# UNetPlusTS: Decomposition-Mixing UNet++ Architecture for Long-Term Time Series Forecasting

Xuelin Cheng[1,2], Xince Chen[1(✉)], Botao Wu[1], Xu Zou[1], Haozheng Yang[1], and Runjie Zhao[1]

[1] School of Software Technology, Zhejiang University, Ningbo 310013, China
xincechen@zju.edu.cn
[2] Shanghai Futuroscope Information Technology Co., Ltd., Shanghai 201203, China

**Abstract.** Long-term time series forecasting has widespread applications in real life, including finance, traffic, weather and sensor data analysis. Time series possess seasonal, trend and irregular components. However, current MLP-based mixing models fall short in effectively capturing multi-scale sequential information and achieving comprehensive feature fusion. The sequence information becomes non-stationary, and as the network depth increases, issues with missing or distorted information flow may arise. In our study, we propose a novel architecture called UNetPlusTS that combines the strengths of Mixer models and Linear models to enhance forecasting capabilities. Specifically, we split series into multiple channels, apply seasonal-trend decomposition to each series and process them independently using our meticulously designed UNet ++ architecture named *UNetPlus Mixing Module* (UPM). Combined with our unique sampling strategy, it promotes deep integration of seasonal and trend components, thereby alleviating non-stationarity. Experimental results on multiple real-world datasets show that UNetPlusTS has significantly improved the forecasting accuracy, demonstrating its effectiveness and robustness.

**Keywords:** Long-term time series forecasting · UNet++ · Mixing Model · Decomposition · Deep learning

## 1 Introduction

Long-term time series forecasting (LTSF) stands as a cornerstone in the realm of time series analysis, involving predicting future values within a sequence based on its historical data. It is extensively applied in a variety of real-world industries, such as traffic state prediction [2], financial market analysis [12], climate estimation [10], sensor data analysis [23], etc.

Among various deep learning models, those Convolutional Neural Networks (CNNs) [1, 16] and Transformer [24, 26] based models have shown tremendous potential in capturing the periodicity and features of time series. For CNN models, due to the locality of the convolutiuon kernel, they struggle to capture contextual and global information


D.-S. Huang et al. (Eds.): ICIC 2024, LNAI 14876, pp. 41–52, 2024.
https://doi.org/10.1007/978-981-97-5666-7_4

across sequences, leading to less efficient performance. Owing to the inevitable accumulation of errors, Transformer-based models exhibit suboptimal performance in long-term series forecasting.

Recent studies have shown that Linear models, through modeling separated trends and residuals [20] or applying instance normalization [6], have achieved favorable results. Mixer model alternately applies MLPs in different directions, namely temporal mixing and feature mixing, effectively capturing time patterns and cross-variable information. However, Linear and Mixer model also face challenges: (1) Linear models are unable to capture sequence information across multiple scales of variation. (2) Separated trends have less information integration at different scales, leading to the occurrence of non-stationarity. (3) The Mixer model of deep architecture may have limitations in the effectiveness of information flow when processing deep structures.

In this article, we propose UNetPlusTS to address aforementioned problems. Through the improved UNet++ architecture [27] adapted for time series, named the *UNetPlus Mixing Module* (UPM), dense residual connections are introduced to address the limitations of weakened or distorted information flow. By integrating with the Mixer model, UNetPlusTS facilitates deep fusion of seasonal and trend components across different scales.

Specifically, we adopt a channel-independent approach [14] to split the time series into multiple channels, which are then individually fed into UPM for processing. We apply seasonal-trend decomposition to the time series to decompose the seasonal and trend components of time series. With our novel design, UPM is capable of hierarchically obtaining trend components and temporal information from macro to micro granularity, and it integrates seasonal components of micro granularity upwards to obtain seasonal features of macro granularity. Our contributions are as follows:

1. We propose a novel architecture UNetPlusTS, which combines the strengths of Mixer models and Linear models to enhance forecasting capabilities.
2. We redesign the UNet ++ architecture called UPM to suit time series forecasting through the integration of multi-scale seasonality and trends.
3. Through comprehensive experiments on a wide range of real-world time series datasets, we showcase the effectiveness and robustness of UNetPlusTS.

## 2 Related Works

### 2.1 Linear Long-Term Time Series Forecasting

Recently, Linear models have gained attention in LTSF. DLinear [20] raised the question of whether transformer are effective for long-term time series forecasting. It decomposes the time series into trend and remainder series, and then models these using two linear layers to achieve the forecasting task. LightTS [21] applies an MLP-based structure on top of two subtle down-sampling strategies, including interval sampling and contiguous sampling. TiDE [5] employs an encoder-decoder acrchitecture, which not only utilizes past sequence values (LookBack) but also leverages some covariate information such as static attributes and dynamic covariates. TSMixer [3] conducts temporal and feature mixing on time series to capture temporal information and cross-variable information.

In this paper, drawing inspiration from these existing models, we integrate the Mixer architecture into long-term time series forecasting in this paper. The integration is achieved by combining the encoder-decoder architecture and UNet++ architecture, using up-sample and down-sample to more effectively integrate multi-scale features to build more comprehensive richer representations.

