Enhancing CycleNet for Time Series Forecasting

Amin Darabi [* 1 2] **Chirag Goel** [* 1 2] **Aayush Bajaj** [* 1 2] **Milosh Devic** [* 1 2]

Abstract

Accurate long-term time series forecasting (LTSF) is critical in various domains, such as energy management, weather prediction, and transportation. CycleNet, a state-of-the-art model for LTSF, effectively captures periodicity using its Residual Cycle Forecasting (RCF) technique. However, its reliance on a Multi-Layer Perceptron (MLP) backbone limits its ability to model inter-feature relationships in multivariate datasets. Additionally, the lack of uncertainty quantification restricts its interpretability, which is essential for practical applications. In this work, we address these limitations by introducing two key modifications to CycleNet. First, we replace the MLP backbone with Time-GNN to better capture spatio-temporal dependencies between features. Second, we enhance the model with probabilistic forecasting, enabling it to output distributions rather than point predictions, thus providing confidence intervals for improved interpretability and decision-making. Our experiments demonstrate improvements and comparable results across several datasets, with Time-GNN enhancing feature dependency modeling and probabilistic forecasting improving both accuracy and interpretability.

1. Introduction

LTSF plays an important role in various domains, such as energy management, weather prediction, and transportation systems. Accurate forecasting in these areas enables better decision-making, resource allocation, and system optimization by anticipating future patterns and anomalies.

CycleNet (Lin et al., 2024) is a recent state-of-the-art model for LTSF that excels in modeling periodic patterns within time series data. It introduces the RCF technique, which explicitly captures periodic components through learnable recurrent cycles and predicts residual components for improved accuracy. For instance, as shown in Figure 1, taken from the original CycleNet paper, electricity consumption data demonstrates clear daily periodic patterns. By learning and replicating a shared daily segment, CycleNet effectively models these cyclic components, enabling efficient and accurate long-horizon predictions.

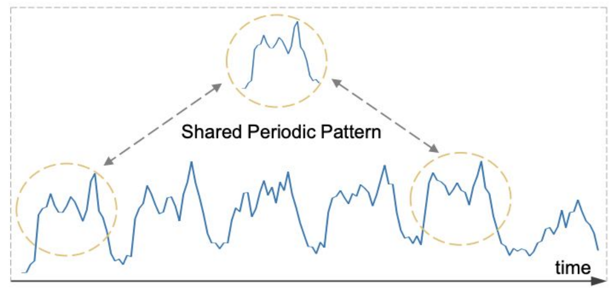

Figure 1. Shared daily periodic patterns present in the Electricity dataset.

It has shown great performance across several datasets, including ETT series (Zhou et al., 2021), Weather, Traffic, Electricity, and Solar-Energy (Lai et al., 2018). It uses the RCF technique to model periodicity explicitly by learning recurrent cycles and predicting residual components. This simple yet powerful method achieved state-of-the-art accuracy while maintaining a lightweight architecture, making it highly efficient and versatile for many applications.

Despite its effectiveness, CycleNet assumes independence between features (or channels), which can limit its performance in scenarios where inter-feature relationships are crucial. For example, datasets that exhibit strong dependencies across spatial or correlated features may require architectures capable of modeling these relationships explicitly. Addressing this limitation presents an opportunity to enhance CycleNet's performance by incorporating techniques that better capture the interactions between features in multivariate time-series data.

2. Related Works

The use of periodic information to enhance time series forecasting is well established in the literature. One prominent category of methods focuses on Seasonal-Trend Decomposition (STD), which aims to separate the original time

[*]Equal contribution [1]DIRO, Université de Montréal, Montréal, Canada [2]MILA Institute, Montréal, Canada.

series into components that capture periodic (seasonal) and long-term (trend) behaviors. Popular models such as Autoformer (Wu et al., 2022), FEDformer (Zhou et al., 2022), and DLinear (Zeng et al., 2022) rely on the classical STD approach, where a basic moving average (MOV) kernel is applied to isolate the trend component. This decomposition allows models to focus separately on distinct features of the time series, improving overall forecasting accuracy.

