COMP4434 Big Data Analytics Project

**CycleNetQM: Quality-Managed Cyclic Neural Network**

CHAI Wenchang 22096793d

DING Honghe 22099687d

GUO Beichen 22103456d

SHEN Zitong 22097304d

# Introduction

Time series forecasting, especially long-term time series forecasting (LTSF), is one of the fundamental problems in data analysis and prediction [2,3,4,5]. While CycleNet [1] has shown promising results with an innovative Residual Cycle Forecasting (RCF) framework [6], our analysis points out two key limitations: the insufficient effect of the extracted periodic pattern and training instability in the joint optimization of cyclic components and backbone networks, which further limits its potential to achieve better prediction accuracy.

To cope with these issues, we present CycleNetQM, a strengthening of the basic ideas of CycleNet with novel mechanisms that imbue patterns with more meaning and regularize training dynamics further. Our improvement on periodic pattern extraction also concentrates on building more stable training strategies so that overall better performance results across various circumstances. These improvements validate the effectiveness of our proposed enhancements in addressing the identified limitations of the original CycleNet architecture.

The implementation of CycleNetQM can be found at <https://github.com/CCMKCCMK/COMP4434Project>.

# Background

The paper *CycleNet: Enhancing Time Series Forecasting through Modeling Periodic Patterns* [1] addresses the challenge of Long-term Time Series Forecasting (LTSF) [3]by focusing on explicit periodic pattern modeling. The authors identify that while current LTSF methods can capture periodic patterns, they rely on complex architectures to do so indirectly, which may not be the most effective approach [1].

Since inherent periodicity plays a crucial role in long-term predictions, serving as the foundation for accurate forecasting, yet current approaches rely on complex models, such as the transformer [10], to extract periodic features through deep architectures, which may be unnecessarily complicated, a simpler, more direct approach to modeling periodic patterns could potentially be more effective and efficient.

The authors introduce a novel approach called CycleNet [1], which consists of two main components:

1. The Residual Cycle Forecasting (RCF) technique uses learnable recurrent cycles to explicitly model inherent periodic patterns and performs predictions on the residual components of the modeled cycles.
2. The Backbone Network employs either a single-layer Linear or a dual-layer MLP and is combined with RCF to form the complete CycleNet architecture.

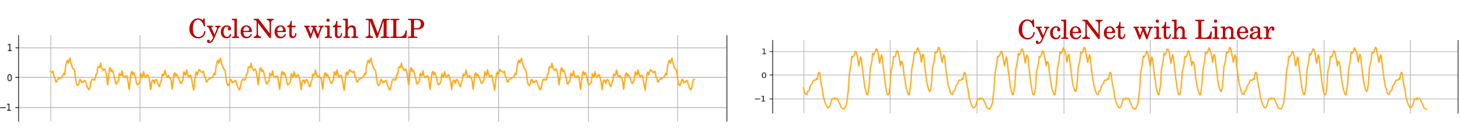
The findings in CycleNet pioneered explicit modeling of periodicity in time series data, introduced a plug-and-play RCF technique that can enhance existing models, and achieved state-of-the-art accuracy while reducing model parameters by over 90% [1].

However, the study also identifies key challenges. First, the significance of periodic patterns may vary, as input and residual data can sometimes be similar, requiring mechanisms to amplify the modeled pattern's impact.

图表

描述已自动生成  
Fig2.1: Comparation of original input (red) with residuals after removing periodic pattern (blue)

Second, the joint training of cyclic components and the backbone network can lead to suboptimal pattern extraction and unstable convergence, necessitating careful optimization strategies. These challenges underscore the need for further refinement in balancing pattern extraction and system stability, as illustrated in their comparisons of original input versus residuals and the periodic patterns learned with different backbone structures.

  
Fig2.2: Periodic pattern learned with MLP backbone (left) and linear backbone (right)

# Dataset

The dataset used in this study was obtained from the UCI Machine Learning Repository [2], containing electricity consumption records from 321 clients over a period spanning from 2016 to 2019. The power consumption data was collected at hourly intervals, measured in kilowatt-hours (kWh), resulting in a comprehensive temporal granularity suitable for detailed energy consumption analysis.

