# DeRNN: Decomposed Recurrent Neural Network for Long-Term Time Series Forecasting

Shanyun Qian[1]

[1] Changsha University Of Science & Technology, Changsha 410114, China

**Abstract.** Long-Term Time Series Forecasting (LTSF) is pivotal in domains like energy and traffic management but necessitates capturing intricate dependencies over extended windows. While Transformer-based models dominate, they suffer from quadratic complexity and positional insensitivity. Conversely, recent lightweight MLP/RNN-based models often forcibly compress conflicting dynamic features—linear trends and non-linear fluctuations—into a single channel, leading to suboptimal accuracy. To address these limitations, we propose the Decomposed Recurrent Neural Network (DeRNN). Our approach decouples global trend modeling from local fluctuation extraction via an asymmetric dual-track architecture. Specifically, we introduce a Trend Anchor Track to preserve global scale via direct linear projection, and a Seasonal Feature Track utilizing Bi-directional GRUs to capture complex non-linear dependencies within a reversible normalized space. Extensive experiments on seven bench-marks demonstrate that DeRNN achieves highly competitive, and in most cases superior, accuracy against state-of-the-art methods while maintaining extremely low latency and memory usage. Furthermore, the model exhibits superior robustness against noise and distribution shifts.

**Keywords:** Long-Term Time Series Forecasting, Time Series Decomposition, Recurrent Neural Network

## 1 Introduction

Long-Term Time Series Forecasting (LTSF) requires capturing intricate dependencies from extended look-back windows to predict long future sequences[1], playing a pivotal role in domains like energy and traffic management[2, 3]. In recent years, Transformer-based architectures have dominated this field by leveraging their global attention mechanisms[4]. However, the quadratic computational complexity ($\mathcal{O}(L^2)$) and weak sensitivity to positional information impose severe resource bottlenecks when processing ultra-long sequences. Recently, lightweight models based on MLPs[5] and RNNs[6] have regained attention. Nevertheless, these methods often forcibly compress simple linear trends and complex non-linear fluctuations into a single channel, leading to suboptimal accuracy in LTSF scenarios. This design suffers from two fundamental limitations: (1) As illustrated in **Fig. 1**, Trend Dynamics and Fluctuation Dynamics are inherently conflicting dynamic features; (2) The indirect decoding mechanism is highly prone to introducing unnecessary cumulative errors.