Recent developments have advanced beyond classical STD-based methods to address their limitations. Leddam (Leddam et al., 2023) introduced a Learnable Decomposition (LD) kernel to replace the traditional MOV kernel, enabling models to adaptively learn periodic structures from data. Similarly, DEPTS (Fan et al., 2022) models periodicity as a parameterized function of time, learning both periodic and residual components in a layer-wise manner using periodic and local blocks. SparseTSF (Liu et al., 2022) further innovates with cross-period sparse forecasting, efficiently decoupling cycles and trends at a significantly reduced computational cost.

Despite their successes, these methods assume fixed or uniform periodicity, which limits their effectiveness in dynamic or multivariate datasets such as traffic data. Traffic forecasting, in particular, presents unique challenges due to its inherently complex spatio-temporal dependencies, where relationships between features (e.g., road network topology or traffic flow) play a pivotal role. Classical STD methods and even modern approaches like SparseTSF struggle to capture these inter-feature relationships.

CycleNet addressed some of these shortcomings with its RCF technique. RCF models global periodic patterns explicitly through learnable recurrent cycles, enabling the separation of periodic and residual components without relying on external decomposition methods. Combined with a simple backbone like a Linear layer or MLP, CycleNet achieves state-of-the-art results across datasets with strong periodicity, such as Weather, ETT, and Electricity.

However, CycleNet's reliance on an MLP backbone limits its ability to capture inter-feature relationships, which are critical for traffic forecasting tasks. This limitation aligns with observations in other works that emphasize the need for models capable of handling both temporal and spatial correlations effectively. GNNs have emerged as powerful tools for modeling multivariate and spatio-temporal dependencies in structured data, such as traffic flow or sensor networks (Wu et al., 2020; Li et al., 2018). Methods like Time-GNN (Zhang et al., 2020) and Spatio-Temporal GNNs (Yu et al., 2018) explicitly capture feature relationships, making them well-suited for traffic forecasting applications.

Additionally, traditional LTSF methods often provide point forecasts, which lack uncertainty quantification. Recent works have explored probabilistic forecasting approaches to provide distributional predictions, improving interpretability and robustness, particularly in domains like traffic where variability and uncertainty are significant (Salinas et al., 2020; Wen et al., 2017).

Our proposed modifications build on these insights to address CycleNet's shortcomings in traffic forecasting. By replacing the MLP backbone with a GNN, we aim to model the inter-feature relationships explicitly. Additionally, by introducing probabilistic forecasting, we extend CycleNet's capabilities to provide more meaningful predictions with uncertainty estimates, enhancing its applicability to real-world scenarios where interpretability is crucial.

This work aligns with current trends in LTSF that emphasize the importance of balancing accuracy, efficiency, and interpretability, particularly for complex datasets like traffic.

3. CycleNet

CycleNet is a state-of-the-art model for LTSF that explicitly focuses on modeling periodic patterns in time series data. It is designed to leverage the inherent regularities in datasets, such as daily or weekly cycles, to improve forecasting accuracy while maintaining computational efficiency. The model's key innovation lies in its RCF technique. RCF decomposes the input data into periodic and residual components which allows each to be modeled separately.

The RCF technique operates in two stages:

1. **Periodic Pattern Modeling**: RCF learns periodic patterns from historical data using a learnable recurrent cycle. These cycles are represented as parameters in the model, and are optimized during training to reflect the most stable periodic patterns in the dataset. Once learned, the cyclic components are removed from the input data so, we are left with the residual components that represent the non-periodic variations.

2. **Residual Forecasting**: After removing the periodic components, the residual components are fed into a backbone model, which can be either a simple Linear layer or a lightweight MLP. The backbone model focuses on predicting the residual values. This separation of periodic and residual components simplifies the forecasting task, as it allows the model to focus on only one type of variation at a time.

