Appendix: Unlocking the Power of LSTM for Long Term Time Series Forecasting

In this Appendix, we provide a motivating example illustrating the limitations of LSTM, a detailed description of the sLSTM structure, including its stabilized version and block design, and the specific experimental settings along with the chosen hyperparameters. We also conduct an ablation study on the stabilizer gate and present the forecasting results for several time series from the electricity dataset using P-sLSTM. Additionally, the code required to reproduce the results presented in the paper is provided.

A. An Example About the Limitation of LSTM

As we can see from figure 4, the traditional LSTM has difficulty in revising historical memory decision, so it cannot successfully make adjustments according to sudden changes in time series data. Solving this problem is an important motivation for designing more advanced RNNs/LSTM structures in TSF.

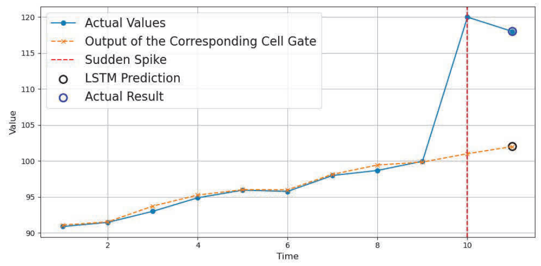

Figure 4: A toy time series example where the past 10 values are used to predict the next value. In this example, there is a spike at the last time step. While LSTM successfully captures the trend of the first 9 time steps, it overlooks the information from the 10th time step, leading to inaccurate predictions.

B. Methodology

B.1. sLSTM

The sLSTM forward pass is defined as follows:

1. **Cell State and Normalized State:**

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{z}_t, \quad \mathbf{n}_t = \mathbf{f}_t \odot \mathbf{n}_{t-1} + \mathbf{i}_t$$

2. **Hidden State:**

$$\mathbf{h}_t = \mathbf{o}_t \odot \tilde{\mathbf{h}}_t, \quad \tilde{\mathbf{h}}_t = \mathbf{c}_t \odot \mathbf{n}_t{}^{-1}$$

3. **Cell Input:**

$$\mathbf{z}_t = \varphi(\tilde{\mathbf{z}}_t), \quad \tilde{\mathbf{z}}_t = \mathbf{W}_\mathbf{z}\mathbf{x}_t + \mathbf{R}_\mathbf{z}\mathbf{h}_{t-1} + \mathbf{b}_\mathbf{z}$$

4. **Input Gate:**

$$\mathbf{i}_t = \exp(\tilde{\mathbf{i}}_t), \quad \tilde{\mathbf{i}}_t = \mathbf{W}_\mathbf{i}\mathbf{x}_t + \mathbf{R}_\mathbf{i}\mathbf{h}_{t-1} + \mathbf{b}_\mathbf{i}$$

5. **Forget Gate:**

$$\mathbf{f}_t = \sigma(\tilde{\mathbf{f}}_t) \text{ or } \exp(\tilde{\mathbf{f}}_t), \quad \tilde{\mathbf{f}}_t = \mathbf{W}_\mathbf{f}\mathbf{x}_t + \mathbf{R}_\mathbf{f}\mathbf{h}_{t-1} + \mathbf{b}_\mathbf{f}$$

6. **Output Gate:**

$$\mathbf{o}_t = \sigma(\tilde{\mathbf{o}}_t), \quad \tilde{\mathbf{o}}_t = \mathbf{W}_\mathbf{o}\mathbf{x}_t + \mathbf{R}_\mathbf{o}\mathbf{h}_{t-1} + \mathbf{b}_\mathbf{o}$$

Here, $\odot$ denotes elementwise multiplication, φ is the cell input activation function (tanh), σ is the sigmoid function, $\exp$ is the exponential function, and the hidden state activation function is the identity.

As illustrated in Figure 5, when new data $\mathbf{x}_t$ arrives, the process begins with the calculation of the 4 gates. The **cell input** $\mathbf{z}_t$ is computed by first calculating the intermediate cell input $\tilde{\mathbf{z}}_t$ using a linear transformation of $\mathbf{x}_t$ and the previous hidden state $\mathbf{h}_{t-1}$. This value is then passed through an activation function φ (typically tanh) to generate the updated cell input $\mathbf{z}_t$, which represents the new information that might be added to the memory.