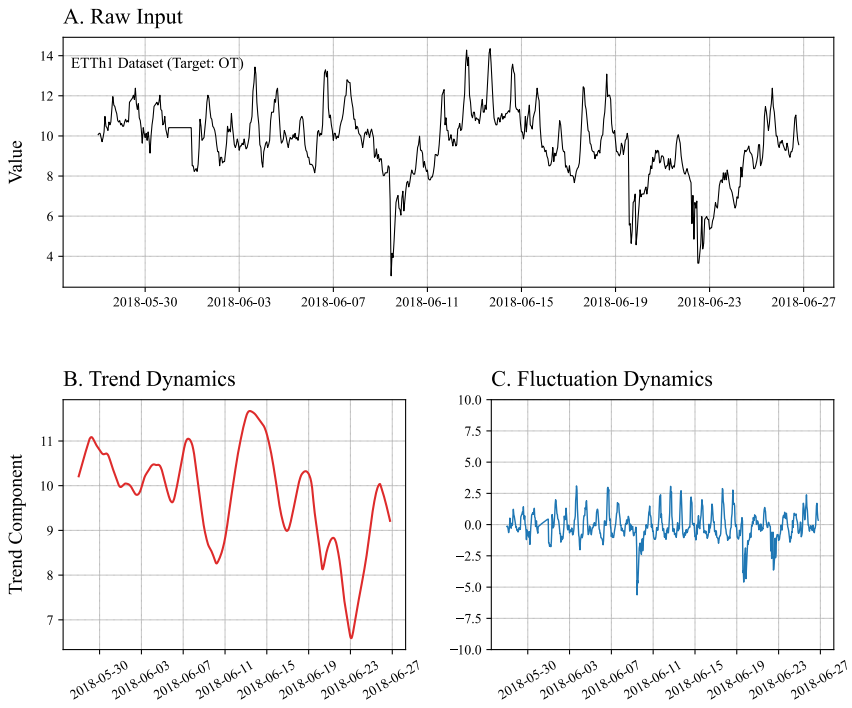


**Fig. 1.** Analysis of temporal dynamics on ETTh1. (A) Raw Input. STL decomposition separates the raw series into (B) a linear-dominant trend and (C) non-linear fluctuations.

To address these issues, this paper proposes the Decomposed Recurrent Neural Network (DeRNN). Our key insight is that simple trends should be handled by simple linear layers, whereas complex dependencies should be intricately modeled by recurrent networks. First, we directly model the global trend of the target variable via a linear layer. Second, the model utilizes a Bi-directional GRU to perform bi-directional context encoding on high-dimensional features, employing an MLP to capture complex non-linear dependencies while circumventing cumulative errors. Finally, a learnable global fusion mechanism is adopted to integrate the linear trends with the non-linear features. In summary, our contributions are as follows:

- We propose DeRNN, a hybrid architecture combining linear trend modeling and non-linear fluctuation feature extraction, effectively resolving the challenge of balancing long-term trends and local details in LTSF.
- We introduce Bi-directional GRU and a direct projection strategy to simplify the decoding process, thereby enhancing robustness in ultra-long sequence forecasting.
- Extensive experiments on mainstream benchmark datasets demonstrate that the improved model performs on par with or significantly outperforms state-of-the-art (SOTA) methods in prediction accuracy across a majority of scenarios, while maintaining extremely low memory usage and inference latency.

# 2 Related Work

## 2.1 Transformer-based Models for LTSF

Transformer-based architectures have significantly advanced LTSF by leveraging self-attention mechanisms to capture long-range dependencies. Early works focused on reducing the quadratic complexity, such as Pyraformer[7], which introduced pyramidal attention to capture multi-resolution temporal patterns. CrossFormer[8] further explored cross-dimension dependencies to strengthen multivariate modeling. Recently, iTransformer[9] proposed inverting the traditional structure by applying attention across variates rather than time steps, achieving SOTA performance. Despite their success, many Transformer models still struggle with positional information encoding and high computational overhead when dealing with ultra-long sequences, as noted in recent critiques[10].

## 2.2 RNN and MLP-based Models for LTSF

In response to the complexity of Transformers, lightweight models based on Multi-Layer Perceptrons (MLPs) and Recurrent Neural Networks (RNNs) have regained prominence. DLinear[10] first challenged the dominance of Transformers using simple linear layers. Subsequently, LightTS[11], TSMixer[5] further enhanced MLP architectures with residual connections and encoding-decoding structures. Parallelly, modern RNN variants are breaking the limitations of sequential bottlenecks. SegRNN[12] employed segment-wise recurrence to improve efficiency, while MLANet[6] integrated adaptive mechanisms for efficient long-term modeling. Our DeRNN follows this resurgence of efficient architectures, utilizing Bi-directional GRUs to capture intricate non-linear dependencies more effectively than pure MLP-based methods.

## 2.3 Time Series Decomposition

Time series decomposition is a fundamental technique to disentangle complex temporal patterns into interpretable components like trend and seasonality. Autoformer[13] pioneered the integration of deep decomposition blocks within neural networks. TimesNet[14] transformed 1D series into 2D variations to model multi-periodicity. Furthermore, Duet[15] leveraged dual clustering to enhance multivariate feature separation. Inspired by these approaches, DeRNN explicitly decomposes the series into linear-dominant trends and non-linear fluctuations, employing specialized modules to handle each component for superior accuracy.

