Less Is More: Fast Multivariate Time Series Forecasting with Light Sampling-oriented MLP Structures

Tianping Zhang
Tsinghua University
Beijing, China
ztp18@mails.tsinghua.edu.cn

Yizhuo Zhang
Tsinghua University
Beijing, China
zyz22@mails.tsinghua.edu.cn

Wei Cao
Microsoft Research of Asia
Beijing, China
weica@microsoft.com

Jiang Bian
Microsoft Research of Asia
Beijing, China
jiang.bian@microsoft.com

Xiaohan Yi
Microsoft Research of Asia
Beijing, China
xiaoyi@microsoft.com

Shun Zheng
Microsoft Research of Asia
Beijing, China
shun.zheng@microsoft.com

Jian Li
Tsinghua University
Beijing, China
lijian83@mail.tsinghua.edu.cn

ABSTRACT
Multivariate time series forecasting has been widely ranging applications in various domains, including finance, traffic, energy, and healthcare. To capture the sophisticated temporal patterns, plenty of research studies designed complex neural network architectures based on many variants of RNNs, GNNs, and Transformers. However, complex models often suffer from computational expense and face a severe challenge in training and inference efficiency when applied to large-scale real-world datasets. In this paper, we introduce LightTS, a light deep learning architecture merely based on simple MLP-based structures. The key idea of LightTS is to apply an MLP-based structure on top of two delicate down-sampling strategies, including interval sampling and continuous sampling, inspired by a crucial fact that down-sampling time series often preserves the majority of its information. We conduct extensive experiments on eight widely used benchmark datasets. Compared with the existing state-of-the-art methods, LightTS demonstrates better performance on five of them and comparable performance on the rest. Moreover, LightTS is highly efficient. It uses less than 5% FLOPS compared with previous SOTA methods on the largest benchmark dataset. In addition, LightTS is robust and has a much smaller variance in forecasting accuracy than previous SOTA methods in long sequence forecasting tasks. Our codes and datasets are available in the anonymous link.

CCS CONCEPTS
• Computing methodologies → Artificial intelligence.

KEYWORDS
time series forecasting, multi-layer perceptrons

1 INTRODUCTION
Multivariate time series, as one of fundamental real-word data types, consists of more than one time-dependent variable, and, more importantly, each variable depends not only on its past values but also has some dependency on other variables. Multivariate time series forecasting has become a critical application task in various domains, including economics, traffic, energy, and healthcare. The vital modeling challenge lies in capturing 1) sophisticated temporal patterns (both short-term local patterns and long-term global patterns) of each variable as well as 2) complex inter-dependency among different variables. Due to the ability of deep learning to model complex patterns, especially its successful applications in computer vision and natural language processing tasks, there have been significantly growing research interests to apply deep neural networks into multivariate time series forecasting [15] rather than traditional methods (such as ARIMA [3], Holt-Winters [5], etc).

Particularly, with increasing computing power and the development of neural network architectures, many recent studies turned their eyes upon the realms of RNNs, GNNs, and Transformers. For instance, LSTNet [8] and TPR-LSTM [17] used the hybrids of Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and the Attention Mechanism [22] to capture the long/short-term dependencies. Informer [27] and Autoformer [23] further explored the potential of Transformer in capturing very long-range dependency of time series. MTGNN [24], StemGNN [4], and IGMTF [25] used Graph Neural Networks (GNNs) to explicitly model the dependencies among variables. While all these proposed complex deep learning models have demonstrated promising performance under specific scenarios, the introduced sophisticated

1https://tinyurl.com/5993cmus
neural network architectures usually indicate computationally expensive training and inference process, especially when facing the time series with long input length and many multiple correlated variables. Moreover, complex neural network architectures are always data-hungry due to the large number of parameters [7, 27], which may cause the trained models not robust when the amount of available training data is limited. Motivated by the challenges above, we naturally raise a question, is it necessary to apply complex and computationally expensive models to achieve state-of-the-art performance in multivariate time series forecasting?

In this paper, we explore the possibility of using simple and lightweight neural network architectures, i.e., merely using simple multi-layer perceptron (MLP) structure, for accurate multivariate time series forecasting. Specifically, relying on a critical observation that down-sampling time series often preserves the majority of its information [11] due to the trending, seasonal and irregular characteristics of time series, we thus propose LightTS, a light deep learning architecture constructed exclusively based on simple MLP-based structures. The key idea of LightTS is to apply an MLP-based structure on top of two delicate down-sampling strategies, including interval sampling and continuous sampling, inspired by crucial characteristics of multivariate time series. To be more concrete, continuous sampling focuses on capturing the short-term local patterns, while interval sampling focuses on capturing the long-term dependencies. On top of the sampling strategies, we propose an MLP-based architecture that can exchange information among different down-sampled sub-sequences and time steps. In such a way, the model can adaptively select useful information for forecasting from both local and global patterns. Furthermore, since each time the model only needs to deal with a fraction of the input sequence after down-sampling, our model is highly efficient in handling time series with very long input length.