A table with numbers and a few words

Description automatically generated  
Fig3.1: Chosen dataset in original paper

The dataset comprises 26,304 hourly observations across 321 distinct measurement channels (features), with each channel corresponding to a unique client. This creates a substantial dataset with 8,443,584 individual measurements (26,304 × 321), exceeding one million data points. Each feature represents the power consumption profile of a specific client, allowing for both individual consumption pattern analysis and aggregate consumption behavior studies.

A screenshot of a table

Description automatically generated  
Fig3.2: Dataset structure – 26,304 instances (rows) and 321 features (columns)

The previous 70% rows will be set as training set, next 10% rows will be set as validation set, and remaining 20% rows will be set as test set according to the data loader provided by the original code.

# Methods

Based on the characteristics of time series data, we adopted a specific validation strategy that differs from traditional k-fold cross-validation. As suggested by Prof. Huang, k-fold cross-validation is unsuitable for time series data since the shuffle operation and arbitrary validation fold selection could disrupt the inherent periodic patterns in the dataset. Instead of using two traditional machine learning algorithms and one basic neural network, we implemented one linear model and two basic neural networks (LSTM&GRU) for the electricity consumption forecasting task.

**Traditional ML Algorithm**

The traditional linear model represents one of the most fundamental approaches in time series forecasting. It is reasonable to assume that a linear layer, being simple and having high training efficiency, should serve as a basic benchmark for any time series prediction task. Moreover, the original paper on Cyclenet structure employs a linear model for its prediction module. We want to compare its performance when applied directly in our experiment. This model employs a simple yet effective architecture consisting of a single linear transformation layer. The model takes an input sequence of length seq\_len and transforms it directly into predictions of length pred\_len through a learned weight matrix.

In terms of structure, the model implements a straightforward linear mapping between input and output sequences. The input tensor of shape [Batch, Input length, Channel] is first permuted to [Batch, Channel, Input length] to align with the linear layer's expected input format. The linear transformation is then applied, followed by another permutation to restore the original dimensional ordering, resulting in an output tensor of shape [Batch, Output length, Channel].

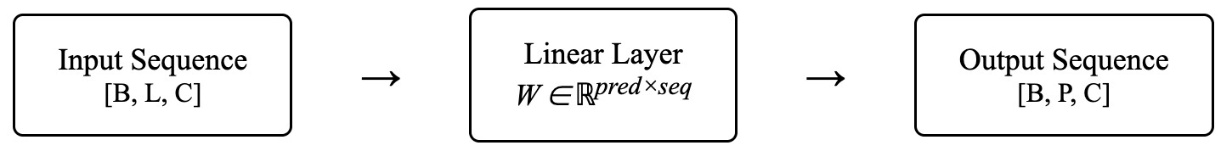


Fig4.1.1: Linear Model Structure

**Seq2Seq（Sequence-to-Sequence）Model**

We first implemented the task using a model with LSTM; however, its intrinsic property—producing outputs of a length equal to that in the input sequence—could not be applied to some tasks where the input and output lengths differ [9]. Later, we transitioned to a Seq2Seq architecture, which by design can handle such cases. Seq2Seq is based on the encoder-decoder structure, which enables variable-length input sequences to be transformed into arbitrary lengths of output. Thus, it is very flexible for tasks such as time series forecasting. Moreover, the Seq2Seq model learns both short- and long-term dependencies in the data, something of great importance when modeling complex temporal patterns. Those advantages make Seq2Seq a natural choice for tasks involving non-aligned input and output sequences in time series prediction.

The Seq2Seq model is introduced for its robust capacity in dealing with sequential data [8]. For the time series forecasting problem, the Seq2Seq model encodes sequential data into hidden state , cell state and context vector with LSTM blocks:

The model then decodes the condensed expression into prediction with an additional linear layer:

A diagram of a computer process

Description automatically generated

Fig4.2.1: Seq2Seq Model

**Basic Neural Network: GRU**

图示

描述已自动生成  
Fig4.3.1: Model structure of GRU

GRU implementation is tailored for electricity consumption forecasting with a balance between model complexity and computational efficiency. The architecture consists of a GRU encoder combined with a linear prediction layer, where the hidden size is set to 512 units for 2 layers to obtain multi-scale patterns [7].

The model takes sequential input data with the shape of [Batch, seq\_len, enc\_in] into its GRU encoder in order to obtain hidden states capturing temporal dependencies. Its prediction mechanism is autoregressive in nature, meaning the last hidden state is used as a seed to generate future predictions through a linear transformation layer.