# 3 Proposed Methodology

## 3.1 Structure Overview

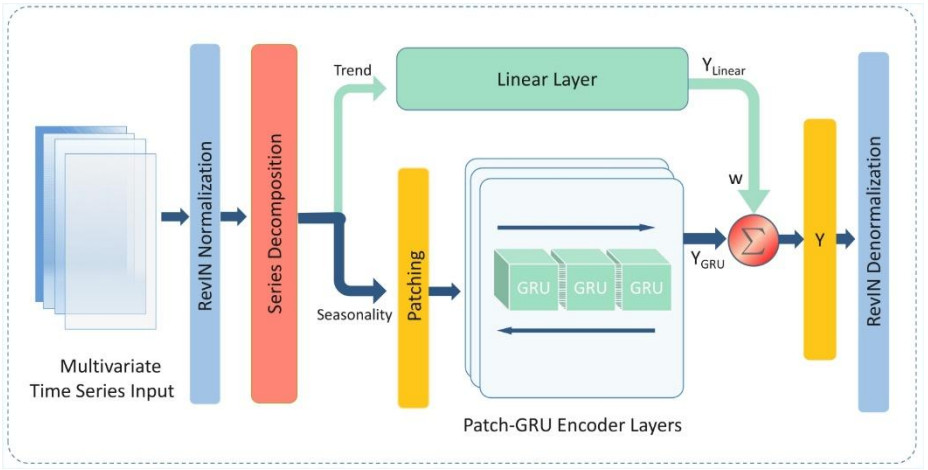


**Fig. 2.** The Architecture of DeRNN.

Figure 2 illustrates the architecture of our proposed Decomposed Recurrent Neural Network (DeRNN). To address the challenges of non-stationarity and error accumulation in long-term multivariate time series forecasting, DeRNN adopts a dual-track strategy that explicitly disentangles the modeling of trend and seasonal patterns.
Specifically, given a multivariate time series input $\mathbf{X} \in \mathbb{R}^{B \times L \times C}$, where $L$ is the look-back window size and $C$ is the number of channels, we first employ Reversible Instance Normalization (RevIN) to mitigate the distribution shift. The model then diverges into two parallel processing tracks:

- **The Trend Anchor Track** A linear pathway that directly captures the low-frequency global trends from the raw target series.
- **The Seasonal Feature Track** A non-linear pathway that utilizes a Patch-GRU Encoder to capture high-frequency local dependencies and global correlations from the normalized series.

Finally, the outputs from both tracks are adaptively fused via a learnable weighting mechanism to generate the final prediction $\mathbf{Y} \in \mathbb{R}^{B \times F \times C}$, where $F$ is the forecasting horizon. The overall process can be formulated as:

$$X_{norm} = RevIN(X), \tag{1}$$

$$\hat{Y}_{trend} = Linear(X_{target}), \tag{2}$$

$$\hat{Y}_{seas} = PatchGRU(X_{norm}), \tag{3}$$

$$\hat{Y}_{final} = \mathrm{Denorm}(w_0 \cdot \hat{Y}_{trend} + w_1 \cdot \hat{Y}_{seas}) \tag{4}$$

where $[w_0, w_1]^\top = \mathrm{Softmax}(\mathbf{W}_{\mathrm{combine}})$ denotes the globally learnable fusion weights, ensuring $w_0 + w_1 = 1$.

### 3.2 Series Decomposition and Trend Modeling

Real-world time series often exhibit non-stationary behaviors, where the statistical properties change over time. Standard RNNs often struggle to distinguish between global trends and local variations. To address this, we implement an implicit decomposition approach aligned with the model's dual-track design.

**Trend Modeling.**
We approximate the slowly changing global trend via a dedicated linear "Anchor Track":

$$\widehat{\boldsymbol{Y}}_{trend} = \boldsymbol{W}_{\boldsymbol{trend}}\boldsymbol{X}_{\boldsymbol{target}} + \boldsymbol{b}_{\boldsymbol{trend}}. \tag{5}$$

This "Anchor Track" ensures that the model preserves the fundamental trajectory of the series, preventing the neural network from overfitting to high-frequency noise when predicting the trend.