The contributions of this paper are summarized as follows:

- We propose LightTS, a simple architecture that is highly efficient and accurate in multivariate time series forecasting tasks. To the best of our knowledge, this is the first work that demonstrates the great potential of the MLP-based structures in multivariate time series forecasting.
- We propose continuous and interval sampling according to the special property of time series. The sampling methods help capture long/short-term patterns effectively and enable greater efficiency for long input sequences.
- Experimental results show that LightTS is competent in both short sequence and long sequence forecasting tasks. LightTS outperforms the state-of-the-art methods on 5 out of 8 widely-used benchmark datasets. In addition, LightTS is highly efficient. LightTS uses less than 5% FLOPS on the largest benchmark dataset compared with previous SOTA methods. Moreover, LightTS is robust and has a much smaller variance in forecasting accuracy than previous SOTA methods in long sequence forecasting tasks.

We hope that our results can inspire further research beyond the realms of RNNs, GNNs, and Transformers in time series forecasting. We organize the rest of the paper as follows. We start with a survey of related work in Section 2. We formally define our problem in 3.1, and we present our model LightTS in Section 3.2. We present experimental results in Section 4. We provide discussion in Section 5. We conclude our paper in Section 6.

2 RELATED WORK

Time series forecasting methods can generally be classified into statistical and deep-learning-based methods.

2.1 Statistical Methods

Statistical methods for time series forecasting have been studied for a long time. Traditional methods include auto-regression (AR), moving average (MA), and auto-regressive moving average (ARMA). The auto-regressive integrated moving average model (ARIMA) [3] extends the ARMA model by incorporating the notion of integration. The vector autoregressive model (VAR) [3] is an extension of the AR model and captures the linear dependencies among different time series. Gaussian Process (GP) [6] is a Bayesian approach that models the distribution of multivariate time series over continuous functions. Statistical methods are popular for their simplicity and interpretability. However, those approaches require strong assumptions (such as the stationary assumption) and do not scale well to large multivariate time series data.

2.2 Deep-learning-based Methods

Recently, deep-learning-based methods have become increasingly popular in time series forecasting. LSTMNet [8] and TPR-LSTM [17] both employ the convolutional neural networks, the recurrent neural networks, and the attention mechanism [22] to model short-term local dependencies among different time series and long-term temporal dependencies. MTGNN [24], StemGNN [4], and IGMTF [25] leverage the graph neural networks to explicitly model the correlations among different time series. SCINet [11] downsamples the time series and uses convolutional filters to extract features and interact information. VARMPL [26] is a hybrid model that combines the AR model and an MLP with a single hidden layer, which is substantially different from LightTS we propose.

Long sequence time series forecasting (LSTF) has received increasing attention in the forecasting community. LSTF is important for various real-world applications (e.g., energy consumption planning). RNN-based models such as LSTMNet [8] have limited prediction capacity in LSTF [27]. Informer [27] improves the vanilla Transformer [22] in time complexity and memory usage for LSTF tasks. Autoformer [23] uses decomposition and an auto-correlation mechanism to achieve better results. We show through experiments that LightTS is more efficient and effective for long sequence time series forecasting.

Proposing a pure MLP-based model for multivariate time series forecasting is partly motivated by N-BEATS [15]. N-BEATS is the first pure deep-learning-based method for univariate time series forecasting that achieves SOTA in the M4 competition [13, 15]. In the ablation study of N-BEATS, where the backward residual connection is disabled, the architecture becomes a pure MLP-based structure. The architecture shows comparable results to N-BEATS.
and the winning solution of the M4 competition\(^3\). This finding supports the fact that, with careful design, MLP-based structures are powerful in capturing the historical patterns of time series.

Recently, several MLP-based architectures have been proposed for computer vision, such as MLP-Mixer [20], gMLP [10], and ResMLP [21]. Such architectures leverage information exchange over channels and spatial tokens. Compared with those architectures, our model is different since we consider the information exchange both in the original input and down-sampled sub-sequences.

3 OUR MODEL: LIGHTTS

3.1 Problem Definition

We first present the sampling strategies used in LightTS. Compared with the other sequential data such as natural language and audio data, time series is a special sequence in that down-sampling the time series often preserves the majority of its information [11]. Nevertheless, the naïve down-sampling method (such as the uniform sampling) could lead to information loss. Motivated by this, we design continuous and interval sampling, which helps the model capture both the local and global temporal patterns without eliminating any tokens.

Our sampling methods transform each time series of length \(T\) to several non-overlapping sub-sequences with length \(C\). For continuous sampling, each time we consecutively select \(C\) tokens as one sub-sequence. Thus, for a input sequence \(X \in \mathbb{R}^T\), we down-sample it to \(\mathbb{R}^{\frac{T}{C}}\) sub-sequences and obtain a 2D matrix \(X_{\text{con}} \in \mathbb{R}^{C \times \frac{T}{C}}\), where the \(j\)-th column is

\[
X_{\text{con}}^j = \{x_{T-T+(j-1)C+1}, x_{T-T+(j-1)C+2}, \ldots, x_{T-T+jC}\} \quad (1)
\]

For interval sampling, each time we sample \(C\) tokens with a fixed time interval. Similar to continuous sampling, the \(j\)-th column of the down-sample matrix for interval sampling \(X_{\text{int}}\) is

\[
X_{\text{int}}^j = \{x_{T-T+j}, x_{T-T+j+1}, \ldots, x_{T-T+j+(C-1)}\} \quad (2)
\]

The proposed sampling methods help the models focus on specific temporal patterns. For example, for continuous sampling, we sub-sample the time series into continuous pieces without long-range