The final forecast is obtained by adding the predicted residuals back to the cyclic components, ensuring that the periodicity is retained in the prediction. CycleNet has demonstrated state-of-the-art performance in datasets with strong periodicity thanks to the way it explicitly models cycles and its lightweight design. Additionally, given the simplicity of the

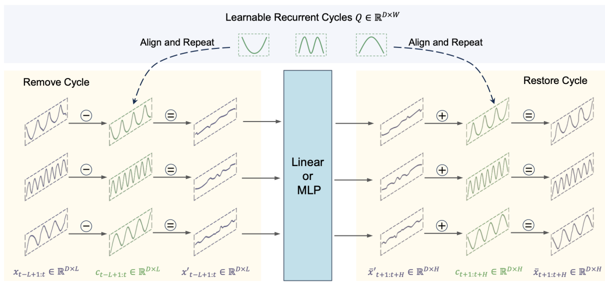

Figure 2. CycleNet Architecture. CycleNet/Linear and CycleNet/MLP represent using a single-layer Linear model and a dual-layer MLP model, respectively, as the backbone of CycleNet. Here, D = 3.

backbone, it can achieve these results with a low computational cost. The architecture of this model can be seen in Figure 2, taken from CycleNet's paper.

However, there are some limitations to CycleNet which motivated the modifications we are proposing. It assumes that features (channels) are independent, which limits its applicability to datasets where inter-feature relationships are crucial. This can lead to suboptimal results on those datasets. Furthermore, CycleNet provides only point predictions and doesn't quantify uncertainty, which limits its interpretability.

4. Motivation

Despite CycleNet's success across various datasets, certain limitations arise due to its current architectural design. Specifically, CycleNet's backbone, a Multi-Layer Perceptron (MLP), assumes independence between features (or channels), as mentioned before. This restricts its ability to model inter-channel relationships. In many multivariate time-series datasets, capturing dependencies between features is crucial to accurately predict future trends. For example, in datasets with strong spatial or correlated relationships between variables, explicitly modeling these dependencies could significantly improve forecasting performance.

In the original CycleNet paper, the authors note that directly applying the RCF technique to other models, such as iTransformer, does not yield significant improvements. They suggest exploring multivariate modeling approaches that enhance CycleNet's ability to capture inter-feature relationships. GNNs are a promising approach due to their ability to explicitly model relationships between variables in a structured and dynamic way. Inspired by this, we explore modifications to CycleNet to address these challenges.

Our investigation focuses on two main directions:

1. **Feature Relationships**: Replacing CycleNet's MLP backbone with a GNN, more specifically Time-GNN (Zhang et al., 2020), to explicitly model the spatio-temporal dependencies between features.
2. **Interpretability**: Predicting the distribution of traffic values (mean and standard deviation) could provide more meaningful insights into how trends might evolve and predicting more likely outcomes. This part doesn't inherently rely on GNNs so we kept the MLP as the backbone for this experiment.

Through these modifications, we aim to enhance CycleNet's ability to handle the complexity of multivariate time-series datasets and provide interpretable and robust forecasts. In this report, we present our methodology, experimental results, and conclusions on the effectiveness of these enhancements across various datasets.

5. Modifications

To address the aforementioned limitations, we introduce two key modifications to CycleNet: (1) replacing the MLP backbone with a GNN to model feature relationships, and (2) extending the model to provide probabilistic forecasts for enhanced interpretability. Before doing any modifications, we first reproduced the results on some of the datasets to validate them. These baseline results can be see in Figure **??**.

5.1. Replacing MLP with GNN

CycleNet's original backbone cannot capture inter-channel dependencies. GNNs, on the other hand, are well-suited for this task since they can explicitly model relationships between features in a structured graph format.

We replaced the MLP backbone in CycleNet with a GNN (TimeGNN) to model the inter-feature relationships in the data. TimeGNN is a very computationally efficient GNN whcih works well for long range predictions therefore, we would be able to maintain one of the key features of CycleNet, which is to be efficient. Furthermore, it scales well. TimeGNN learns a dynamic graph structure for each sliding window of the time series data. In other words, it learns a new adjacency matrix for each sliding window, enabling it to model evolving relationships between channels over time. This can better handle multivariate dependencies, especially in datasets with complex inter-variable relationships. It uses GraphSAGE over the learned graph structure to aggregate information across neighboring nodes, capturing spatio-temporal dynamics and interdependencies over time.