**Residual Modeling.**
Simultaneously, the input **X** is normalized by subtracting its instance-specific mean and dividing by its variance via RevIN:

$$X_{\mathrm{norm}}^{(i)} = \left(X^{(i)} - \mu_i\right)/\sigma_i \tag{6}$$

It effectively standardizes the local distribution, mitigating non-stationary distribution shifts. By capturing the structural trajectory via the Trend Anchor Track, we enable the Seasonal Feature Track to focus exclusively on these standardized, zero-mean oscillating patterns, thus decoupling sequence normalization from topological trend modeling.

### 3.3 Patch-GRU Encoder for Seasonal Patterns

Modeling long-term dependencies in the residual component is challenging due to the vanishing gradient problem and high computational cost of point-wise RNNs. To overcome this, we propose the Patch-GRU Encoder.

**Patching and Embedding.**
Instead of processing the time series point-by-point, we segment the normalized input series $\mathbf{X}_{\mathrm{norm}}$ into non-overlapping patches. Let $P$ denote the patch length. The input sequence of length $L$ is reshaped into $N = \lfloor L/P \rfloor$ segments. This patching operation, followed by a linear projection (embedding), significantly reduces the trajectory length from $L$ to $N$, enhancing the model's ability to capture long-term history:

$$E_{patch} = \mathrm{Embed}\big(\mathrm{Patch}(X_{norm})\big). \tag{7}$$

where $\mathbf{E}_{\mathrm{patch}} \in \mathbb{R}^{B\cdot C\times N\times D}$ and $D$ is the hidden dimension.

**Bi-directional GRU Encoding.**
To capture the temporal dependencies among patches, we utilize a Bi-directional Gated Recurrent Unit (Bi-GRU). The Bi-GRU captures the context from both past and future directions within the look-back window, enriching the representation of each patch:

$$H = \mathrm{BiGRU}\big(E_{patch}\big). \tag{8}$$

The bidirectional mechanism ensures that the final hidden states encapsulate the global semantic information of the entire look-back window.

### 3.4 Direct Projection and Learnable Global Fusion

**Direct Projection.**
DeRNN employ a Direct Projection strategy instead of traditional recursive structure. We take the final hidden states of the Bi-GRU, flatten them, and map them directly to the entire forecast horizon $F$ via a Multi-Layer Perceptron (MLP):

$$\hat{Y}_{seas} = \mathrm{MLP}\left(\mathrm{Reshape}\big(H_{final}\big)\right). \tag{9}$$

This design allows the model to generate all $F$ future steps in a single forward pass, significantly improving inference speed and avoiding severe error propagation in long-term forecasting ($F \geq 720$).

**Learnable Global Fusion.**
To combine the outputs from the Trend Anchor Track and the Seasonal Feature Track, we introduce a globally learnable parameter vector $\mathbf{W}_{\mathrm{combine}} \in \mathbb{R}^{2}$. We utilize a statically learned weighting strategy constrained by a Softmax operator:

$$[w_0, w_1]^{\top} = \mathrm{Softmax}(\mathbf{W}_{\mathrm{combine}}) \tag{10}$$

$$\hat{Y}_{final} = w_0 \cdot \hat{Y}_{trend} + w_1 \cdot \hat{Y}_{seas}. \tag{11}$$

Finally, $\hat{\mathbf{Y}}_{\mathrm{final}}$ is passed through the RevIN Denormalization module to restore the original scale of the time series. This parameterized fusion explicitly defines the dimensions and constraints of the weights, allowing DeRNN to automatically discover the optimal corpus-level balance between trend extrapolation and seasonal pattern matching.