---

\(^3\)The MLP-based architecture has an OWA of 0.822 in the M4 dataset, where OWA of the winning solution in the M4 competition is 0.821 and the second best is 0.838 [13]. OWA (overall weighted average of sMAPE and MASE) is the major accuracy measure of the M4 competition [13].
information, thus the model would pay more attention to the short-
term local patterns. For interval sampling, we put aside local details,
and the model would focus on long-term global patterns. Such
design makes the model more effective and efficient to train since
we only need to handle a fraction of the input sequence, especially
when the input sequence is very long. We emphasize that unlike
the naive down-sampling methods, we do not eliminate any tokens
here. Instead, we keep all the original tokens and transform them
into several non-overlapping sub-sequences. The following section
presents an MLP-based architecture to learn the useful features
from the down-sampled sub-sequences.

3.4 Information Exchange Block
Information Exchange Block (IEBlock) is the basic building block we
design for LightTS. In short, an IEBlock takes a 2D matrix of shape
$H \times W$, where $H$ is the temporal dimension and $W$ is the channel
dimension. The goal of an IEBlock is to leverage the information
exchange along different dimensions and outputs another feature
map of shape $F \times W$ ($F$ is the hyperparameter that corresponds to
the output feature dimension). The obtained matrix can be regarded
as the extracted features of the input matrix. IEBlocks are the key
components of LightTS and are used in both the down-sampling
part and the prediction part.

We present the architecture of IEBlock in Figure 2. We use $Z =
(z_{ij})_{H \times W}$ to denote the input 2D matrix, $z_{ij} = (z_{ij1}, z_{ij2},..., z_{ijH})^T$
to denote the $i$-th column and $z_j = (z_{j1}, z_{j2},..., z_{jW})$ to denote the $j$-
th row. We first apply an MLP of $\mathbb{R}^H \to \mathbb{R}^{F'}$ on each column, where
$F' \ll F$. We refer to such operation as the temporal projection:
$$z_{i}^{T} = \text{MLP}(z_{ij}), \forall i = 1, 2, ..., W$$

The temporal projection extracts features along the temporal di-

We call such an architecture the “bottleneck architecture”, where
the middle layer feature dimension is far less than the output layer
feature dimension.

Note that we use IEBlocks in both the down-sampling part and
prediction part. In different parts, $H$ and $W$ could have different
meanings. For example, in the sampling part, $H$ corresponds
to the length of sub-sequence $C$, and $W$ corresponds to the number
of sub-sequences $T$. In the prediction part, $H$ corresponds to the
feature dimension extracted in the first part, and $W$ corresponds
to the number of time series variables $N$.

In an IEBlock, we repeatedly apply the channel projection on
each time step. When the input time series is long, such an operation
leads to expensive computational costs. Therefore, we propose a
bottleneck design in IEBlock. First, we use a temporal projection
to map the number of rows from $H$ to $F'$. Then, the channel projection
is applied $F'$ times to communicate information. Finally, we use
an output projection to map the features $F'$ to the desired output
length $F$. We call this bottleneck design since the hyperparameter
$F'$ is much smaller than $H$ and $F$.

3.5 Training Procedure
Now, we can describe the training procedure of LightTS. In the first
part, we transform each time series with length $T$ to a 2D matrix
with shape $C \times \frac{T}{C}$. We then use two IEBlocks (IEBlock-A and IEBlock-
B in the Figure 1) to extract the corresponding temporal features
and obtain the feature matrices with shape $\mathbb{R}^{F' \times \hat{F}}$. Each feature
matrix is then down-projected to $\mathbb{R}^F$ with a simple linear mapping
$\mathbb{R}^{F'} \to \mathbb{R}$. Hence, we map each time series to a feature vector
of dimension $F$. This part focuses on capturing the long/short-term
temporal patterns without considering the correlations of different
time series variables.

In the second part, we concatenate all the obtained features in
the first part. We concatenate the features from continuous and
interval sampling on the temporal dimension and all the time se-
ries variables on the channel dimension. We thus obtain an input
matrix of shape $\mathbb{R}^{2FC \times N}$. Finally, we introduce another IEBlock-C of
$\mathbb{R}^{2FC \times N} \to \mathbb{R}^{L \times N}$. IEBlock-C aims to combine the short-term local
patterns and long-term global patterns from the temporal dimen-
sion and the correlations among different input variables from the
channel dimension. The output of IEBlock-C is our final prediction.
4 EXPERIMENTS

We validate LightTS on eight public benchmark datasets, from short sequence forecasting to long sequence forecasting. Following the previous studies [8, 27], we use the single-step setting for short sequence forecasting and the multi-step setting for long sequence forecasting. We demonstrate the advantages of LightTS over previous methods in terms of accuracy, efficiency, and robustness, respectively.