The **input gate** $\mathbf{i}_t$ determines how much of this new information should be integrated into the cell state. This gate is calculated by applying an exponential function to the intermediate input gate value $\tilde{\mathbf{i}}_t$, which is derived from $\mathbf{x}_t$ and $\mathbf{h}_{t-1}$. The input gate modulates the influence of the new input on the cell state.

Concurrently, the **forget gate** $\mathbf{f}_t$ evaluates how much of the previous cell state $\mathbf{c}_{t-1}$ should be retained by applying either a sigmoid or an exponential function to the intermediate forget gate $\tilde{\mathbf{f}}_t$. This forget gate allows the model to selectively forget or keep past information based on the relevance of the new input.

The **output gate** $\mathbf{o}_t$ regulates how much of the intermediate hidden state should contribute to the final output. This gate is computed by applying the sigmoid function to the intermediate output gate value $\tilde{\mathbf{o}}_t$.

With the four gates calculated, the updated **cell state** $\mathbf{c}_t$ is computed by combining the retained past information (influenced by $\mathbf{f}_t$) with the new input (influenced by $\mathbf{i}_t$). This ensures that the cell state integrates both historical and current data.

The **normalizer state** $\mathbf{n}_t$ is also updated to maintain the stability of the cell state's magnitude.

Finally, the **hidden state** $\mathbf{h}_t$ is derived by normalizing the updated cell state $\mathbf{c}_t$ with the normalizer state $\mathbf{n}_t$, resulting in an intermediate hidden state $\tilde{\mathbf{h}}_t$. This intermediate state is then modulated by the output gate $\mathbf{o}_t$ to determine how much of it should contribute to the final output.

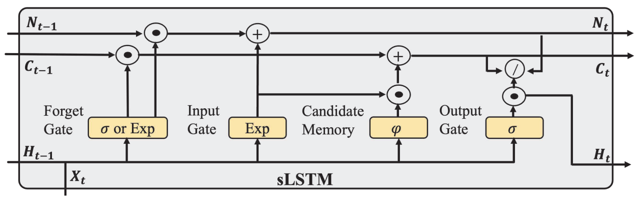

Figure 5: sLSTM Structure.

B.2. sLSTM Stabilized Version

In the sLSTM model, the forget gate with an exponential activation function $\mathbf{f}_t$ is defined as:

$$\mathbf{f}_t = \exp(\tilde{\mathbf{f}}_t), \quad \tilde{\mathbf{f}}_t = \mathbf{W_f}\mathbf{x}_t + \mathbf{R_f}\mathbf{h}_{t-1} + \mathbf{b_f}$$

Here, $\tilde{\mathbf{f}}_t$ is the pre-activation of the forget gate, determined by the input $\mathbf{x}_t$, the previous hidden state $\mathbf{h}_{t-1}$, and the weight matrices $\mathbf{W_f}$ and $\mathbf{R_f}$, along with the bias term $\mathbf{b_f}$.

For sLSTM stabilized version, the stabilizer state m_t is introduced:

$$m_t = \max(\log(\mathbf{f}_t) + m_{t-1}, \log(\mathbf{i}_t)) \tag{13}$$

In this equation, m_t combines information from the logarithm of the forget gate $\log(\mathbf{f}_t)$ and the previous stabilizer state m_{t-1}, with a comparison against the logarithm of the input gate $\log(\mathbf{i}_t)$.

The stabilized versions of the input and forget gates, $\mathbf{i}'_t$ and $\mathbf{f}'_t$, are then computed as:

$$\mathbf{i}'_t = \exp(\log(\mathbf{i}_t) - m_t) \tag{14}$$

$$\mathbf{f}'_t = \exp(\log(\mathbf{f}_t) + m_{t-1} - m_t) \tag{15}$$

In above equations, the stabilizer m_t adjusts the input and forget gates to prevent instability during training.

Based on the proof of equivalence between the sLSTM and the stabilization version in (Beck et al. 2024), adding the stabilizer state does not alter the original hidden state. Since the loss function depends solely on the hidden state and has no dependency on the stabilizer state, there is consequently no gradient for the stabilization state. Therefore, the purpose of the stabilizer state is to stabilize the exponential function, without affecting the output or memory of the sLSTM model. Below, we illustrate this by examining the long-term behavior of the forget gate in the stabilized version and comparing it with the original forget gate.