The implementation is carried out in several steps. Each feature is represented as a node in the graph, and the edges capture spatial relationships. After the periodic components are removed by the RCF, the residual components are passed through TimeGNN, which utilizes its temporal and spatial modeling capabilities to generate forecasts. An illustration

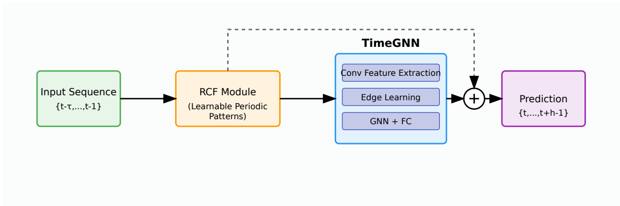

Figure 3. CycleNet/TimeGNN Architecture

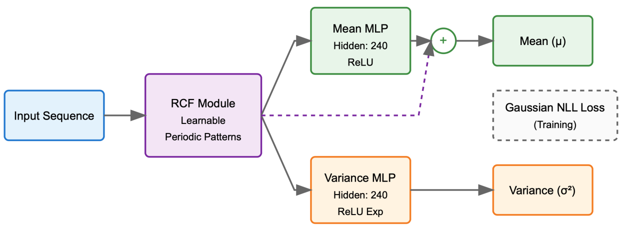

Figure 4. Probabilistic CycleNet Architecture

of this framework is shown in Figure 3.

5.2. Probabilistic Forecasting

Knowing the range of possible outcomes can be as important, or perhaps even more important than the forecast itself. Point predictions do not capture the inherent uncertainty in the datasets. By predicting distributions, the model can provide confidence intervals, enhancing its interpretability and its robustness.

This allows it to predict the full distribution of possible outcomes. Our approach assumes that the target values follow a Gaussian distribution, where both the mean and variance are functions of the input sequence. We modified the MLP layer of the model to output two values for each time step. We used Gaussian Negative Log-Likelihood (GaussianNLL) loss (Nix & Weigend, 1994) for training instead of the standard Mean Squared Error (MSE). This update allows the model to learn predictive uncertainty without sacrificing its efficient architecture.

5.2.1. Probabilistic Framework

Given a time series X with D variables and L historical observations, our goal is to predict not only the expected values of the next H time steps but also their associated uncertainties. We model the conditional distribution of future values as:

$$p(x_{t+1:t+H}|x_{t-L+1:t}) = \mathcal{N}(\mu_{t+1:t+H}, \sigma^2_{t+1:t+H})$$

where $\mu_{t+1:t+H}$ represents the predicted means and $\sigma^2_{t+1:t+H}$ represents the predicted variances for the future H time steps.

5.2.2. Dual MLP Architecture

Our extended architecture maintains CycleNet's RCF technique but incorporates two parallel MLPs:

1. Mean MLP (f_μ): Predicts the expected values, following the original CycleNet design 2. Variance MLP (f_σ): Estimates the variance of predictions

Both MLPs process the same residual components after cycle removal:

$$x'_{t-L+1:t} = x_{t-L+1:t} - c_{t-L+1:t}$$

where $c_{t-L+1:t}$ represents the cyclic components extracted by RCF. The MLPs then produce:

$$\mu_{t+1:t+H} = f_\mu(x'_{t-L+1:t}) + c_{t+1:t+H}$$
$$\log(\sigma^2_{t+1:t+H}) = f_\sigma(x'_{t-L+1:t})$$

Note that we predict the logarithm of the variance to ensure positivity without additional constraints.

5.2.3. Gaussian Negative Log-Likelihood Loss

We train the model by minimizing the Gaussian Negative Log-Likelihood (NLL) loss:

$$\mathcal{L} = \frac{1}{2H}\sum_{h=1}^{H}\left(\frac{(x_{t+h} - \mu_{t+h})^2}{\sigma^2_{t+h}} + \log(\sigma^2_{t+h})\right)$$

This loss function naturally balances the precision of predictions with their uncertainty estimates. It penalizes both overconfident and underconfident predictions, encouraging the model to produce well-calibrated uncertainty estimates.