### 2.2 U-Net Series Architectures

U-Net [15] was initially applied in biomedical image segmentation, adopting a symmetric U-shaped network design, mainly comprising down-sampling encoding, up-sampling decoding, and skip connections. In the context of long-term time series forecasting, for shallow layers, the down-sampling factor is small, and the seasonal components possess more detailed features. In deep layers, and the trend components exhibit more macroscopic trend features. When the two are combined, the prediction effect will be better. UNet++ [27] is an improvement over U-Net, it replaced longer skip connections with shorter ones. Therefore, it can integrate features from different levels while enabling overall gradient updates.

## 3 Methodology

### 3.1 Problem Definition

Consider the historical sequence data defined as $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_\mathbf{L}\}$, where $L$ specifies the size of look-back window and $N$ denotes the number of features. Here, each $\mathbf{x_t}$ represents a feature vector with $N$ channels at time step $t$. In the realm of time series forecasting, the primary objective is to accurately predict a sequence of $T$ future observations which are denoted by $\mathbf{Y} = \{\mathbf{x_{L+1}}, \ldots, \mathbf{x_{L+T}}\} \in \mathbb{R}^{T \times N}$. The goal is to train a model $\mathbf{M}$: $\mathbb{R}^{L \times N} \rightarrow \mathbb{R}^{T \times N}$ that forecasts $\widehat{\mathbf{Y}} = \mathbf{M}(\mathbf{X})$ approximating the $\mathbf{Y}$.

### 3.2 Model Architecture

The overview architecture of UNetPlusTS is shown in Fig. 1. Given a multi-variate time series input $\mathbf{X}$ with $N$ dimensions, to disentagle its complex variations, we first split it into $N$ univariate time series $\mathbf{X} \in \mathbb{R}^{\mathbf{L} \times 1}$, these series are projected into deep feature by the embedding layer and ultimately obtain a collection of time series, denote as $\mathcal{X} = \{\mathbf{x}_0, \ldots, \mathbf{x_N}\}$, where $\mathbf{x_N} \in \mathbb{R}^{L \times C}$, $C$ denotes the latent space dimension.

Next, to obtain the initial seasonal component $\mathcal{X}_\mathrm{s}$ (we denote as $\mathcal{S}^{0,0}$here) and trend component $\mathcal{X}_\mathrm{t}$ (we denote as $\mathcal{T}^{0,0}$here), seasonal-trend decomposition [19] is applied to the aforementioned prepared series. For our input series $\mathcal{X} \in \mathbb{R}^{L \times C}$, the process can be formalized:

$$\begin{aligned} \mathcal{T}^{0,0} &= \mathrm{AvgPool}(\mathrm{Padding}(\mathcal{X})) \\ \mathcal{S}^{0,0} &= \mathcal{X} - \mathcal{T}^{0,0} \end{aligned} \tag{1}$$

where $\mathcal{S}^{0,0}, \mathcal{T}^{0,0} \in \mathbb{R}^{L \times D}$denote the initial seasonal and trend component respectively. $AvgPool(\cdot)$ is utilized for moving average and $Padding(\cdot)$ for keeping the

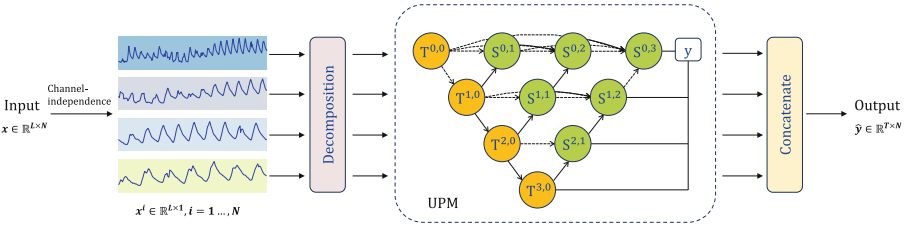


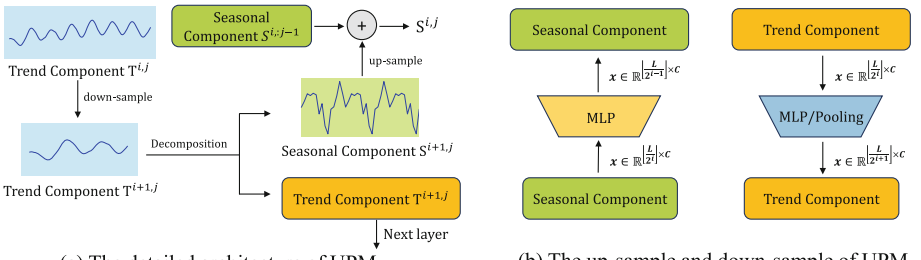

**Fig. 1.** The architecture of UNetPlusTS. Multivariate time series are split into multiple channels. Use seasonal-trend decomposition and then enter the UNetPlusTS backbone.

series length unchanged. For convenience, in the following section, we will use $\mathcal{T}^{i,j}, \mathcal{S}^{i,j} = SeriesDecomp(\mathcal{X})$to represent the aforementioned formulae.

The preprocessed data are then fed into a *UNetPlus Mixing Module* (UPM) to further process the seasonal and trend components, ultimately fuse them to get the final prediction output. The detail of UPM will be described in the next section.