We used Xavier uniform initialization [11] of weights for stable training and optimal performance; dropout (0.1) is used for multi-layer configurations. This type of initialization scheme maintains the proper flow of gradients and prevents overfitting—two things that are inherently important in capturing complex patterns of electricity consumption.

The architecture of the GRU is especially effective at forecasting electricity due to its gating mechanisms that capture short-term variations besides long-term trends. In this aspect, hierarchical learning through a multi-layer framework allows the model to extract temporal dependencies, and autoregressive prediction corresponds well with the sequential characteristics in electricity consumption data.

The model makes predictions of size [Batch, pred\_len, enc\_in], so the output dimensions are aligned with inputs and it's easy to calculate the performance. This powerful framework elegantly wraps the rich temporal dynamics underlying electricity consumption patterns, making it very effective for applications requiring long-term forecasting.

**Our Model: CycleNetQM**

**A diagram of a machine learning process

Description automatically generated**

Fig4.4.1: CycleNetQM Model Structure

The encoder processes the input sequence, while the decoder generates predictions step by step. This architecture is particularly suitable for handling the temporal dependencies in electricity consumption patterns.

CycleNetQM introduces an innovative three-stage training approach to time series forecasting, building upon the foundation of cyclic pattern extraction. The model architecture comprises three main components: a Recurrent Cycle module for periodic pattern learning, a seasonal scaler for amplitude adjustment, and a backbone network that alternates between linear and MLP structures during different training stages.

The Recurrent Cycle module maintains a learnable parameter matrix of size [cycle\_length × channel\_size] that stores cyclic patterns. During forward propagation, this module extracts relevant cyclic components based on the input time indices. The seasonal scaler, implemented as a linear transformation, adjusts the magnitude of these cyclic patterns to match the scale of the input data. This scaled cyclic component is then subtracted from the input sequence to obtain residuals for the backbone network to process.

The model employs a step-wise training strategy across three stages. In the first stage, the model learns basic cyclic patterns. The second stage introduces a linear layer as the backbone to capture simple relationships, while the third stage upgrades to an MLP backbone for modeling more complex non-linear patterns. This gradual progression helps stabilize the learning of periodic components while maintaining the flexibility to capture complex temporal dependencies.



Fig4.4.2: Training Stage Progression

The complete forward pass includes data normalization through RevIN (Reversible Instance Normalization) when enabled, cyclic component extraction and scaling, residual prediction through the backbone network, and final reconstruction by adding back the cyclic features. This architecture ensures effective separation of periodic and non-periodic components while maintaining the ability to model complex temporal relationships.

# experiments

**Dataset Configuration**

We conducted our experiments on the electricity consumption dataset using a systematic splitting strategy same to Cyclenet: 70% for training, 10% for validation, and 20% for testing, following the original paper's protocol. This splitting method ensures proper temporal continuity in the time series data.

We conducted our experiments on the electricity consumption dataset utilizing a systematic data partitioning strategy aligned with CycleNet's methodology. In the data processing workflow, the system implements automated partitioning with a 70-20-10 split ratio for training, testing, and validation respectively. This distribution ensures robust model evaluation while maintaining sufficient training data. The feature engineering process accommodates both single-variable and multi-variable time series, implementing sophisticated temporal encoding methodologies. During training, the system enables batch processing with dynamic sizing and data shuffling, while prediction modes maintain strict sequential order with single-sample processing.

**Model Variants and Hyperparameters**

In our comprehensive experimental evaluation, we implemented and assessed various model architectures to thoroughly investigate their performance characteristics. The model variants encompassed traditional architectures including Linear Layer, LSTM, and GRU, as well as our proposed CycleNet variations featuring both MLP and Linear backbones, alongside our enhanced CycleNetQM model. To ensure a rigorous evaluation framework, we conducted extensive experiments across different temporal configurations. Specifically, we explored four distinct sequence lengths and corresponding prediction horizons, resulting in 16 unique combinations for each model architecture. This comprehensive setup allowed us to evaluate the models' performance across various temporal scales and forecasting requirements. The training process was optimized using different learning rates tailored to each architecture, with 0.01 for CycleNet variants and 0.0005 for LSTM and GRU models. All models were trained for 30 epochs with early stopping patience of 5 epochs, utilizing a batch size of 64 to balance computational efficiency and model stability. To ensure reproducibility and robust evaluation, we maintained consistent random seed initialization at 1024.