5.2.4. Training Procedure

The training process consists of the following steps: First, initialize the recurrent cycles Q to absolute zero and set the parameters of the MLPs randomly close to zero. The model will then train to minimize the loss using the Adam optimizer with a very small learning rate. This helps ensure stable convergence and prevents the variance in predictions from interfering with the learning of the mean predictions during the early stages of training.

6. Training Setup

6.1. Replacing MLP with GNN

For our experiments, we replaced CycleNet's original MLP backbone with TimeGNN to better capture spatio-temporal dependencies between features. Below are the details of the training setup:

- **Datasets:** We conducted experiments on benchmark LTSF datasets including ETTh1, ETTh2, ETTm1, ETTm2, and Electricity. Each dataset was preprocessed to standardize features and split into training, validation, and test sets.

- **Model Architecture:** The modified CycleNet integrates the RCF module with TimeGNN as the backbone.
- **Loss Function:** MSE was used as the primary loss function to minimize the difference between predicted and actual values across future time steps.
- **Optimizer:** The Adam optimizer was used for training, with default settings for learning rate and momentum. A learning rate scheduler was applied to dynamically adjust the learning rate during training.
- **Batch Size:** A batch size of 32 was used for all experiments to balance memory efficiency and convergence speed.
- **Early Stopping:** Early stopping was employed based on the validation loss to prevent overfitting and ensure robust performance.
- **Evaluation Metrics:** We used MAE and MSE to evaluate model performance on the test sets. All metrics were averaged over multiple runs to ensure consistency.
- **Implementation Details:** The model was implemented using PyTorch. Each experiment was run for a maximum of 50 epochs, with training halted earlier if the validation loss plateaued.

6.2. Probabilistic Forecasting

6.2.1. Datasets

We evaluate our probabilistic extension of CycleNet on four benchmark datasets from the ETT (Electricity Transformer Temperature) family: ETTm1, ETTm2, ETTh1, and ETTh2. These datasets contain multi-variate time series data collected from electricity transformers at different sampling frequencies (minute and hour level). Each dataset includes seven features: oil temperature, load, and various external factors that influence transformer behavior.

The datasets are processed using a sliding window approach with the following specifications:

- Input sequence length: 720 time steps
- Prediction horizon: 240 time steps
- Training validation/test split: 80%20%
- All data will scale with the mean and std of the training set over different channels before the training

6.2.2. Model Architecture

Our probabilistic CycleNet implementation maintains the core Residual Cycle Forecasting (RCF) technique while extending it with dual MLP networks:

1. Mean Prediction Network:

- Input layer: Matches input sequence dimensions
- Hidden layer: 240 units with ReLU activation
- Output layer: 240 units (matching prediction horizon)

2. Variance Prediction Network:

- Parallel architecture to mean network
- Input layer: Matches input sequence dimensions
- Hidden layer: 240 units with ReLU activation
- Output layer: 240 units with exponential activation to ensure positive variance

The RCF component uses learnable periodic patterns that are shared between both networks, maintaining CycleNet's efficient parameter usage while enabling uncertainty estimation.

6.3. Training Procedure

The model is trained using the following configuration:

- Loss function: Gaussian Negative Log-Likelihood (NLL)
- Optimizer: Adam with learning rate 0.0001
- Batch size: 200
- Maximum epochs: 100 with early stopping

7. Results

The proposed modifications to CycleNet demonstrate some improvements and comparable results across various datasets. As shown in Table 1, the CycleNet/MLP-Distribution achieves comparable or better MSE performance than the standard CycleNet/MLP while introducing predictive variance. This probabilistic extension offers more confidence in predictions, which is particularly visible in the ETTm2 dataset as CycleNet/MLP-Distribution outperforms the original untouched model.