### 3.3 UNetPlus Mixing Module (UPM)

In UNetPlusTS, we introduce a novel time series forecasting network that combines the UNet++ architecture [27] with decomposition and the Mixer architecture. The detail of UNetPlus Mixing Module is shown in Fig. 2.




(a) The detailed architecture of UPM (b) The up-sample and down-sample of UPM

**Fig. 2.** The detail of UNetPlus Mixing Module. (a) The trend component undergoes down-sampling, followed by series decomposition. Seasonal component is used for up-sampling and fusion. (b) Up-sample use temporal MLP, featuring two dense layers with a GELU activation function. Down-sample use either MLP or Pooling methods (Average Pooling or Max Pooling).

Similar to UNet++ architecture, UNetPlusTS also embeds U-Nets of multiple depths in its architecture. In time series analysis, it is noteworthy that the seasonal and trend components possess distinct characteristic, corresponding to short-term and long-term changes or stationary and non-stationary dynamics, respectively [4, 17]. Therefore, we propose *UNetPlus Mixing Module* (UPM) to mix seasonal and trend components in varying layers. This method combines deep-level trend information with shallow-level seasonal information. Short skip connections integrate local information, while long skip connections ensure that global trend information is not lost.

**Network Connectivity** To extract more information from seasonal and trend components at different scales, a hierarchical structure is employed here. For trend components, the finer-grained items contain more noise and unexpected variations [8]. Therefore, guiding the micro-granularity from a coarse-grained trend can better eliminate noise. Subsequently, further decomposition can more effectively separate the trend and seasonal components. Let $\mathcal{T}^{i,0} \in \mathbb{R}^{\lfloor \frac{L}{2^i} \rfloor \times C}$denotes the input where $i$ indexes the down-sampling layer along the encoder, it can be defined as follows:

$$\mathcal{T}^{i,0} = \begin{cases} SeriesDecomp\,(\mathcal{X}_{in}) & i = 0 \\ \mathcal{D}(SeriesDecomp(\mathcal{T}^{i-1,0})) & i \in (0, M) \end{cases} \tag{2}$$

where function $D(\cdot)$ denotes a down-sampling layer and $M$ represents the number of levels. Notably, we focus solely on the trend component $\mathcal{T}^{i,j}$obtained from seasonal-trend composition, disregarding the seasonal component. At $i = 0$, trend component is the initial result decomposed from the input $\mathcal{X}$ mentioned previously.

Contrast to trend component, the fine-grained seasonal component contains more characteristic information, including the periodicity of the coarse-grained seasonal component. Using the fine-grained seasonal component to guide the inference of the macro-scale seasonal component can fuse more features. Thanks to the meticulous design of skip connections in UNet++, allowing the aggregation layer to decide how to fuse the various features carried along the skip connections with the decoder features. In our paper, from a horizontal perspective, each seasonal component node in the decoder is fused with features of all its preceding nodes at the same temporal level. From a vertical perspective, the seasonal component integrates multi-scale features from its preceding node. It can be formalized as:

$$\mathcal{S}^{i,j} = \begin{cases} SeriesDecomp(\mathcal{T}^{i,j}) & j = 0 \\ [\mathcal{S}^{i,k}]_{k=0}^{j-1} + \mathcal{U}(\mathcal{S}^{i+1,j-1}) & j \in (0, M) \end{cases} \tag{3}$$

where function $U(\cdot)$ denotes an up-sampling layer, and $[\cdot]$ denotes the concatenation layer. The output of decoders is defined as:

$$Y_{out,i} = \mathcal{T}^{i,0} + \mathcal{S}^{i,M-i-1} \tag{4}$$

$Y_{out,i}$ is the output of the $i$-level decoder. And the final output of the UPM is defined as $Y_{out} = \frac{1}{N} \sum_{i=0}^{M-1} Y_{out,i} \in \mathbb{R}^{T \times C}$, where $T$ is the prediction length. After concatenation, the prediction output is $\mathbf{Y} \in \mathbb{R}^{T \times N}$.

**Down-Sample and Up-Sample** As shown in Fig. 2(b), a Multi-Layer Perceptron (MLP) containing two dense layers and a GELU activation function is used for up-sampling on the seasonal component in the temporal dimension. After the up-sampling, the seasonal component is restored to the temporal scale of the previous layer. An MLP or pooling method (Average Pooling or Max Pooling) is used for down-sampling on the trend component in the temporal dimension. The comparison of these different down-sampling methods will be discussed in the following section.

# 4 Experiments

## 4.1 Datasets

We evaluate UNetPlusTS on 6 large-scale datasets in our experiments, including ETT datasets (ETTh1, ETTh2, ETTm1, ETTm2) [24], Electricity [19] and Weather [14]. We divided each dataset into the training, validation, and testing subsets according to the chronological order by the ratio of 6:2:2 for ETT dataset and 7:1:2 for other datasets.

## 4.2 Experimental

**Model Configuration and Metrics.** Our experiments are conducted using the PyTorch framework on an Nvidia RTX4090 GPU (24 GB). For long-term forecasting, we adopted a model configuration consistent with those described in related studies [18]: The input sequence length is set to $L = 96$, forecasting sequence lengths are $T \in \{96, 192, 336, 720\}$, the batch size is set to 16. The number of UPM layers $M$ is set to 4. We train 10 epochs and Adam is utilized for optimization. To evaluate the performance of long-term forecasting, we employ mean square error (MSE) and mean absolute error (MAE).