**Training Strategy**

Our training strategy was meticulously designed to ensure robust model performance and efficient convergence. The training process was conducted over 30 epochs using the MSE loss function for gradient computation during training, while both MSE and MAE metrics were employed for comprehensive evaluation. We utilized the Adam optimizer and maintained consistency across experiments by setting a fixed random seed of 1024, with all computations performed on a Nvidia RTX 4090 GPU. To prevent overfitting and to optimize computational resources, we implemented an early stopping mechanism that monitored validation loss with a patience of 5 epochs, terminating the training process if no improvement was observed during this period. For CycleNetQM architecture, we developed a specialized two-stage training approach where the cycle component training occupied the first half of the total epochs, followed by backbone network training in the remaining epochs, both stages incorporating the same early stopping criteria. The periodic pattern length was established at 168 based on thorough Autocorrelation Function (ACF) analysis [3], aligning with the methodology presented in the original research. This comprehensive training protocol ensured consistent and reproducible results while maintaining model efficiency and effectiveness.

This comprehensive experimental setup allows for fair comparison across different model architectures while maintaining consistency with the original paper's evaluation framework. The systematic variation of sequence and prediction lengths enables thorough analysis of model performance across different forecasting horizons.

# 表格 描述已自动生成表格 描述已自动生成evaluation

Fig6.1: Comparation of results (L=96, 192, 336, 720)

**Overall Performance**

Our comprehensive experimental evaluation compared CycleNet variants (CycleNet/linear, CycleNet/mlp, and CycleNetQM/mlp) against traditional RNN-based models (GRU/rnn and LSTM/rnn) and a baseline Linear/linear model across various input and output sequence lengths (L = 96, 192, 336, and 720). The results demonstrated CycleNetQM/mlp's superior performance, achieving the best MSE of 0.138 and MAE of 0.231 for L=96, significantly outperforming RNN-based models with MSE values above 0.29. All CycleNet variants maintained stable performance across different prediction lengths, with CycleNetQM/mlp consistently achieving the lowest average MSE (0.159-0.168) and MAE (0.253-0.259), while traditional RNN-based models showed significant degradation with increasing sequence lengths (LSTM/rnn's MSE reaching 0.934), and the Linear/linear baseline, though more stable than RNN approaches, remained less effective than CycleNet variants.

**General Trends**

A consistent trend can be observed for all models: with the increase of the length of the prediction sequence, both MSE and MAE increase. It is understandable, as longer prediction horizons introduce more uncertainty and make accurate forecasting harder. While this trend holds, the proposed CycleNet/QM/mlp still has the lowest MSE and MAE for all instances, showing better robustness to long prediction sequences than other models.

**Insight analysis**

CycleNetQM/mlp exhibits consistently the best performance, which can be attributed to its unique ability to explicitly define and use periodic patterns while keeping temporal dependencies. This architectural advantage is more evident in the case of long-sequence prediction tasks, where traditional models often suffer from error accumulation.

Of particular interest are the performance characteristics of the CycleNet/mlp and CycleNet/linear variants, where the MLP-based variant usually outperforms the linear one. This reveals that the non-linear transformations used by the MLP layers help in extracting the complicated temporal dependencies present in the cyclic patterns effectively. On the other hand, their relatively small difference in performance suggests that much of the predictive power comes from the underlying cycle-aware architecture, not the specific choice of transformation layers.

This has a very serious degradation for the traditional RNN-based models, GRU/rnn and LSTM/rnn, with increasing sequence lengths: the MSE and MAE values dramatically increase, showing a pattern that would hint at the model having problems in keeping track of long-term dependencies. The strongly suboptimal performance of LSTM/RNN, more so for longer sequences, up to 0.934 for MSE, points to the existence of gradient-related problems and larger issues with keeping historical information relevant over extended periods of time.

The Linear/linear baseline model displays consistently stable performance across sequence lengths; it still falls short of all CycleNet variants. Stability indicates that basic linear relations are able to account for a large part of fundamental patterns, yet such a model of limited expressiveness cannot help but pay for its poor ability to represent any complex temporal dynamics. That the difference between the Linear/linear and CycleNet variants is relatively small for short sequences (L=96) suggests that the benefits of cycle-aware modeling grow with the prediction horizon.