In terms of MAE performance (Table 2), replacing the MLP backbone with TimeGNN proves to be advantageous, as it better captures spatio-temporal relationships. This is evident in datasets like Electricity, where CycleNet+TimeGNN

Dataset	ETTm1	ETTm2	ETTh1	ETTh2
CycleNet/MLP	0.376	0.141	0.475	0.211
CycleNet/MLP-Distribution	0.376	0.136	0.460	0.199

Table 1. Mean MSE across 7 channels for 720→240 step prediction. The probabilistic extension maintains or improves MSE performance while providing variance in our prediction.

Dataset	ETTh1	ETTh2	ETTm1	ETTm2	Electricity
CycleNet/MLP	0.441	0.409	0.396	0.314	0.259
TimeGNN original	-	-	-	-	0.309
CycleNet/TimeGNN	0.459	0.391	0.379	0.280	0.269

Table 2. Mean MAE across different channels on different datasets.

reduces the MAE significantly compared to the original TimeGNN.

Figures 6 and 7 further highlight the effectiveness of the proposed approaches. In Figure 6, the CycleNet/MLP-Distribution accurately predicts the time series data for ETTm1 and ETTm2 channels, with variance bands indicating higher confidence in early time steps. Figure 7 showcases the predictions of CycleNet+TimeGNN on ETTm1, demonstrating its ability to track spatio-temporal dynamics closely and align with the ground truth.

8. Conclusion

Predicting the distribution proves to be beneficial in providing more accurate estimates, as demonstrated by the performance of CycleNet/MLP-Distribution. Furthermore, replacing the MLP backbone with TimeGNN leads to performance improvements by effectively modeling spatio-temporal relationships between features. Notably, the performance of CycleNet+TimeGNN on the electricity dataset highlights the effectiveness of the RCF technique, yielding significant gains over the original TimeGNN architecture.

References

Fan, X., Liu, Z., Xiong, J., and Wang, Y. Depts: Deep periodic and residual forecasting for time series, 2022.

Lai, G., Chang, W.-C., Yang, Y., and Liu, H. Modeling long- and short-term temporal patterns with deep neural networks, 2018.

Leddam, S., Zhang, R., and Jiang, Z. Learnable decomposition for seasonal-trend forecasting, 2023.

Li, Y., Yu, R., Shahabi, C., and Liu, Y. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, 2018.

Lin, S., Lin, W., Hu, X., Wu, W., Mo, R., and Zhong, H. Cyclenet: Enhancing time series forecasting through modeling periodic patterns, 2024.

Liu, Y., Chen, J., and Zhu, M. Sparsetsf: Cross-period sparse forecasting for time series, 2022.

Nix, D. and Weigend, A. Estimating the mean and variance of the target probability distribution. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, pp. 55–60 vol.1. IEEE, 1994.

Salinas, D., Flunkert, V., Gasthaus, J., and Januschowski, T. Deepar: Probabilistic forecasting with autoregressive recurrent networks, 2020.

Wen, Q., Xu, J., Wang, J., and Zhang, L. S. Multi-horizon time series forecasting with temporal attention learning, 2017.

Wu, H., Xu, J., Wang, J., and Long, M. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting, 2022. URL `https://arxiv.org/abs/2106.13008`.

Wu, Y., Green, B., and Liu, Y. Connecting the dots: Multivariate time series forecasting with graph neural networks, 2020.

Yu, B., Yin, H., and Zhu, Z. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting, 2018.

Zeng, A., Li, X., Wen, Q., Zhou, T., and Zhang, L. S. Are transformers effective for time series forecasting?, 2022.

Zhang, J., Zheng, Y., and Qi, D. Spatio-temporal graph neural networks for traffic flow forecasting, 2020.

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., and Zhang, W. Informer: Beyond efficient transformer for long sequence time-series forecasting, 2021.

Zhou, T., Ma, Z., Wen, Q., Jin, R., Yang, W., Zha, H., and Sun, L. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting, 2022.