## 4 Experimental Evaluation

### 4.1 Experimental Settings

**Datasets and Baselines.**
We evaluate DeRNN on seven widely used real-world benchmarks, including ETT, Traffic, Electricity, and Weather. We compare our method against state-of-the-art baselines, including Transformer-based models such as iTransformer, TimesNet, CrossFormer, Pyraformer, and Autoformer, as well as MLP/RNN-based models such as

TSMixer, DLinear, and LightTS. To ensure a fair comparison, the selection of baselines, dataset split proportions, and data pre-processing steps are kept consistent with the standardized benchmarks established by iTransformer.

**Implementation Details.**
Following previous studies, the historical look-back window is fixed at $L = 96$, and the prediction horizon is set to $T \in 96,192,336,720$. To facilitate reproducibility, we detail the specific configuration of DeRNN: the patch length is set to $P = 16$(8 for specific high-frequency datasets) and the hidden state dimension for the Bi-GRU is $D = 128$. The model is optimized using the Adam optimizer with an initial learning rate of $5 \times 10^{-4}$, an exponential learning rate decay, and a batch size of 32. The training process runs for a maximum of 30 epochs, utilizing an early stopping mechanism with a patience of 5 epochs based on the validation loss. All experiments are conducted on a single NVIDIA RTX 3060 GPU via PyTorch 2.0. We use Mean Squared Error (MSE) and Mean Absolute Error (MAE) as the evaluation metrics, defined as:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2, \quad MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|. \tag{12}$$

where $y_i$ and $\hat{y}_i$ denote the ground truth and predicted values, respectively, and $N$ represents the total number of samples.

## 4.2 Main Results

**Table 1.** Multivariate long-term forecasting results. The best results are highlighted in bold and the second best are underlined.