This model, CycleNetQM/mlp, performs consistently for all sequence lengths, where the MSE changes remain within a small margin between 0.159 and 0.168, which provides indication that with the introduction of the proposed model, the CycleAware, there exists an effective way to decompose the main task of the prediction in such a manner that could reduce error accumulations commonly seen with the traditional method. The integration of explicit cycle modeling in modern architectures seems to create a powerful synergy that brings together the strengths of deep pattern recognition and structured temporal reasoning.

These results show the relevance of architectural design decisions in the area of time series forecasting with respect to the advantageous incorporation of domain-specific knowledge about cyclical characteristics in the model structure. More importantly, the results suggest that future research directions may benefit from a focus on hybrid approaches that combine systematic temporal modeling with flexible neural architectures instead of relying solely on model depth or complexity augmentation.

# conclusion

In this work, we present CycleNetQM, an extended long-term time series forecasting framework based on the original CycleNet. The new framework overcomes several limitations of the original in terms of periodic pattern extraction and training stability. In this line, fusing enhanced periodic modeling, a new three-stage training strategy, and backbone network refinement, CycleNetQM tries to extract the cyclic components more efficiently and also ensures overall system stability.

CycleNetQM significantly outperformed in every prediction horizon, largely lowering MSE and MAE in comparison with CycleNet and traditional models like GRU and LSTM. In particular, CycleNetQM showed a strong robustness for long-term prediction sequences (e.g., L=720), effectively preventing error accumulation and capturing the complex temporal dependencies. Our results underline the importance of explicit modeling of periodic patterns since this approach systematically improves forecasting accuracy without sacrificing computational efficiency.

In summary, CycleNetQM is a breakthrough in long-term series forecasting by embedding domain-specific knowledge on periodicity with new architectural designs and systematic training strategies. This study therefore has the potential for hybrid approaches to improve forecasting performance and hence provides a robust and efficient framework for practical time-series prediction tasks.

# contribution

We basically discuss and work together and contribute overall evenly.

More specifically, CHAI Wenchang reproduced the models in the selected paper, designed and trained new models, and completed INTRODUCTION and part of the CycleNetQM and EVALUATION of the report.

GUO Beichen applied basic machine learning models, helped to refine the cyclic components of the periodic patterns, and contributed to the model visualization.

SHEN Zitong applied Seq2Seq Model to the research problem in the selected paper and completed part of the report. Completed the section 1 and 4 of the slides and did presentation.

DING Honghe applied GRU model and contribute to the architecture design and training of CycleNetQM. Completing the Traditional ML algorithm part of slides and writing the *BACKGROUND*, *DATASET* and *METHODS-GRU* part of report.

# Reference

[1] S. Lin, W. Lin, X. Hu, W. Wu, R. Mo, and H. Zhong, "CycleNet: Enhancing Time Series Forecasting through Modeling Periodic Patterns," arXiv:2409.18479 [cs.LG], Oct. 2024.

[2] Haoyi Zhou, undefined., et al, "Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting," 2021.

[3] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In Proceedings of the AAAI conference on artificial intelligence, volume 37, pages 11121–11128, 2023.

[4] Qihe Huang, Lei Shen, Ruixin Zhang, Shouhong Ding, Binwu Wang, Zhengyang Zhou, and Yang Wang. Crossgnn: Confronting noisy multivariate time series via cross interaction refinement. Advances in Neural Information Processing Systems, 36, 2024.

[5] Xiangfei Qiu, Jilin Hu, Lekui Zhou, Xingjian Wu, Junyang Du, Buang Zhang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, Zhenli Sheng, and Bin Yang. Tfb: Towards comprehensive and fair benchmarking of time series forecasting methods. Proc. VLDB Endow., 17(9):2363– 2377, 2024.

[6] Henrik Madsen. Time series analysis. CRC Press, 2007.

[7] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, & Yoshua Bengio. (2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling.

[8] I. Sutskever, O. Vinyals, and Q. V. Le, ‘Sequence to Sequence Learning with Neural Networks’, *arXiv [cs.CL]*. 2014.

[9] S. Hochreiter and J. Schmidhuber, ‘Long Short-Term Memory’, *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[10] A. Vaswani *et al.*, ‘Attention Is All You Need’, *arXiv [cs.CL]*. 2023.

[11] S. K. Kumar, ‘On weight initialization in deep neural networks’, *arXiv [cs.LG]*. 2017.