**Baselines.** We compare UNetPlusTS with following 13 SOTA methods from different categories: (1) RNN-based models: LSTM [9] and LSSL [7]. (2) MLP-based models: TiDE [5], LightTS [21], FiLM [25] and DLinear [20]. (3) CNN-based models: TCN [1] and MICN [16]. (4) Transformer-based models: Informer [24], Autoformer [19], FEDformer [26], PatchTST [14] and Crossformer [22].

## 4.3 Main Results

Table 1 presents the long-term forecasting performance measured by MSE and MAE. It can be observed that UNetPlusTS achieves impressive performance. Among the 60 long-term forecasting cases of 6 datasets, we achieved the best results in 52 cases. Compared with other models, PatchTST uses patching mechanism to handle rapid fluctuation [13], potentially lossing locality information. DLinear models trend and seasonal component with simple dense layers, thus failing to effectively capture sequence information. By contrast, UNetPlusTS is capable of capturing global information while effectively integrating seasonal and trend components, resulting in improved performance.

## 4.4 Ablation Study

**Effect of Each Component.** To better verify the effectiveness of channel independence and the UPM module, we conducted detailed ablation experiments. Here, without Channel-Independence (w/o CI), without UPM (w/o UPM), Reversed UPM (Rev UPM) are variants of UNetPlusTS. In the w/o CI configuration, channel-independence is removed from UNetPlusTS. In the w/o UPM configuration, the number of UPM layers is set to $M = 0$. In the Rev UPM configuration, the sampling strategy for the seasonal and

**Table 1.** Comparing UNetPlusTS with SOTA benchmarks on test set of datasets. The best results are in red and the second best are underline.