A. Reproduced Results

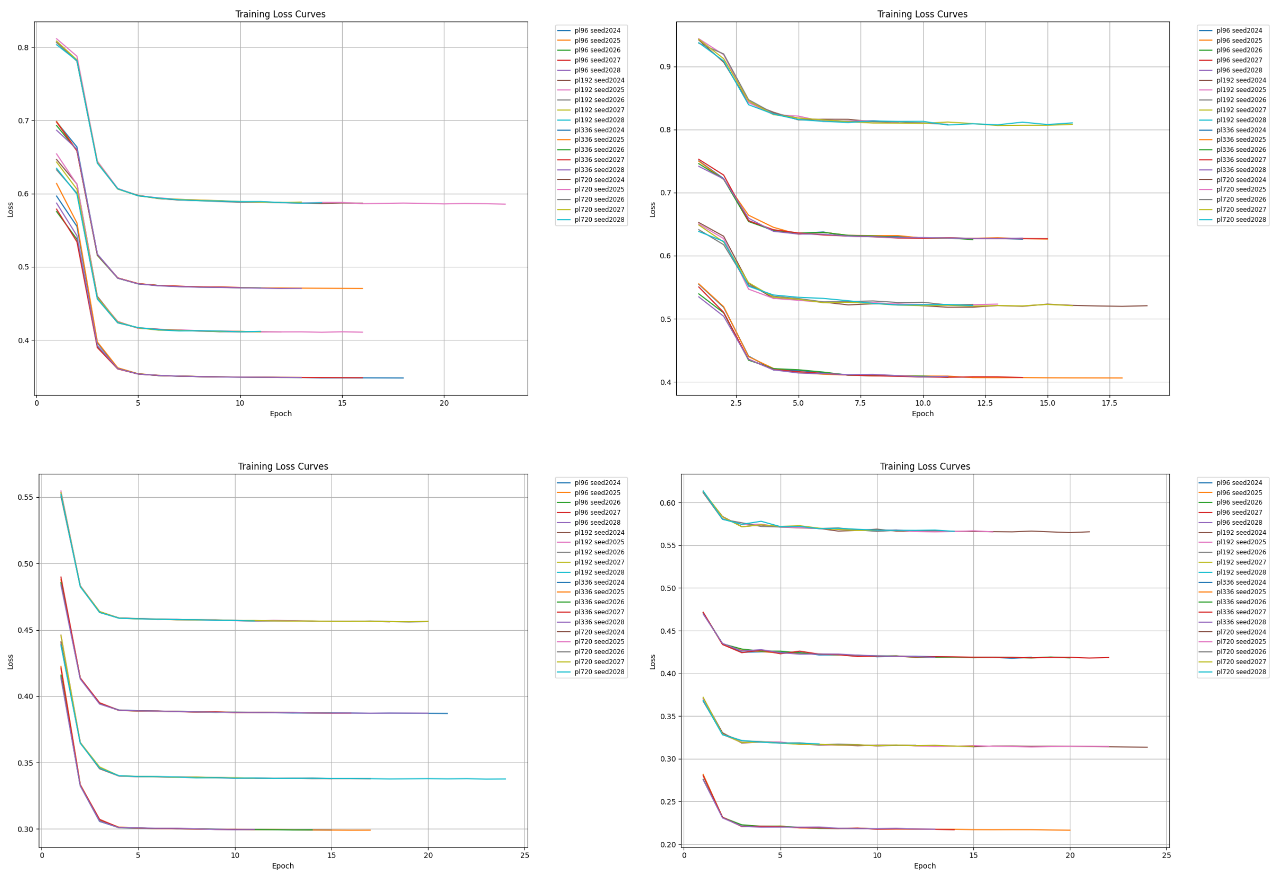

Figure 5. (a) Training Loss Curves for ETTH1 (b) Training Loss Curves for ETTH2 (c) Training Loss Curves for ETTM1 (d) Training Loss Curves for ETTM2