| Models | | DeRNN (Ours) | | iTransformer (ICLR 2024) | | TSMixer (AAAI 2024) | | TimesNet (ICLR 2023) | | DLinear (AAAI 2023) | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Datasets** | **Metric** | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| **ETTh1** | 96 | **0.377** | **0.398** | 0.395 | 0.410 | 0.494 | 0.502 | 0.389 | 0.412 | 0.396 | 0.411 |
| | 192 | **0.432** | **0.429** | 0.449 | 0.441 | 0.597 | 0.567 | 0.439 | 0.442 | 0.445 | 0.440 |
| | 336 | **0.467** | **0.447** | 0.492 | 0.465 | 0.677 | 0.618 | 0.494 | 0.471 | 0.487 | 0.465 |
| | 720 | **0.499** | **0.489** | 0.522 | 0.504 | 0.752 | 0.674 | 0.518 | 0.494 | 0.513 | 0.510 |
| **ETTh2** | 96 | **0.293** | **0.344** | 0.300 | 0.350 | 1.056 | 0.806 | 0.337 | 0.371 | 0.341 | 0.395 |
| | 192 | **0.371** | **0.396** | 0.382 | 0.400 | 2.587 | 1.403 | 0.405 | 0.415 | 0.482 | 0.479 |
| | 336 | 0.425 | 0.437 | **0.424** | **0.432** | 2.407 | 1.347 | 0.451 | 0.449 | 0.593 | 0.542 |
| | 720 | 0.440 | 0.457 | **0.426** | **0.445** | 2.051 | 1.218 | 0.434 | 0.448 | 0.840 | 0.661 |
| **ETTm1** | 96 | **0.322** | **0.361** | 0.341 | 0.376 | 0.479 | 0.470 | 0.334 | 0.375 | 0.346 | 0.374 |
| | 192 | **0.368** | **0.386** | 0.381 | 0.395 | 0.480 | 0.482 | 0.408 | 0.414 | 0.382 | 0.391 |
| | 336 | **0.399** | **0.410** | 0.419 | 0.419 | 0.541 | 0.525 | 0.415 | 0.422 | 0.415 | 0.415 |
| | 720 | **0.461** | **0.447** | 0.486 | 0.456 | 0.617 | 0.574 | 0.479 | 0.458 | 0.473 | 0.451 |
| **ETTm2** | 96 | **0.175** | **0.258** | 0.184 | 0.267 | 0.250 | 0.366 | 0.188 | 0.268 | 0.193 | 0.293 |
| | 192 | **0.243** | **0.302** | 0.253 | 0.312 | 0.492 | 0.559 | 0.252 | 0.307 | 0.285 | 0.361 |
| | 336 | **0.301** | **0.340** | 0.315 | 0.352 | 0.833 | 0.735 | 0.321 | 0.350 | 0.385 | 0.429 |
| | 720 | **0.396** | **0.396** | 0.412 | 0.406 | 2.543 | 1.352 | 0.420 | 0.406 | 0.556 | 0.523 |
| **Traffic** | 96 | 0.455 | 0.299 | **0.395** | **0.269** | 0.541 | 0.365 | 0.589 | 0.315 | 0.711 | 0.437 |
| | 192 | 0.465 | 0.303 | **0.413** | **0.277** | 0.556 | 0.385 | 0.618 | 0.324 | 0.662 | 0.417 |
| | 336 | 0.482 | 0.310 | **0.422** | **0.282** | 0.582 | 0.395 | 0.632 | 0.336 | 0.669 | 0.419 |
| | 720 | 0.514 | 0.332 | **0.460** | **0.302** | 0.623 | 0.422 | 0.660 | 0.349 | 0.709 | 0.437 |
| **Weather** | 96 | 0.176 | 0.218 | 0.175 | **0.216** | 0.180 | 0.252 | 0.183 | 0.217 | 0.195 | 0.252 |
| | 192 | 0.223 | **0.258** | 0.225 | **0.258** | 0.218 | 0.287 | 0.225 | 0.265 | 0.239 | 0.299 |
| | 336 | 0.278 | **0.298** | 0.280 | **0.298** | **0.261** | 0.321 | 0.282 | 0.304 | 0.281 | 0.331 |
| | 720 | 0.354 | **0.348** | 0.361 | 0.351 | **0.318** | 0.363 | 0.359 | 0.354 | 0.345 | 0.382 |
| **Electricity** | 96 | 0.165 | 0.252 | **0.148** | **0.240** | 0.204 | 0.308 | 0.168 | 0.272 | 0.210 | 0.302 |
| | 192 | 0.175 | 0.261 | **0.164** | **0.256** | 0.218 | 0.329 | 0.186 | 0.288 | 0.210 | 0.305 |
| | 336 | 0.192 | 0.279 | **0.178** | **0.271** | 0.239 | 0.350 | 0.203 | 0.304 | 0.223 | 0.319 |
| | 720 | 0.233 | **0.312** | 0.228 | **0.312** | 0.272 | 0.373 | **0.221** | 0.317 | 0.258 | 0.350 |