| Model | | UNetPlusTS (Ours) | | PatchTST 2023a | | Crossformer 2023 | | MICN 2023 | | FiLM 2022a | | DLinear 2022b | | FEDformer 2021 | | Autoformer 2021 | | Informer 2021 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Metric | | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| Weather | 96 | 0.180 | 0.220 | 0.186 | 0.227 | 0.195 | 0.271 | 0.195 | 0.236 | 0.195 | 0.236 | 0.195 | 0.252 | 0.217 | 0.296 | 0.266 | 0.336 | 0.300 | 0.384 |
| | 192 | 0.227 | 0.259 | 0.234 | 0.265 | 0.209 | 0.277 | 0.239 | 0.271 | 0.239 | 0.271 | 0.237 | 0.295 | 0.276 | 0.336 | 0.307 | 0.367 | 0.598 | 0.544 |
| | 336 | 0.281 | 0.298 | 0.284 | 0.301 | 0.373 | 0.332 | 0.289 | 0.306 | 0.289 | 0.306 | 0.282 | 0.331 | 0.339 | 0.380 | 0.359 | 0.395 | 0.578 | 0.523 |
| | 720 | 0.357 | 0.347 | 0.356 | 0.349 | 0.379 | 0.401 | 0.361 | 0.351 | 0.361 | 0.351 | 0.345 | 0.382 | 0.403 | 0.428 | 0.419 | 0.428 | 1.059 | 0.741 |
| | Avg | 0.261 | 0.281 | 0.265 | 0.285 | 0.264 | 0.320 | 0.268 | 0.321 | 0.271 | 0.291 | 0.265 | 0.315 | 0.309 | 0.360 | 0.338 | 0.382 | 0.634 | 0.548 |
| Electricity | 96 | 0.185 | 0.264 | 0.190 | 0.296 | 0.219 | 0.314 | 0.180 | 0.293 | 0.198 | 0.274 | 0.210 | 0.193 | 0.193 | 0.308 | 0.201 | 0.317 | 0.274 | 0.368 |
| | 192 | 0.188 | 0.270 | 0.199 | 0.304 | 0.231 | 0.322 | 0.189 | 0.302 | 0.198 | 0.278 | 0.210 | 0.305 | 0.201 | 0.315 | 0.222 | 0.334 | 0.296 | 0.386 |
| | 336 | 0.204 | 0.285 | 0.217 | 0.319 | 0.246 | 0.337 | 0.198 | 0.312 | 0.217 | 0.300 | 0.223 | 0.319 | 0.214 | 0.303 | 0.231 | 0.443 | 0.300 | 0.394 |
| | 720 | 0.245 | 0.319 | 0.258 | 0.352 | 0.280 | 0.363 | 0.217 | 0.330 | 0.278 | 0.356 | 0.258 | 0.350 | 0.246 | 0.355 | 0.254 | 0.361 | 0.373 | 0.439 |
| | Avg | 0.205 | 0.285 | 0.216 | 0.318 | 0.244 | 0.334 | 0.196 | 0.309 | 0.223 | 0.302 | 0.225 | 0.319 | 0.214 | 0.327 | 0.227 | 0.338 | 0.311 | 0.397 |
| ETTh1 | 96 | 0.379 | 0.392 | 0.460 | 0.447 | 0.423 | 0.448 | 0.426 | 0.446 | 0.438 | 0.433 | 0.397 | 0.412 | 0.395 | 0.424 | 0.449 | 0.459 | 0.865 | 0.713 |
| | 192 | 0.432 | 0.425 | 0.512 | 0.477 | 0.471 | 0.474 | 0.454 | 0.464 | 0.493 | 0.466 | 0.446 | 0.441 | 0.469 | 0.470 | 0.500 | 0.482 | 1.008 | 0.792 |
| | 336 | 0.478 | 0.446 | 0.546 | 0.496 | 0.570 | 0.546 | 0.493 | 0.487 | 0.547 | 0.495 | 0.489 | 0.467 | 0.530 | 0.499 | 0.521 | 0.496 | 1.107 | 0.809 |
| | 720 | 0.494 | 0.479 | 0.544 | 0.517 | 0.653 | 0.621 | 0.526 | 0.526 | 0.586 | 0.538 | 0.513 | 0.510 | 0.598 | 0.544 | 0.514 | 0.512 | 1.181 | 0.865 |
| | Avg | 0.445 | 0.435 | 0.516 | 0.484 | 0.529 | 0.522 | 0.475 | 0.480 | 0.516 | 0.483 | 0.461 | 0.457 | 0.498 | 0.484 | 0.496 | 0.487 | 1.040 | 0.795 |
| ETTh2 | 96 | 0.286 | 0.337 | 0.308 | 0.355 | 0.745 | 0.584 | 0.372 | 0.424 | 0.322 | 0.364 | 0.340 | 0.394 | 0.358 | 0.397 | 0.346 | 0.388 | 3.755 | 1.525 |
| | 192 | 0.369 | 0.389 | 0.393 | 0.405 | 0.877 | 0.656 | 0.492 | 0.492 | 0.404 | 0.414 | 0.482 | 0.479 | 0.429 | 0.439 | 0.456 | 0.452 | 5.602 | 1.931 |
| | 336 | 0.411 | 0.425 | 0.427 | 0.436 | 1.043 | 0.731 | 0.607 | 0.555 | 0.435 | 0.445 | 0.591 | 0.541 | 0.496 | 0.487 | 0.482 | 0.486 | 4.721 | 1.835 |
| | 720 | 0.426 | 0.441 | 0.436 | 0.450 | 1.104 | 0.763 | 0.824 | 0.655 | 0.447 | 0.458 | 0.839 | 0.661 | 0.463 | 0.474 | 0.515 | 0.511 | 3.647 | 1.625 |
| | Avg | 0.375 | 0.400 | 0.391 | 0.411 | 0.942 | 0.684 | 0.574 | 0.531 | 0.402 | 0.420 | 0.563 | 0.519 | 0.437 | 0.449 | 0.450 | 0.459 | 4.431 | 1.729 |
| ETTm1 | 96 | 0.326 | 0.362 | 0.352 | 0.374 | 0.404 | 0.426 | 0.365 | 0.387 | 0.353 | 0.370 | 0.346 | 0.374 | 0.379 | 0.419 | 0.505 | 0.475 | 0.672 | 0.571 |
| | 192 | 0.369 | 0.380 | 0.390 | 0.393 | 0.450 | 0.451 | 0.403 | 0.408 | 0.389 | 0.387 | 0.382 | 0.391 | 0.426 | 0.441 | 0.553 | 0.496 | 0.795 | 0.669 |
| | 336 | 0.402 | 0.404 | 0.421 | 0.414 | 0.532 | 0.515 | 0.436 | 0.431 | 0.421 | 0.408 | 0.415 | 0.415 | 0.445 | 0.459 | 0.621 | 0.537 | 1.212 | 0.871 |
| | 720 | 0.466 | 0.441 | 0.462 | 0.449 | 0.666 | 0.589 | 0.489 | 0.462 | 0.481 | 0.441 | 0.473 | 0.451 | 0.543 | 0.490 | 0.671 | 0.561 | 1.166 | 0.823 |
| | Avg | 0.390 | 0.396 | 0.406 | 0.407 | 0.513 | 0.495 | 0.423 | 0.422 | 0.411 | 0.402 | 0.404 | 0.408 | 0.448 | 0.452 | 0.588 | 0.517 | 0.961 | 0.734 |
| ETTm2 | 96 | 0.175 | 0.256 | 0.183 | 0.267 | 0.287 | 0.366 | 0.197 | 0.296 | 0.183 | 0.266 | 0.193 | 0.293 | 0.203 | 0.287 | 0.255 | 0.339 | 0.365 | 0.453 |
| | 192 | 0.241 | 0.299 | 0.255 | 0.314 | 0.414 | 0.492 | 0.284 | 0.361 | 0.248 | 0.305 | 0.284 | 0.361 | 0.269 | 0.328 | 0.281 | 0.340 | 0.533 | 0.563 |
| | 336 | 0.299 | 0.338 | 0.309 | 0.347 | 0.597 | 0.542 | 0.381 | 0.429 | 0.309 | 0.343 | 0.382 | 0.429 | 0.382 | 0.429 | 0.339 | 0.372 | 1.363 | 0.887 |
| | 720 | 0.396 | 0.394 | 0.412 | 0.404 | 1.730 | 1.042 | 0.549 | 0.522 | 0.410 | 0.400 | 0.473 | 0.451 | 0.558 | 0.525 | 0.433 | 0.432 | 3.379 | 1.338 |
| | Avg | 0.277 | 0.321 | 0.290 | 0.334 | 0.757 | 0.610 | 0.353 | 0.402 | 0.287 | 0.329 | 0.354 | 0.402 | 0.305 | 0.349 | 0.327 | 0.371 | 1.410 | 0.810 |
| 1st Count | | 52 | | 1 | | 1 | | 4 | | 0 | | 1 | | 1 | | 0 | | 0 | |