B. Predictions on ETTM1

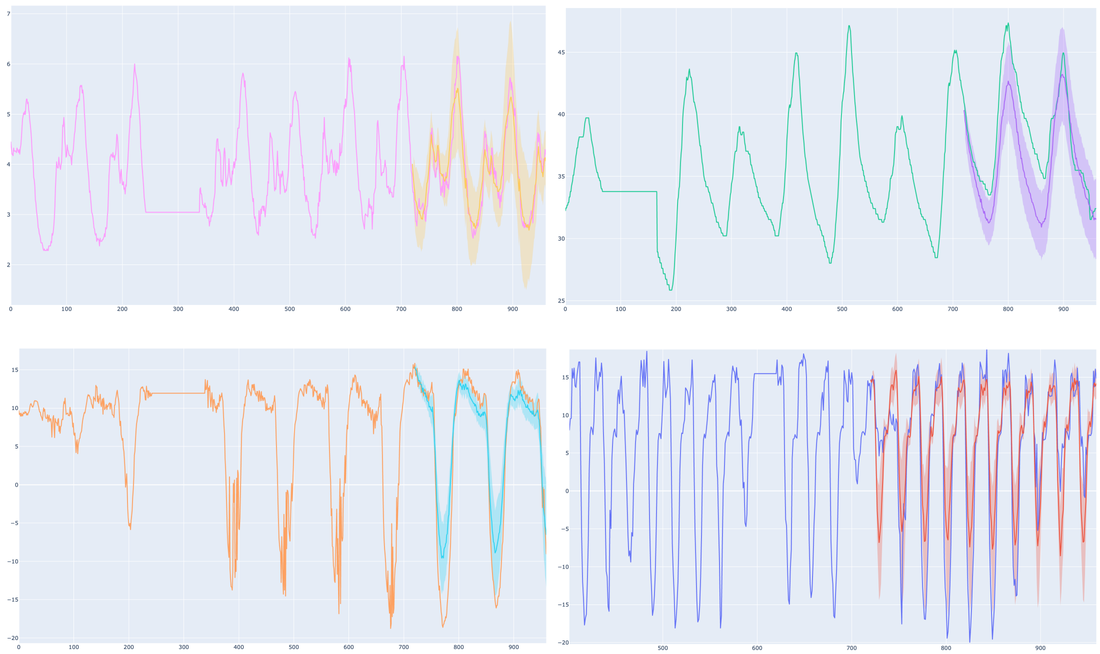

Figure 6. **CycleNet/MLP-Distribution**: (a) The prediction of ETTm1 channel 5. The band represents the root square of our predicted variance. (b) The prediction of ETTm2 channel 7. The model predicts early upcoming time steps with higher confidence. (c) The prediction of ETTm1 channel 3. (d) The prediction of ETTh1 channel 1.

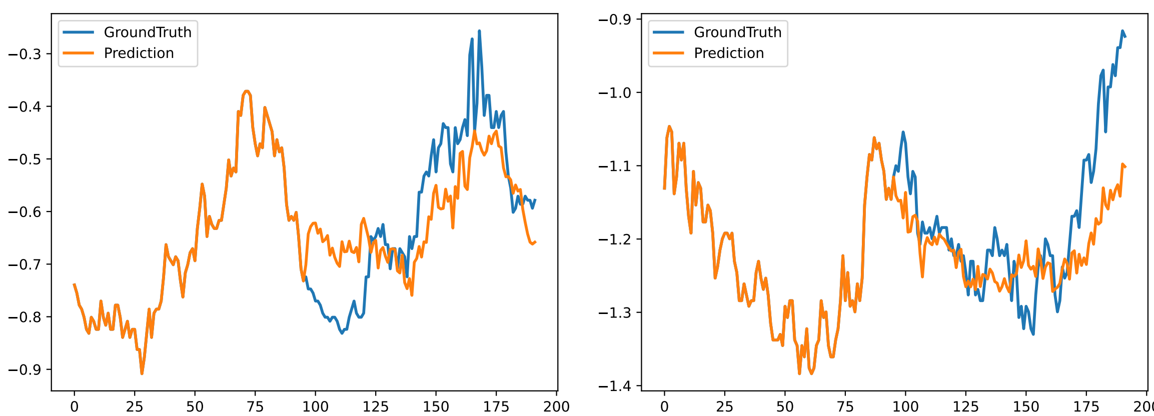

Figure 7. **CycleNet/TimeGNN**: The prediction of ETTm1 channel 7 of 2 (respectively) randomly selected data points.