Table 1: Dataset statistics

| Datasets        | Variants | Timesteps | Granularity | Start time |
|-----------------|----------|-----------|-------------|------------|
| ETTh1           | 7        | 17420     | 1 hour      | 2016/7/1   |
| ETTh2           | 7        | 17420     | 1 hour      | 2016/7/1   |
| ETTm1           | 7        | 69680     | 15 minutes  | 2016/7/1   |
| Weather         | 12       | 35064     | 1 hour      | 2010/1/1   |
| Solar-Energy    | 137      | 17544     | 10 minutes  | 2006/1/1   |
| Traffic         | 862      | 52560     | 1 hour      | 2015/1/1   |
| Electricity     | 321      | 26304     | 1 hour      | 2012/1/1   |
| Exchange-Rate   | 8        | 7588      | 1 day       | 1990/1/1   |

4.1 Experimental Settings

4.1.1 Datasets. In Table 1, we summarize the statistics of eight benchmark datasets. Following the experimental setting in the previous studies [11, 27], for short sequence forecasting, we use the Solar-Energy, Traffic, Electricity and Exchange-Rate datasets [8, 24]; for long sequence forecasting, we use the Electricity Transformer Temperature (ETTh1, ETTh2, ETTm1), Electricity and Weather datasets [27]. We provide more details about the datasets in Appendix A.1.

4.1.2 Evaluation Metrics. For a fair comparison, we use the same evaluation metrics as previous studies [8, 27]. We use the Mean Squared Error (MSE) and Mean Absolute Error (MAE) for long sequence forecasting. We use the Root Relative Squared Error (RSE) and Empirical Correlation Coefficient (CORR) for short sequence forecasting. Readers can find more detailed information in Appendix A.2.

4.2 Baseline Methods for Comparisons

We summarize the baseline methods in the following:

4.2.1 Short sequence forecasting.

- AR: An auto-regressive model.
- VARMLP [26]. A hybrid model that combines the AR model and an MLP.
- GP [16]. A Gaussian Process model.
- RNN-GRU [24]. A recurrent neural network with GRU hidden units.
- LSTNet [8]. A hybrid model that combines convolutional neural networks and recurrent neural networks.
- TPR-LSTM [17]. A hybrid model that combines recurrent neural networks and the attention mechanism.
- TCN [2]. A typical temporal convolutional network.
- SCINet [11]. A forecasting model with sample convolution and interaction.
- MTGNN [24]. A GNN-based method that explicitly models the correlations among different time series.

4.2.2 Long sequence forecasting. In addition to LSTNet and SCINet, we compare with the following baseline methods for long sequence forecasting:

- LogTrans [9]. A variant of Transformers with LogSparse self-attention mechanism.
- Reformer [7]. An efficient variant of Transformers with locality-sensitive hashing.
- LSTMa [1]. A variant of recurrent neural networks with dynamic length of encoding vectors.
- Informer [27]. A variant of Transformers with ProbSparse self-attention mechanism.
- Autoformer [23]. A variant of Transformers with a decomposition forecasting architecture.

4.3 Main Results

Table 2 and Table 3 show the main results of LightTS. We can observe that LightTS can achieve state-of-the-art or comparable results in most cases. In the following, we discuss the results of short sequence forecasting and long sequence forecasting, respectively.

4.3.1 Long sequence forecasting. Table 2 presents the experimental results on long sequence forecasting tasks. LightTS achieves state-of-the-art results over all the horizons in ETTh1, ETTh2, ETTm1, and Electricity datasets and achieves second best results in the Weather dataset. In particular, for the longest prediction horizon, LightTS lowers MSE by 9.21%, 33.90%, 34.18%, and 13.60% on the ETTh1, ETTh2, ETTm1, and Electricity datasets, respectively. Compared with transformer models (LogTrans, Reformer, Informer), RNN-based models (LSTNet, LSTMa), and CNN-based models (TCN, SCINet), LightTS achieves significant improvement in long sequence forecasting tasks. On the one hand, continuous and interval sampling can help the model capture the short-term local patterns and long-term global dependencies. On the other hand, such a down-sampling design is critical for the model to process a very long input sequence efficiently.

4.3.2 Short sequence forecasting. Table 3 presents the experimental results on the task of short sequence forecasting. We can observe that LightTS achieves state-of-the-art results on the Solar-Energy dataset. In particular, on Solar-Energy dataset, LightTS lowers down RSE by 4.16%, 4.61%, 3.90%, 2.73% on horizon 3, 6, 12, 24 respectively. We also observe an inconsistency in the evaluation metrics of RSE and CORR. For example, on Traffic datasets, LightTS achieves state-of-the-art results in terms of RSE but slightly lags behind on CORR. For Traffic, Electricity, and Exchange-Rate datasets, none of the existing models can consistently outperform other models. LightTS provides comparable results to MTGNN and SCINet on these datasets. Moreover, as we will see in the next section, LightTS is efficient and has significant advantages over MTGNN and SCINet in FLOPS and running time.

4.4 Comparisons on FLOPS and Running Time

We compare the FLOPS and running time of LightTS with previous SOTA models (Autoformer, MTGNN, SCINet). We present the results of the three largest datasets in short sequence forecasting and four datasets in long sequence forecasting in Table 4 and 5. The hyperparameters of LightTS, MTGNN, and SCINet align with
### Table 2: Baseline comparisons under multi-step setting for long sequence time series forecasting tasks.

| Methods          | Metrics | Solar-Energy | Traffic | Electricity | Exchange-Rate |
|------------------|---------|--------------|---------|-------------|---------------|
|                  |         | 3 6 12 24    | 3 6 12 24 | 3 6 12 24   | 3 6 12 24     |