trend components is reversed, that is, down-sampling is applied to the seasonal components, and up-sampling is applied to the trend components. We set the batch size as 16, time series input sequence length as 96, and prediction lengths as {96, 192, 336, 720}. The results are shown in Table 2. All metrics of these three variants are significantly worse than UNetPlusTS. The complete version of UNetPlusTS achieves the best results, which also proves that removing any component will affect the final output.

**Effect of Down-Sample Method.** In UNetPlusTS, we employ either MLP or Pooling method for the down-sampling of the trend component. Here, we compare the effects of four down-sampling methods: MLP, Average Pooling, Max Pooling, and Conv1d on model's performance on Weather and ETTm1 datasets. Input sequence length is 96 and prediction lengths are {96, 192, 336, 720}. All other settings of the experiment were set to default. We repeated the experiment three times and took the average. The results are shown in Table 3. It can be observed that different down-sampling methods have a minimal impact on the results, but MLP can slightly improve the performance. The reason for this might be that, when using MLP for down-sampling, global information can be captured through the fully connected layers, more effectively capturing and retaining

**Table 2.** Results of ablation study on Weather and ETTm1.

| Model | | UNetPlusTS | | w/o CI | | w/o UPM | | Rev UPM | |
|---|---|---|---|---|---|---|---|---|---|
| Metric | | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| Weather | 96 | 0.180 | 0.220 | 0.186 | 0.228 | 0.196 | 0.235 | 0.196 | 0.236 |
| | 192 | 0.227 | 0.259 | 0.234 | 0.266 | 0.239 | 0.270 | 0.241 | 0.273 |
| | 336 | 0.281 | 0.298 | 0.288 | 0.304 | 0.291 | 0.307 | 0.294 | 0.308 |
| | 720 | 0.357 | 0.347 | 0.361 | 0.352 | 0.363 | 0.353 | 0.367 | 0.356 |
| | Avg | 0.261 | 0.281 | 0.267 | 0.288 | 0.272 | 0.291 | 0.275 | 0.293 |
| ETTm1 | 96 | 0.326 | 0.362 | 0.334 | 0.371 | 0.353 | 0.374 | 0.351 | 0.379 |
| | 192 | 0.369 | 0.380 | 0.374 | 0.389 | 0.391 | 0.391 | 0.394 | 0.403 |
| | 336 | 0.402 | 0.404 | 0.405 | 0.409 | 0.422 | 0.411 | 0.403 | 0.408 |
| | 720 | 0.446 | 0.441 | 0.469 | 0.445 | 0.486 | 0.448 | 0.473 | 0.450 |
| | Avg | 0.390 | 0.396 | 0.396 | 0.404 | 0.413 | 0.406 | 0.405 | 0.410 |

trend component information. In contrast, pooling or convolutional layers select local maxima, thereby losing some global information. In practice, we can consider choosing MLP or Pooling methods as the down-sampling method.

**Table 3.** Results of different down-sampling methods on Weather and ETTm1.

| Model | | MLP | | MaxPool | | AvgPool | | Conv1d | |
|---|---|---|---|---|---|---|---|---|---|
| Metric | | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| Weather | 96 | 0.180 | 0.220 | 0.183 | 0.223 | 0.183 | 0.223 | 0.186 | 0.223 |
| | 192 | 0.227 | 0.259 | 0.230 | 0.262 | 0.230 | 0.262 | 0.232 | 0.268 |
| | 336 | 0.281 | 0.298 | 0.282 | 0.299 | 0.282 | 0.299 | 0.286 | 0.304 |
| | 720 | 0.357 | 0.347 | 0.357 | 0.347 | 0.358 | 0.347 | 0.362 | 0.352 |
| | Avg | 0.261 | 0.281 | 0.263 | 0.282 | 0.263 | 0.282 | 0.266 | 0.286 |
| ETTm1 | 96 | 0.326 | 0.362 | 0.328 | 0.363 | 0.328 | 0.363 | 0.332 | 0.365 |
| | 192 | 0.369 | 0.380 | 0.369 | 0.382 | 0.369 | 0.382 | 0.374 | 0.387 |
| | 336 | 0.402 | 0.404 | 0.400 | 0.403 | 0.400 | 0.402 | 0.407 | 0.407 |
| | 720 | 0.446 | 0.441 | 0.463 | 0.441 | 0.464 | 0.440 | 0.470 | 0.445 |
| | Avg | 0.390 | 0.396 | 0.390 | 0.396 | 0.390 | 0.396 | 0.395 | 0.401 |