Consider $t \to \infty$, we first examine the update equation for the stabilizer state m_t as showing in Equation (13). At each time step, m_t is determined by taking the maximum of two values: $\log(\mathbf{f}_t) + m_{t-1}$ and $\log(\mathbf{i}_t)$. If $\log(\mathbf{f}_t) + m_{t-1}$ is consistently larger, m_t will be primarily influenced by the forget gate. Otherwise, it will be driven by the input gate. Consider Equation (15) of stabilized forget gate, using the update for m_t, we can analyze two possible cases:

Case 1: $\log(\mathbf{f}_t) + m_{t-1} \geq \log(\mathbf{i}_t)$

In this case, Equation (13) can be written as

$$m_t = \log(\mathbf{f}_t) + m_{t-1}$$

Substituting this into the equation for $\mathbf{f}'_t$ gives:

$$\mathbf{f}'_t = \exp(\log(\mathbf{f}_t) + m_{t-1} - (\log(\mathbf{f}_t) + m_{t-1})) = \exp(0) = 1$$

The cell state now becomes:

$$\mathbf{c}_t = \mathbf{f}'_t \cdot \mathbf{c}_{t-1} + \mathbf{i}'_t \cdot \mathbf{z}_t = 1 \cdot \mathbf{c}_{t-1} + \mathbf{i}'_t \cdot \mathbf{z}_t$$

Thus, when the forget gate dominates, the stabilized forget gate $\mathbf{f}'_t$ converges to 1, effectively neutralizing the impact of the forget gate and ensuring that the cell state relies primarily on the input gate for updates.

Case 2: $\log(\mathbf{f}_t) + m_{t-1} < \log(\mathbf{i}_t)$

In this case, Equation (13) can be written as

$$m_t = \log(\mathbf{i}_t)$$

Substituting into the equation for $\mathbf{f}'_t$ gives:

$$\mathbf{f}'_t = \exp(\log(\mathbf{f}_t) + m_{t-1} - \log(\mathbf{i}_t)) = \exp(\log(\mathbf{f}_t) - \log(\mathbf{i}_t) + m_{t-1})$$

Assuming that over time, $\log(\mathbf{f}_t)$ and $\log(\mathbf{i}_t)$ reach a balance and that $\log(\mathbf{i}_t)$ is consistently larger, m_t will be primarily set by $\log(\mathbf{i}_t)$, leading to:

$$m_t \approx \log(\mathbf{i}_t)$$

Thus, the stabilized forget gate can be simplified to:

$$\mathbf{f}'_t \approx \exp(\log(\mathbf{f}_t) - \log(\mathbf{i}_t) + \log(\mathbf{i}_t)) = \exp(\log(\mathbf{f}_t)) = \mathbf{f}_t$$

In this scenario, the stabilized forget gate $\mathbf{f}'_t$ will reflect the behavior of the original forget gate $\mathbf{f}_t$, moderated by the stabilizer state.

Above Cases 1 and 2 demonstrate that the stabilizer state effectively controls unbounded growth while ensuring that it does not alter the memory or the behavior of the sLSTM model. For the stabilizer state m_t, specifically, when $\log(\mathbf{f}_t) + m_{t-1}$ dominates, the forget gate does not cause any forgetting. Instead, it allows the memory (the cell state) from the previous time step to carry forward to the next time step without any reduction. When $\log(\mathbf{i}_t)$ dominates, the forget gate reflects the original forget gate value $\mathbf{f}'_t \approx \mathbf{f}_t$. This aligns with the analysis in (Beck et al. 2024).

B.3. sLSTM Block

The sLSTM block structure (Beck et al. 2024) is illustrated in Figure 6. This block is integrated within a pre-LayerNorm residual structure, which helps stabilize training and enhances gradient flow throughout the network. Initially, the input data is optionally passed through a causal convolution with a window size of 4, followed by the application of a Swish activation function specifically to the input and forget gates. This convolutional step is crucial for capturing local dependencies in the input sequence.

The block then processes the input, forget, and output gates (represented as $\mathbf{i}$, $\mathbf{f}$, and $\mathbf{o}$ respectively) along with the cell update $\mathbf{z}$, using a block-diagonal linear layer. This layer features four diagonal blocks (or heads) each corresponding to one of the recurrent gate pre-activations from the previous hidden state. This multi-headed design enables the model to manage and mix multiple memory streams independently within each head, enhancing its ability to handle complex temporal dependencies.

Next, the resulting hidden state is passed through a GroupNorm layer, which applies head-wise LayerNorm independently to each of the four heads. The final stage involves up- and down-projection using a gated MLP, which includes a GeLU activation function and a projection factor of 4/3. This projection step aligns the number of parameters and optimizes the model's capacity for representation.