### Table 3: Baseline comparisons under single-step setting for short sequence time series forecasting tasks.

| Methods          | Metrics | Solar-Energy | Traffic | Electricity | Exchange-Rate |
|------------------|---------|--------------|---------|-------------|---------------|
|                  |         | 3 6 12 24    | 3 6 12 24 | 3 6 12 24   | 3 6 12 24     |
Table 4: The FLOPS of Autoformer, MTGNN, SCINet and LightTS [19]. The best model is marked in bold. We report the FLOPS of the models corresponding to the longest forecasting horizon of each dataset in Table 2 and 3. Solar: Solar-Energy. ECL: Electricity.

| FLOPS (M) | Long Sequence | Short Sequence |
|-----------|---------------|----------------|
|           | ETTh1 | ETTm1 | Weather | ECL | Traffic | Solar | ECL |
| AutoFormer | 7581 | 7581 | 7600 | 11652 | - | - | - |
| MTGNN | - | - | - | - | 2370 | 377 | 883 |
| SCINet | 6 | 17 | 15 | 5078 | 16348 | 205 | 57 |
| LightTS | 4 | 3 | 7 | 328 | 90 | 10 | 30 |
- dash denotes that the method does not implement on this task.

Table 5: We report the running time of one epoch in seconds for Autoformer, MTGNN, SCINet and LightTS. We report the running time of each dataset with the longest forecasting horizon in Table 2 and 3. The experimental environment and batch size are the same for each model. The best model is marked in bold. Solar: Solar-Energy. ECL: Electricity.

| Time (s) | Long Sequence | Short Sequence |
|----------|---------------|----------------|
|           | ETTh1 | ETTm1 | Weather | ECL | Traffic | Solar | ECL |
| AutoFormer | 80 | 355 | 280 | 330 | - | - | - |
| MTGNN | - | - | - | - | 1470 | 213 | 850 |
| SCINet | 80 | 850 | 260 | 950 | 455 | 275 | 3750 |
| LightTS | 2 | 9 | 7 | 160 | 33 | 33 | 50 |
- dash denotes that the method does not implement on this task.

the ones that generate the forecasting results in Table 2 and 3. We can observe that LightTS has significant advantages in FLOPS and running time in both short sequence and long sequence forecasting. For the Traffic dataset (the largest dataset with the greatest number of variables), the FLOPS of LightTS is 96.2% smaller than MTGNN and 99.4% smaller than SCINet. In addition, LightTS achieves a speedup of 44.5x over MTGNN and 13.8x over SCINet in terms of running time per epoch. For the Electricity dataset in long sequence forecasting tasks, the FLOPS of LightTS is 93.5% smaller than SCINet and 97.2% smaller than Autoformer. We perform the experiments on a server with two 12-core Intel(R) Xeon(R) CPU E5-2690 v4 @ 2.60GHz and one Tesla V100 PCIe 16GB.

4.5 Robustness Analysis

Robustness is a critical problem in long sequence time series forecasting. A wrong forecast on the trend or seasonality can gradually accumulate over time, resulting in incredibly misleading forecasting. During the experiments, we find that previous models for long sequence forecasting are not robust across random seeds, while LightTS can provide stable results. We present the standard deviation of Autoformer, SCINet, and LightTS on the ETTh1, ETTm1, Weather, and Electricity datasets in Table 6. We can observe that LightTS has a much smaller variance in prediction accuracy than Autoformer and SCINet. We also present the shaded area of the forecasting by different random seeds for LightTS, SCINet, and Autoformer in Figure 3. We can observe that LightTS has a much smaller shaded area of forecasting across different random seeds than SCINet and Autoformer. LightTS has a significant advantage over existing methods in long sequence forecasting from both accuracy and robustness.

4.6 Ablation Study

We conduct an ablation study to investigate the effectiveness of the components we propose in LightTS. We name LightTS without different components as follows:

- w/o CP: LightTS without the channel projection. In this way, we do not rely on the interdependency of different time series to make predictions.
- w/o IS: LightTS without the interval sampling.
- w/o CS: LightTS without the continuous sampling.
Figure 4: The ground truth (black) and the forecast (orange) for one variable in the Traffic dataset with the forecasting horizon of 24. The variable has daily patterns (short-term local patterns) and weekly patterns (long-range patterns). The green arrows point at daily patterns, where (a) and (b) which have continuous sampling are more accurate in forecasting than (c). The blue arrows point at weekly patterns, where (a) and (c) which have interval sampling are more accurate in forecasting than (b).

We repeat each experiment 5 times with different random seeds. We report the mean and standard deviation of RSE and CORR over five runs in Table 7. The introduction of channel projection significantly improves the performance of Solar and Electricity datasets. The interdependencies in these two datasets are evident. On the contrary, the interdependency in the Exchange-Rate dataset is not evident. The observation is similar in MTGNN [24] where MTGNN also fails to achieve desirable results in Exchange-Rate. Removing channel projection improves the results in the Exchange-Rate dataset. When the interdependencies among different time series are not evident, forcing the model to capture interdependency may negatively impact the model’s performance. The effects of interval and continuous sampling are evident for all the datasets. We also demonstrate how interval sampling and continuous sampling affect the forecasting of LightTS in Figure 4. The results demonstrate that continuous sampling helps LightTS capture the short-term local patterns, and interval sampling helps LightTS capture the long-range patterns.