**Effect of Historical Sequence Lengths.** Different lengths of historical sequence inputs may affect the forecasting performance of the model. In order to further explore its ability to leverage longer input sequences, we conduct experiments with various input lengths $L = \{48, 96, 192, 336, 512, 720\}$. We compare the performance of UNetPlusTS with DLinear [20], LightTS [21] and PatchTST [14] on Weather, ETTm1 and ETTm2 datasets. For brevity, the MAE results for prediction lengths of {96, 336, 720} are presented in Fig. 3. By analyzing the experimental results, we can see that as the length of the historical sequence increases from 96 to 336 or 512, our model gradually achieves the best performance. Although there is a slight decline when reaching 512 or 720, it can still maintain performance aligned with previous levels, possibly due to overfitting. Another possible reason is that, as the input size increases, the model may learn more, leading to poorer performance. Many transformer-based models (such as Informer and Autoformer) do not necessarily capture more information to achieve better results with longer input lengths, due to their complexity and are prone to overfitting. [3, 14] On the contrary, due to its simplicity and fewer parameters, DLinear is less prone to overfitting.

Overall, as the length of the input sequence increases, UNetPlusTS achieves better predictive performance, outperforming DLinear. This also demonstrates the effectiveness of the UNetPlusTS structure, which, through appropriate integration of seasonal and trend components, improves prediction accuracy without significantly increasing model complexity.

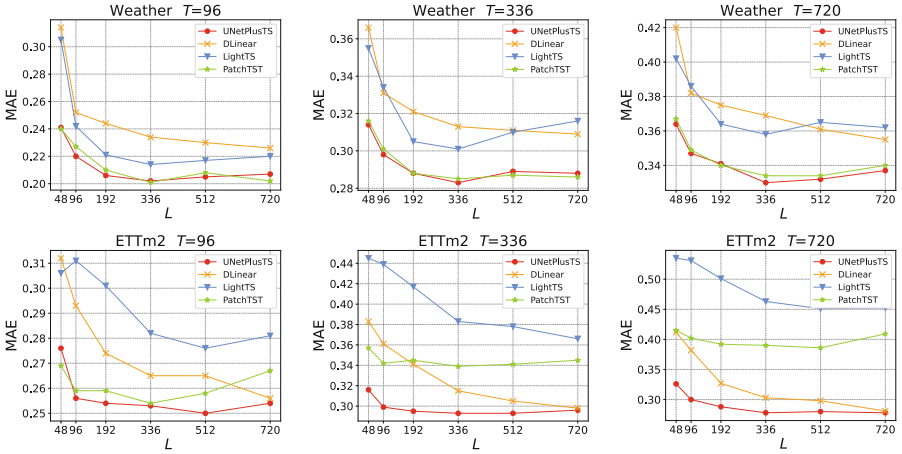
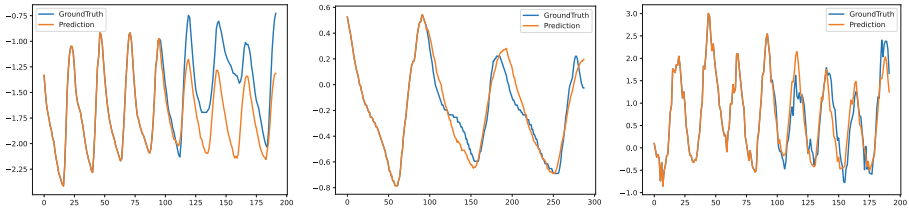



**Fig. 3.** MAE scores of UNetPlusTS, DLinear, LightTS and PatchTST with varying historical sequence input lengths on Weather and ETTm1. Historical sequence lengths are $L \in \{48, 96, 12, 336, 512, 720\}$, prediction lengths are $T \in \{96, 336, 720\}$.

## 4.5 Visualization

To further explore the performance of UNetPlusTS, we visualized its effects on long-term time series forecasting on ETTm1, ETTh2 and ECL datasets. Here, our input sequence length is 96 and prediction length is 96. As shown in Fig. 4, our model is capable of capturing their periodicity within a reasonable range and making certain forecasting about their trends.




(a) ETTh2 (b) ETTm2 (c) Electricity

**Fig. 4.** Visualization of long-term time series forecasting by UNetPlusTS.

### 4.6 Parameter Sensitivity

We explored the impact of the number of encoder-decoder layer $M$ on the model's performance on ETTm1 and Weather datasets. As shown in Fig. 5, the variation in the number of layers $M$ has a minimal impact on the performance of UNetPlusTS. However, it is worth noting that as the number of layers $M$ increases, there might be a slight improvement in model performance. In practice, we generally use $M = 4$ for optimal performance and efficiency.