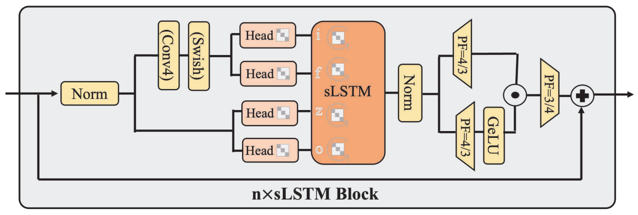

Figure 6: sLSTM Block Structure.

C. Experimental Details

C.1. Dataset

In our experiments, we utilize several datasets that have been employed in previous research (Liu et al. 2024):
ETTm1: A subset of the ETT dataset, ETTm1 contains 7 factors related to electricity consumption, recorded every 15 minutes between July 2016 and July 2018 (Liu et al. 2024).
PEMS03: This dataset is part of the PEMS collection, encompassing public traffic network data in California, with a recording interval of 6 minutes (Liu et al. 2024).

1. **Weather Dataset:** This includes 21 meteorological variables collected from the Weather Station of the Max Planck Biogeochemistry Institute, with the interval of 10 minutes (Liu et al. 2024).
2. **Electricity Dataset:** This records the electricity consumption of 321 consumers, with the interval of one hour (Liu et al. 2024).
3. **Solar Dataset:** This records the solar power production of 137 PV plants in 2006 with the interval of 10 minutes (Liu et al. 2024).
4. **ETTm1:** A subset of the ETT dataset, ETTm1 contains 7 factors related to electricity consumption, recorded every 15 minutes between July 2016 and July 2018 (Liu et al. 2024).
5. **PEMS03:** This dataset is part of the PEMS collection, which contains the public traffic network data in California, with the interval of 6 minutes (Liu et al. 2024).

In our experiments, we use some existing dataset that are used in previous research (Liu et al. 2024). 1. Weather dataset includes 21 meteorological variables collected from the Weather Station of the Max Planck Biogeochemistry Institute, with the interval of 10 minutes (Liu et al. 2024).. 2. Electricity dataset records the electricity consumption of 321 comsumers, with the interval of one hour (Liu et al. 2024). 3. Solar records the solar power production of 137 PV plants in 2006 with the interval of 10 minutes (Liu et al. 2024).. 4. ETTm1 is a part of ETT, contains 7 factors of electricity from July 2016 to July 2018, with the interval of 15 minutes (Liu et al. 2024).. 4. PEMS03 is a part of PEMS, which contains the public traffic network data in California, with the interval of 6 minutes (Liu et al. 2024).

C.2. Implementation and Hyper-parameters

Our experiments were conducted using the Time Series Library (https://github.com/thuml/Time-Series-Library). We employed mean squared error as our loss function and used the Adam optimizer with an initial learning rate of 1e-3.

In Table 6-10, we list some of the hyper-parameters that can reproduce our experimental results.

For hyper-parameter tuning, we explored multiple combinations, as detailed in Table 11.

Table 6: Hyper-parameters of Weather dataset following the order of prediction length (96, 192, 336, 720) and 336 look-back window.

Hyper-parameters	Value
Batch Size	16/16/16/16
Patch Size	56/56/56/56
Stride	56/56/56/56
Number of sLSTM blocks	1/1/1/1
Embedding Dimension of Projection Layer	100/100/100/100
Number of Heads of Memory Mixing	2/2/2/2
Convolution size in sLSTM block	8/8/8/8
Dropout Rate	0.1/0.1/0.1/0.1

Table 7: Hyper-parameters of Electricity dataset following the order of prediction length (96, 192, 336, 720) and 336 look-back window.

Hyper-parameters	Value
Batch Size	16/16/16/16
Patch Size	56/16/16/16
Stride	56/16/16/16
Number of sLSTM blocks	2/1/1/1
Embedding Dimension of Projection Layer	600/600/600/600
Number of Heads of Memory Mixing	3/3/3/3
Convolution size in sLSTM block	32/32/32/32
Dropout Rate	0.1/0.1/0.1/0.1

Table 8: Hyper-parameters of Solar dataset following the order of prediction length (96, 192, 336, 720) and 336 look-back window.