Table 7: The results of the ablation study on different datasets. The forecasting horizon is set to 24 for each dataset. Best result in each setting is marked in bold. Solar: Solar-Energy. Exchange: Exchange-Rate.

| Methods  | Metrics | Solar   | Traffic | Electricity | Exchange |
|----------|---------|---------|---------|-------------|----------|
| LightTS  | RSE     | 0.4133  | 0.4416  | 0.0985      | 0.0453   |
|          | CORR    | 0.9065  | 0.8699  | 0.9191      | 0.9360   |
| w/o CP   | RSE     | 0.4469  | 0.4489  | 0.0998      | 0.0441   |
|          | CORR    | 0.8912  | 0.8639  | 0.9119      | 0.9364   |
| w/o IS   | RSE     | 0.4166  | 0.4571  | 0.1008      | 0.0486   |
|          | CORR    | 0.9062  | 0.8591  | 0.9126      | 0.9345   |
| w/o CS   | RSE     | 0.4174  | 0.4425  | 0.0997      | 0.0475   |
|          | CORR    | 0.9044  | 0.8698  | 0.9193      | 0.9333   |

Figure 5: The correlation between the interdependency modeled by LightTS and MTGNN. The improvement of channel projection for Solar and Electricity is significant in our ablation study, which suggests that the interdependencies among different variables are evident in these two datasets.

For the first difficulty, we use the Deep SHAP (DeepLIFT + Shapley values) [12, 18] to explain (and quantify) the interdependency among different time series in LightTS. Deep SHAP is a popular method that interprets the prediction of deep learning models based on DeepLIFT [18] and SHAP [12]. For a given input matrix $X = (x_{ij})_{N \times T}$, we assume the predictions of LightTS are...
$\hat{y} = (\hat{y}_1, \hat{y}_2, ..., \hat{y}_N)$. For the prediction of the $k$-th time series $\hat{y}_k$ ($k \in \{1, 2, ..., N\}$), Deep SHAP can explain how each element in the input matrix contributes to the prediction by LightTS. Assume the Deep SHAP values of the input $X$ are $S^e = (s^e_{ij})_{N \times T}$, where $s^e_{ij}$ is the attribution of input $x_{ij}$ for prediction $\hat{y}_k$. Due to the fact that Deep SHAP is an additive feature attribution method [12], the sum of the Deep SHAP values approximates the prediction [12]:

$$\hat{y}_k = \phi_0 + \sum_{i=1}^{N} \sum_{j=1}^{T} s^e_{ij}$$

where $\phi_0$ represents the model output with all the inputs missing [12]. Therefore, $s^e_{ij}$ evaluates how the input of time series $i$ contributes to the prediction $\hat{y}_k$ of time series $k$, in other words, how the prediction of time series $k$ by LightTS depends on time series $i$. We use the matrix $E = (e_1, e_2, ..., e_N)$ to quantify the interdependency among different time series in LightTS, where $e_k = (s^e_{1k}, s^e_{2k}, ..., s^e_{Nk})$. See Appendix A.3 for more implementation details.

The question turns to how to measure the quality of the interdependency modeled by LightTS. The “real” interdependency among time series is not defined for many multivariate time series data. In this case, we use the interdependency modeled by MTGNN [24] as the “ground truth” to measure the quality of the interdependency modeled by LightTS. MTGNN [24] is a GNN-based method that explicitly learns the graph structure during training and is one of the state-of-the-art methods in multivariate time series forecasting. Following the above steps, we can also calculate matrix $M = (m_1, m_2, ..., m_N)$ to quantify the interdependency in MTGNN. We calculate the following metric: $corr(e_i, m_1) = \frac{1}{N} \sum_{i=1}^{N} corr(e_i, m_1)$ where $corr(\cdot, \cdot)$ is the Pearson correlation between two vectors. We present the results of 8 cases in our experiments in Figure 5. We can see that the interdependency modeled by LightTS highly correlates with the one modeled by MTGNN. These empirical findings show that a simple channel projection is sufficient in learning the interdependency among different time series.

6 CONCLUSION

This paper proposes LightTS, a simple model merely based on multi-layer perceptrons. The key idea of LightTS is to apply an MLP-based framework on top of two down-sampling strategies. Continuous and interval sampling helps the model capture the short-term local patterns and long-term temporal dependencies. Furthermore, since the model only needs to deal with a fraction of the input sequence after down-sampling, our model is highly efficient in handling time series with very long input sequences. Extensive experiments show that LightTS is accurate, efficient, and robust in short sequence and long sequence multivariate time series forecasting tasks.

REFERENCES

[1] Dmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014).

[2] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271 (2018).

[3] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. 2015. Time series analysis: forecasting and control. John Wiley & Sons.

[4] Defu Cao, Yujing Wang, Juanyong Duan, Cezhang Chen, Xia Zhu, Congrui Huang, Yunhai Tong, Bixiong Xu, Jie Bai, Jie Tong, et al. 2021. Spectral temporal graph neural network for multivariate time-series forecasting. arXiv preprint arXiv:2103.07719 (2021).

[5] Chris Chatfield. 1978. The Holt-winters forecasting procedure. Journal of the Royal Statistical Society: Series C (Applied Statistics) 27, 3 (1978), 264–279.

[6] Roger Frigola. 2015. Bayesian time series learning with Gaussian processes. Ph.D. Dissertation. University of Cambridge.

[7] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskyka. 2020. Transformer: the efficient transformer. arXiv preprint arXiv:2001.04451 (2020).