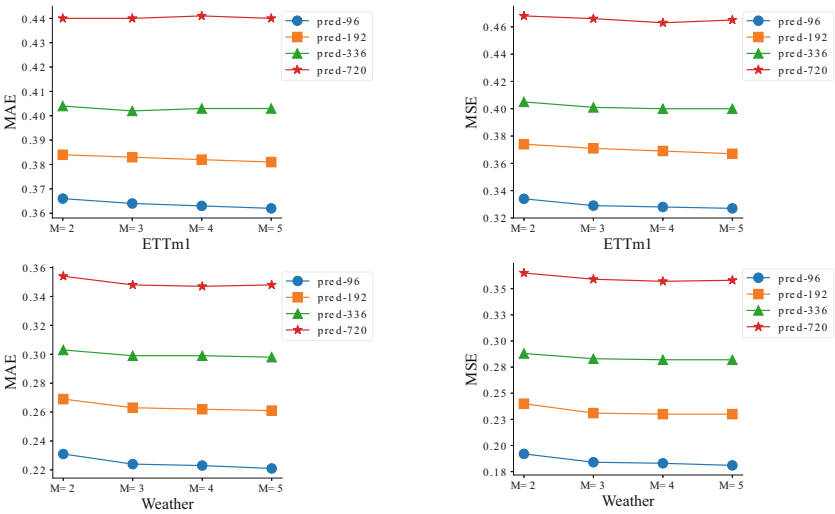



**Fig. 5.** Performance metrics with varying numbers of encoder-decoder layers.

## 5 Conclusion

In this paper, we propose a novel and effective model UNetPlusTS for long-term time series forecasting by introducing two key components: multi-scale mixing and improved UNet++ architecture. We split time series into different channels and decompose them into seasonal and trend components. Compared to previous works, by introducing UNet++ architecture, combined with a unique sampling strategy, we have achieved deep integration of seasonal and trend components and stationary information flow. Our model shows strong long-term time series forecasting performance on multiple real-world datasets, and further extension experiments also verify its effectiveness and robustness. In the future, we will further discuss the application of UNet series models in time series forecasting, and propose better methods and architectures to achieve higher prediction accuracy.

**Acknowledgments.** This work is supported by the Joint Innovation Laboratory for Future Observation, Zhejiang University, focusing on the research of cloud-native observability technology based on Guance Cloud (JS20220726-0016).

## References


1. Bai, S., Kolter, J.Z., Koltun, V.: An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271 (2018)
2. Cai, L., Janowicz, K., Mai, G., Yan, B., Zhu, R.: Traffic transformer: capturing the continuity and periodicity of time series for traffic forecasting. Trans. GIS **24**(3), 736–755 (2020)
3. Chen, S.A., Li, C.L., Yoder, N., Arik, S.O., Pfister, T.: TSmixer: an all-MLP architecture for time series forecasting. arXiv preprint arXiv:2303.06053 (2023)
4. Cleveland, R.B., Cleveland, W.S., McRae, J.E., Terpenning, I., et al.: STL: a seasonal-trend decomposition. J. Off. Stat. **6**(1), 3–73 (1990)
5. Das, A., Kong, W., Leach, A., Sen, R., Yu, R.: Long-term forecasting with tide: time-series dense encoder. arXiv preprint arXiv:2304.08424 (2023)
6. Fan, W., Zheng, S., Wang, P., Xie, R., Bian, J., Fu, Y.: Addressing distribution shift in time series forecasting with instance normalization flows. arXiv preprint arXiv:2401.16777 (2024)
7. Gu, A., Goel, K., Re, C.: Efficiently modeling long sequences with structured state spaces. In: International Conference on Learning Representations (2022). https://openreview.net/forum?id=uYLFoz1vlAC
8. Hamilton, J.D.: Time Series Analysis. Princeton University Press, Princeton (2020)
9. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
10. Karevan, Z., Suykens, J.A.: Transductive lstm for time-series prediction: an application to weather forecasting. Neural Netw. **125**, 1–9 (2020)
11. Lai, G., Chang, W.C., Yang, Y., Liu, H.: Modeling long-and short-term temporal patterns with deep neural networks. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, pp. 95–104 (2018)
12. Liu, S., Wu, K., Jiang, C., Huang, B., Ma, D.: Financial time-series forecasting: towards synergizing performance and interpretability within a hybrid machine learning approach. arXiv preprint arXiv:2401.00534 (2023)
13. Liu, Y., et al.: iTransformer: inverted transformers are effective for time series forecasting. arXiv preprint arXiv:2310.06625 (2023)
14. Nie, Y., H. Nguyen, N., Sinthong, P., Kalagnanam, J.: A time series is worth 64 words: long-term forecasting with transformers. In: International Conference on Learning Representations (2023)
15. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W., Frangi, A. (eds.) MICCAI 2015. LNCS, Part III, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
16. Wang, H., Peng, J., Huang, F., Wang, J., Chen, J., Xiao, Y.: MICN: multi-scale local and global context modeling for long-term series forecasting. In: The Eleventh International Conference on Learning Representations (2022)
17. Wang, S., et al.: TimeMixer: decomposable multiscale mixing for time series forecasting. In: The Twelfth International Conference on Learning Representations (2024). https://openreview.net/forum?id=7oLshfEIC2
18. Wu, H., Hu, T., Liu, Y., Zhou, H., Wang, J., Long, M.: TimesNet: temporal 2D-variation modeling for general time series analysis. In: The Eleventh International Conference on Learning Representations (2022)
19. Wu, H., Xu, J., Wang, J., Long, M.: Autoformer: decomposition transformers with auto-correlation for long-term series forecasting. In: Advances in Neural Information Processing Systems (2021)