Hyper-parameters	Value
Batch Size	32/32/32/32
Patch Size	16/16/16/16
Stride	16/16/16/16
Number of sLSTM blocks	1/1/1/1
Embedding Dimension of Projection Layer	100/100/100/100
Number of Heads of Memory Mixing	2/2/2/2
Convolution size in sLSTM block	4/4/4/4
Dropout Rate	0/0/0/0

D. Additional Analysis

Comparison between P-sLSTM and PatchTST As shown in Table 12, P-sLSTM consistently outperforms

Table 9: Hyper-parameters of ETTm1 dataset following the order of prediction length (96, 192, 336, 720) and 336 look-back window.

Hyper-parameters	Value
Batch Size	32/32/32/32
Patch Size	6/6/6/6
Stride	6/6/6/6
Number of sLSTM blocks	1/1/1/1
Embedding Dimension of Projection Layer	100/100/100/100
Number of Heads of Memory Mixing	2/4/4/4
Convolution size in sLSTM block	32/4/4/2
Dropout Rate	0.1/0/0/0

Table 10: Hyper-parameters of PEMS03 dataset following the order of prediction length (12, 24, 48, 96) and 96 look-back window.

Hyper-parameters	Value
Batch Size	32/32/32/32
Patch Size	16/16/16/16
Stride	16/16/16/16
Number of sLSTM blocks	2/2/2/2
Embedding Dimension of Projection Layer	300/300/300/300
Number of Heads of Memory Mixing	6/6/6/6
Convolution size in sLSTM block	8/8/8/8
Dropout Rate	0.1/0.1/0.1/0.1

PatchTST on the Solar and PEMS03 datasets and performs comparably on Electricity dataset. PatchTST results are from (Liu et al. 2024).

Comparison between LSTM and P-LSTM As shown in Table 13, Patching can improve the performance of LSTM; however, P-sLSTM is still perform much better than P-LSTM, indicating the effectiveness of exponential gating and memory mixing.

Ablation Study of Stabilizer State As shown in Table 14, there is minimal difference between the results with and without the stabilizer state, with the difference being around 0.001-0.002, which is negligible. These results align with our analysis of the stabilizer version of sLSTM, where using the stabilizer state yields a new forget gate equivalent to the original forget gate.

Forecasting Results of Electricity Dataset As illustrated in Figures 7 through 10, P-sLSTM effectively predicts relatively stationary and periodic time series. Despite some fluctuations in the data, as shown in the upper left panel of Figure 7, P-sLSTM successfully captures sudden changes and aligns well with the overall data trend. However, when the data exhibits more severe and drastic fluctuations, P-sLSTM struggles to predict sudden rises, sudden drops, and extreme volatility. This is one of our future research directions.

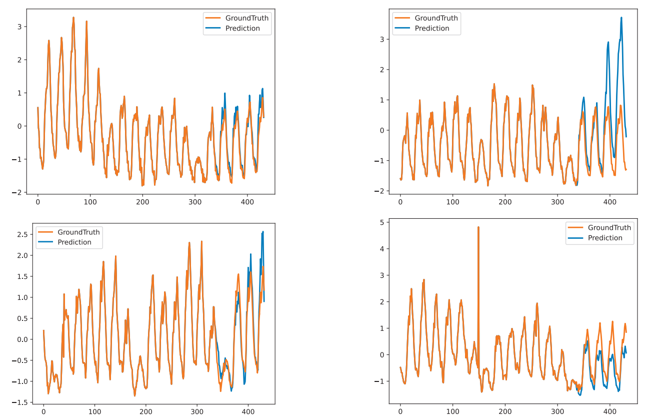

Figure 7: The forecasting of a few time series from the Electricity dataset run with P-sLSTM (Input Length = 336, Forecasting Length = 96). Prediction: blue; Actual data: orange.

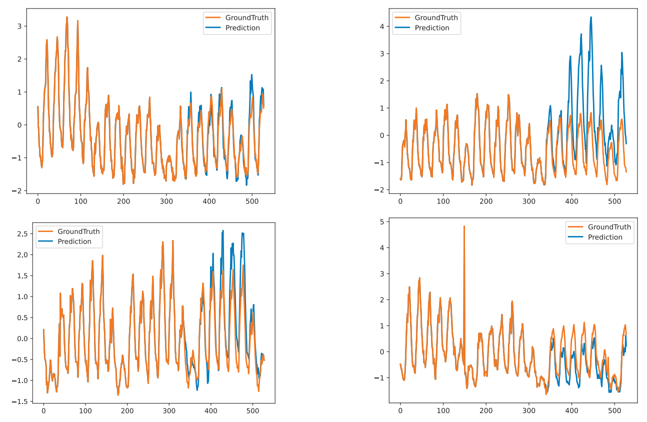

Figure 8: The forecasting of a few time series from the Electricity dataset run with P-sLSTM (Input Length = 336, Forecasting Length = 192). Prediction: blue; Actual data: orange.