[8] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. 2018. Modeling long- and short-term temporal patterns with deep neural networks. In The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval. 95–104.

[9] Shuyang Li, Xiaoyang Jin, Yao Xuan, Xiyue Zhou, Wenwu Chen, Yu-Xiang Wang, and Xifeng Yan. 2019. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. Advances in Neural Information Processing Systems 32 (2019), 5243–5253.

[10] Hanxiao Liu, Zihang Dai, David R So, and Quoc V Le. 2021. Pay Attention to MLPs. arXiv preprint arXiv:2105.06950 (2021).

[11] Minhao Liu, Ailing Zeng, Qixia Li, and Qiang Xu. 2021. Time Series is a Special Sequence: Forecasting with Sample Convolution and Interaction. arXiv preprint arXiv:2106.09905 (2021).

[12] Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In Proceedings of the 31st international conference on neural information processing systems. 4768–4777.

[13] Spyros Makridakis, Evangelios Spiliotis, and Vassilios Assimakopoulos. 2018. The M4 Competition: Results, findings, conclusion and way forward. International Journal of Forecasting 34, 4 (2018), 802–808.

[14] Shun-Yao Shih, Fan-Keng Sun, and Hung-yi Lee. 2019. Temporal pattern attention for multivariate time series forecasting. Machine Learning 108, 8 (2019), 1421–1441.

[15] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning important features through propagating activation differences. In International Conference on Machine Learning. PMLR, 3145–3153.

[16] Vladislav Sovrasov. 2019. Flops counter for convolutional networks in pytorch framework. https://github.com/sovrasov/flops-counter-pytorch.

[17] Stephen Roberts, Michael Osborne, Mark Ethden, Steven Reece, Neale Gibson, and Suzanne Aigrain. 2015. Gaussian processes for time-series modelling. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences 371, 2015 (2015), 20150550.

[18] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning important features through propagating activation differences. In International Conference on Machine Learning. PMLR, 3145–3153.

[19] Shun-Yao Shih, Fan-Keng Sun, and Hung-yi Lee. 2019. Temporal pattern attention for multivariate time series forecasting. Machine Learning 108, 8 (2019), 1421–1441.

[20] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Peter Steiner, Daniel Keysers, Jakob Uszkoreit, et al. 2021. MLP-mixer: An all-mlp architecture for vision. In Thirty-Fifth Conference on Neural Information Processing Systems.

[21] Hugo Touvron, Piotr Bojanowski, Mathilde Caron, Matthieu Cord, Amael Dehghani, Edouard Grave, Gautier Izacard, Armand Joulin, Gabriel Synnaeve, Jacob Uszkoreit, et al. 2021. Resmlp: Feedforward networks for image classification with data-efficient training. arXiv preprint arXiv:2105.03404 (2021).

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Advances in neural information processing systems. 5998–6008.

[23] Zonghan Wu, Shuai Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. 2020. Connecting the dots: Multivariate time series forecasting with graph neural networks. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 753–763.

[24] Wentao Xu, Weiqing Liu, Jiang Bian, Jin Yan, and Tiejian Liu. 2021. Instance-wise Graph-based Framework for Multivariate Time Series Forecasting. arXiv preprint arXiv:2109.06489 (2021).

[25] G Peter Zhang. 2003. Time series forecasting using a hybrid ARIMA and neural network model. Neurocomputing 50 (2003), 159–175.

[26] Haoyi Zhou, Shanghang Zhang, Jieqiu Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Inferred: Beyond efficient transformer for long sequence-time series forecasting. In Proceedings of AAAI.
A IMPLEMENTATION DETAILS

A.1 Datasets Description

In Table 1, we show basic information of datasets which we use to evaluate performance. Here we introduce details of these datasets.

- **Electricity Transformer Temperature (ETT):** Introduced by [27], ETT datasets record 2-year electric power in two different counties in China. ETTm1 has a sample rate of 15 minutes, while ETTh1 and ETTh2 have a sample rate of 1 hour. These three datasets have seven variables in each timestamp. We use the three ETT datasets mentioned above to evaluate long sequence forecasting abilities. The train/val/test is 6/2/2.

- **Weather:** This dataset contains 4-year climatological data of multiple U.S. locations, in which data points are collected every 1 hour. This dataset has 12 variables in every timestamp. We use this dataset to evaluate long sequence forecasting abilities. The train/val/test is 7/1/2.

- **Electricity:** Electricity dataset (also known as Electricity Consuming Load, or ECL) collects electricity consumption of 321 users in two years (2012-2014) every 15 minutes. We use this dataset to evaluate long and short sequence forecasting abilities. In long sequence forecasting, the train/val/test is 7/1/2. In short sequence forecasting, the train/val/test is 6/2/2.

- **Solar-Energy:** This dataset (also called Solar) records the solar power production of 137 photovoltaic plants in Alabama State in 2006 every 10 minutes. We use this dataset to evaluate short sequence forecasting abilities. The train/val/test is 6/2/2.

- **Traffic:** This dataset collects two years (2015-2016) of hourly road occupancy rates data in California. Each timestamp contains 862 variables. We use this dataset to evaluate short sequence forecasting abilities. The train/val/test is 6/2/2.

