevier.

[4] K. Bi, L. Xie, H. Zhang, X. Chen, X. Gu, and Q. Tian, "Accurate medium-range global weather forecasting with 3D neural networks," *Nature*, vol. 619, no. 7970, pp. 533–538, 2023, publisher: Nature Publishing Group.

[5] H. A. Afan, W. H. M. W. Mohtar, F. Khaleel, A. H. Kamel, S. S. Mansoor, R. Alsultani, A. N. Ahmed, M. Sherif, and A. El-Shafie, "Data-driven water quality prediction for wastewater treatment plants," *Heliyon*, vol. 10, no. 18, 2024, publisher: Elsevier.

[6] L. Yin, X. Cao, and D. Liu, "Weighted fully-connected regression networks for one-day-ahead hourly photovoltaic power forecasting," *Applied Energy*, vol. 332, p. 120527, 2023, publisher: Elsevier.

[7] G. Alkhayat and R. Mehmood, "A review and taxonomy of wind and solar energy forecasting methods based on deep learning," *Energy and AI*, vol. 4, p. 100060, 2021, publisher: Elsevier.

[8] L. Li, C. Jian, F. Wan, D. Geng, Z. Fang, L. Chen, Y. Gao, W. Jiang, and J. Zhu, "LagCNN: A Fast yet Effective Model for Multivariate Long-term Time Series Forecasting," in *CIKM*, 2024, pp. 1235–1244.

[9] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, "DeepAR: Probabilistic forecasting with autoregressive recurrent networks," *International journal of forecasting*, vol. 36, no. 3, pp. 1181–1191, 2020, publisher: Elsevier.

[10] X. Liang, E. Yang, C. Deng, and Y. Yang, "CrossFormer: Cross-Modal Representation Learning via Heterogeneous Graph Transformer," *ACM Trans. Multim. Comput. Commun. Appl.*, vol. 20, no. 12, pp. 380:1–380:21, Dec. 2024.

[11] C. Challu, K. G. Olivares, B. N. Oreshkin, F. G. Ramírez, M. M. Canseco, and A. Dubrawski, "NHITS: Neural Hierarchical Interpolation for Time Series Forecasting," in *AAAI*, 2023, pp. 6989–6997.

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, vol. 30, 2017.

[13] S. Wang, H. Wu, X. Shi, T. Hu, H. Luo, L. Ma, J. Y. Zhang, and J. Zhou, "TimeMixer: Decomposable Multiscale Mixing for Time Series Forecasting," in *ICLR*, 2024, pp. 4166–4192.

[14] S. Huang, Z. Zhao, C. Li, and L. Bai, "TimeKAN: KAN-based Frequency Decomposition Learning Architecture for Long-term Time Series Forecasting," in *The Thirteenth International Conference on Learning Representations*, 2025, pp. 93 540–93 555.

[15] H. Wang, L. Pan, Z. Chen, D. Yang, S. Zhang, Y. Yang, X. Liu, H. Li, and D. Tao, "Fredf: Learning to forecast in frequency domain," in *The Thirteenth International Conference on Learning Representations*, 2024, pp. 7329–7358.

[16] P. Foret, A. Kleiner, H. Mobahi, and B. Neyshabur, "Sharpness-aware Minimization for Efficiently Improving Generalization," in *The Ninth International Conference on Learning Representations*, 2021, pp. 2315–2333.

[17] G. P. Zhang, "Time series forecasting using a hybrid ARIMA and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, publisher: Elsevier.

[18] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997, publisher: MIT press.

[19] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting," in *Advances in Neural Information Processing Systems*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34. Curran Associates, Inc., 2021, pp. 22 419–22 430.

[20] R. Ilbert, A. Odonnat, V. Feofanov, A. Virmaux, G. Paolo, T. Palpanas, and I. Redko, "SAMformer: Unlocking the Potential of Transformers in Time Series Forecasting with Sharpness-Aware Minimization and Channel-Wise Attention," in *Proceedings of the 41st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 235. PMLR, 2024, pp. 20 924–20 954.

[21] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, 2021, pp. 11 106–11 115, issue: 12.

[22] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, and M. Long, "iTransformer: Inverted Transformers Are Effective for Time Series Forecasting," in *The Twelfth International Conference on Learning Representations*, 2024, pp. 4004–4028.

[23] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," in *First conference on language modeling*, 2024, pp. 1–32.

[24] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A Time Series is Worth 64 Words: Long-term Forecasting with Transformers," in *The Eleventh International Conference on Learning Representations*, 2023, pp. 33 132–33 155.

[25] H. Wang, J. Peng, F. Huang, J. Wang, J. Chen, and Y. Xiao, "MICN: Multi-scale Local and Global Context Modeling for Long-term Series Forecasting," in *The eleventh international conference on learning representations*, 2023, pp. 13 014–13 035.

[26] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis," in *The Eleventh International Conference on Learning Representations*, 2023, pp. 6423–6445.

[27] L. donghao and w. xue, "ModernTCN: A Modern Pure Convolution Structure for General Time Series Analysis," in *The Twelfth International Conference on Learning Representations*, 2024.

[28] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," in *NIPS 2014 Workshop on Deep Learning, December 2014*, 2014.

[29] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?" in *Proceedings of the AAAI conference on artificial intelligence*, vol. 37, 2023, pp. 11 121–11 128, issue: 9.

[30] Z. Xu, A. Zeng, and Q. Xu, "FITS: Modeling Time Series with \10k\ Parameters," in *The Twelfth International Conference on Learning Representations*, 2024, pp. 29 143–29 166.

[31] S. Lin, W. Lin, W. Wu, H. Chen, and J. Yang, "SparseTSF: Modeling Long-term Time Series Forecasting with *1k* Parameters," in *Proceedings of the Forty-first International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 235. PMLR, 2024, pp. 30 211–30 226.

[32] Y. Yue, J. Jiang, Z. Ye, N. Gao, Y. Liu, and K. Zhang, "Sharpness-Aware Minimization Revisited: Weighted Sharpness as a Regularization Term," in *KDD*, 2023, pp. 3185–3194.

[33] T. Li, P. Zhou, Z. He, X. Cheng, and X. Huang, "Friendly Sharpness-Aware Minimization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2024, pp. 5631–5640.

[34] Z. Weng, J. Han, W. Jiang, and H. Liu, "Sde: A simplified and disentangled dependency encoding framework for state space models in time series forecasting," in *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, 2025, pp. 3168–3179.

[35] K. Yi, Q. Zhang, W. Fan, S. Wang, P. Wang, H. He, N. An, D. Lian, L. Cao, and Z. Niu, "Frequency-domain mlps are more effective learners in time series forecasting," *Advances in Neural Information Processing Systems*, vol. 36, pp. 76 656–76 679, 2023.