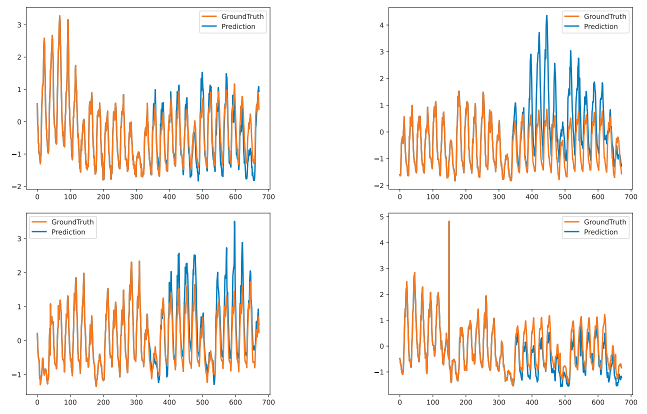

Figure 9: The forecasting of a few time series from the Electricity dataset run with P-sLSTM (Input Length = 336, Forecasting Length = 336). Prediction: blue; Actual data: orange.

Table 11: Hyper-parameters Searching of our experiments.

Hyper-parameters	Values/Range
Batch Size	4, 8, 16, 32, 64
Patch size	6, 16, 26, 36, 46, 56, 66, 76, 86
Stride	6, 16, 26, 36, 46, 56, 66, 76, 86
Number of sSLTM blocks	1, 2, 3
Embedding Dimension of Projection Layer	8, 16, 32, 64, 100, 128, 300, 600
Number of Heads of Memory Mixing	1, 2, 3
Convolution size in xLSTM block	2, 4, 8, 16, 32, 64
Dropout Rate	0, 0.1, 0.2

Table 12: Comparison between P-sLSTM and PatchTST. Values represent MSE.

Dataset	**P-sLSTM**	**PatchTST**
Electricity	0.130	0.130
	0.148	0.148
	0.165	0.167
	0.199	0.202
Solar	0.167	0.191
	0.180	0.208
	0.190	0.209
	0.196	0.222
PEMS03	0.077	0.099
	0.109	0.142
	0.163	0.176
	0.209	0.269

Table 13: Comparison between LSTM, P-LSTM and P-sLSTM. Values represent MSE.

Dataset	**LSTM**	**P-LSTM**	**P-sLSTM**
Electricity	0.350	0.252	0.130
	0.354	0.253	0.148
	0.357	0.270	0.165
	0.368	0.303	0.199
ETTm1	0.771	0.308	0.292
	0.820	0.344	0.329
	0.927	0.378	0.362
	0.960	0.423	0.421

Table 14: Ablation Study of Stabilizer State (S.S.) on Weather dataset. Values represent MSE/MAE.

Length	Accuracy with S.S.	Accuracy without S.S.
96	0.149 / 0.208	0.150 / 0.209
192	0.197 / 0.256	0.197 / 0.257
336	0.249 / 0.297	0.251 / 0.298
720	0.320 / 0.350	0.322 / 0.352

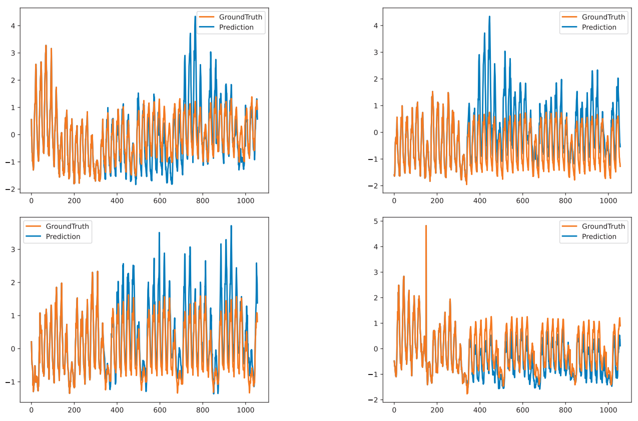

Figure 10: The forecasting of a few time series from the Electricity dataset run with P-sLSTM (Input Length = 336, Forecasting Length = 720). Prediction: blue; Actual data: orange.