- **Exchange-Rate:** This dataset collects daily exchange rates of eight countries, including Australia, Canada, China, Japan, New Zealand, Singapore, Switzerland, and the United Kingdom, from 1990 to 2016. We use this dataset to evaluate short sequence forecasting abilities. The train/val/test is 6/2/2.

A.2 Evaluation Metrics

A.2.1 Long Sequence Forecasting. Following [27], we use Mean Squared Error (MSE) and Mean Absolute Error (MAE) defined as follows:

- **Mean Squared Error (MSE):**
  \[
  MSE = \frac{1}{n} \sum_{i=1}^{n} (Y - \hat{Y})^2
  \]

- **Mean Absolute Error (MAE):**
  \[
  MAE = \frac{1}{n} \sum_{i=1}^{n} |Y - \hat{Y}|
  \]

where \(Y, \hat{Y} \in \mathbb{R}^{l \times N}\) (\(l\) is the output length, \(N\) is the number of variables) are the ground truth and prediction, \(n\) is the size of the test set. For both MSE and MAE, a lower value means better performance.

A.2.2 Short Sequence Forecasting. Following [8], we use Root Relative Squared Error (RSE), and Empirical Correlation Coefficient (CORR) defined as follows:

- **Root Relative Squared Error (RSE):**
  \[
  RSE = \frac{\sqrt{\sum_{(t,i)\in \Omega_{test}} (Y_{ti} - \hat{Y}_{ti})^2}}{\sqrt{\sum_{(t,i)\in \Omega_{test}} (Y_{ti} - \text{mean}(Y))^2}}
  \]

- **Empirical Correlation Coefficient (CORR):**
  \[
  CORR = \frac{\sum_{i=1}^{n} (Y_{ti} - \text{mean}(Y))(\hat{Y}_{ti} - \text{mean}(\hat{Y}))}{\sqrt{\sum_{i=1}^{n} (Y_{ti} - \text{mean}(Y))^2}(\hat{Y}_{ti} - \text{mean}(\hat{Y}))^2}
  \]

where \(Y, \hat{Y} \in \mathbb{R}^{l \times N}\) (\(l\) is the output length, \(N\) is the number of variables) are the ground truth and prediction, and \(\Omega_{test}\) is test set. For RSE, a lower value means better performance. For CORR, higher CORR means better performance.

A.3 SHAP Values

In Section 5, we use the Deep SHAP (DeepLIFT + Shapley values) [12, 18] to explain the prediction of LightTS. We use 100 samples (input matrix) randomly selected from the training set as the background dataset for integrating out features. The SHAP value of each input element is its average SHAP value on 100 randomly selected test samples.

B ADDITIONAL EXPERIMENT RESULTS

In this section, we show additional experiment results on long sequence forecasting in Table 8.
Table 8: Additional baseline comparisons under multi-step setting for long sequence time series forecasting tasks.

| Methods    | Metrics | ETTh1 24 | ETTh2 48 | ETTm1 24 | Weather 48 | Electricity 168 |
|------------|---------|---------|---------|----------|-----------|-----------------|
|            | MSE     | 0.656   | 0.726   | 0.341    | 0.365     | 0.267           |
|            | MAE     | 0.600   | 0.663   | 0.495    | 0.405     | 0.366           |
| LogTrans   | MSE     | 0.887   | 1.381   | 0.598    | 0.583     | 1.312           |
|            | MAE     | 0.630   | 1.475   | 0.489    | 0.497     | 0.911           |
| Reformer   | MSE     | 0.536   | 1.049   | 0.511    | 0.476     | 0.388           |
|            | MAE     | 0.528   | 0.689   | 0.517    | 0.464     | 0.444           |
| LSTMa      | MSE     | 1.175   | 1.344   | 2.632    | 1.856     | 0.757           |
|            | MAE     | 0.793   | 1.337   | 1.058    | 0.507     | 0.337           |
| LSTNet     | MSE     | 0.509   | 0.551   | 0.446    | 0.325     | 0.279           |
|            | MAE     | 0.523   | 0.563   | 0.440    | 0.325     | 0.269           |
| Informer   | MSE     | 0.509   | 0.551   | 0.446    | 0.325     | 0.279           |
|            | MAE     | 0.523   | 0.563   | 0.440    | 0.325     | 0.269           |
| Autoformer*| MSE     | 0.408   | 0.502   | 0.302    | 0.150     | 0.175           |
|            | MAE     | 0.434   | 0.451   | 0.374    | 0.264     | 0.259           |
| SCINet*    | MSE     | 0.395   | 0.395   | 0.287    | 0.128     | 0.175           |
|            | MAE     | 0.385   | 0.411   | 0.231    | 0.322     | 0.252           |
| LightTS    | MSE     | 0.356   | 0.356   | 0.178    | 0.105     | 0.140           |
|            | MAE     | 0.560   | 0.384   | 0.269    | 0.197     | 0.244           |
| Improvement| MSE     | 11.05%  | 8.74%   | 7.53%    | 5.32%     | 5.39%           |
|            | MAE     | 7.53%   | 6.57%   | 6.27%    | 14.72%    | 3.17%           |

Results are taken from [27] (for results from LogTrans to Informer). Best result in one setting is marked in **bold**, and second best is marked in *italic.* We use 5 seeds to calculate the average result of SCINet and AutoFormer. We follow the same look-back settings in Informer and SCINet [11, 27] for each dataset.