| Models | | DeRNN (Ours) | | CrossFormer (ICLR 2023) | | LightTS (2022) | | Pyraformer (ICLR 2022) | | Autoformer (NeurIPS 2021) | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Datasets** | **Metric** | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| **ETTh1** | 96 | **0.377** | **0.398** | 0.406 | 0.437 | 0.435 | 0.444 | 0.680 | 0.624 | 0.497 | 0.479 |
| | 192 | **0.432** | **0.429** | 0.456 | 0.456 | 0.494 | 0.478 | 0.896 | 0.743 | 0.528 | 0.506 |
| | 336 | **0.467** | **0.447** | 0.613 | 0.575 | 0.549 | 0.510 | 0.957 | 0.770 | 0.540 | 0.508 |
| | 720 | **0.499** | **0.489** | 0.809 | 0.680 | 0.612 | 0.569 | 1.000 | 0.794 | 0.541 | 0.520 |
| **ETTh2** | 96 | **0.293** | **0.344** | 0.740 | 0.609 | 0.414 | 0.451 | 1.320 | 0.897 | 0.368 | 0.406 |
| | 192 | **0.371** | **0.396** | 3.012 | 1.354 | 0.568 | 0.533 | 6.149 | 1.987 | 0.437 | 0.443 |
| | 336 | 0.425 | 0.437 | 1.973 | 1.105 | 0.669 | 0.576 | 4.614 | 1.835 | 0.483 | 0.487 |
| | 720 | 0.440 | 0.457 | 3.209 | 1.525 | 1.011 | 0.719 | 4.312 | 1.800 | 0.519 | 0.514 |
| **ETTm1** | 96 | **0.322** | **0.361** | 0.360 | 0.405 | 0.393 | 0.413 | 0.628 | 0.542 | 0.529 | 0.484 |
| | 192 | **0.368** | **0.386** | 0.383 | 0.414 | 0.436 | 0.441 | 0.612 | 0.532 | 0.635 | 0.533 |
| | 336 | **0.399** | **0.410** | 0.603 | 0.559 | 0.464 | 0.463 | 0.687 | 0.601 | 0.551 | 0.510 |
| | 720 | **0.461** | **0.447** | 0.993 | 0.754 | 0.556 | 0.526 | 0.922 | 0.734 | 0.747 | 0.582 |
| **ETTm2** | 96 | **0.175** | **0.258** | 0.236 | 0.327 | 0.227 | 0.326 | 0.349 | 0.445 | 0.242 | 0.314 |
| | 192 | **0.243** | **0.302** | 0.862 | 0.645 | 0.348 | 0.413 | 0.741 | 0.614 | 0.282 | 0.341 |
| | 336 | **0.301** | **0.340** | 1.315 | 0.835 | 0.459 | 0.484 | 1.198 | 0.851 | 0.330 | 0.368 |
| | 720 | **0.396** | **0.396** | 3.985 | 1.454 | 1.026 | 0.738 | 3.561 | 1.439 | 0.455 | 0.436 |
| **Traffic** | 96 | 0.455 | 0.299 | 0.515 | 0.270 | 0.712 | 0.440 | 0.698 | 0.403 | 0.674 | 0.405 |
| | 192 | 0.465 | 0.303 | 0.548 | 0.293 | 0.697 | 0.443 | 0.683 | 0.389 | 0.645 | 0.412 |
| | 336 | 0.482 | 0.310 | 0.576 | 0.303 | 0.717 | 0.452 | 0.687 | 0.389 | 0.610 | 0.379 |
| | 720 | 0.514 | 0.332 | 0.611 | 0.314 | 0.771 | 0.471 | 0.732 | 0.411 | 0.636 | 0.392 |
| **Weather** | 96 | 0.176 | 0.218 | **0.174** | 0.239 | 0.181 | 0.239 | 0.208 | 0.291 | 0.364 | 0.390 |
| | 192 | 0.223 | **0.258** | 0.232 | 0.301 | **0.215** | 0.274 | 0.225 | 0.311 | 0.290 | 0.350 |
| | 336 | 0.278 | **0.298** | 0.276 | 0.340 | 0.265 | 0.314 | 0.291 | 0.364 | 0.351 | 0.385 |
| | 720 | 0.354 | **0.348** | 0.372 | 0.412 | 0.331 | 0.364 | 0.376 | 0.402 | 0.420 | 0.420 |
| **Electricity** | 96 | 0.165 | 0.252 | 0.156 | 0.256 | 0.213 | 0.316 | 0.278 | 0.373 | 0.205 | 0.322 |
| | 192 | 0.175 | 0.261 | **0.164** | 0.263 | 0.222 | 0.325 | 0.294 | 0.390 | 0.223 | 0.334 |
| | 336 | 0.192 | 0.279 | 0.187 | 0.286 | 0.242 | 0.344 | 0.298 | 0.393 | 0.248 | 0.353 |
| | 720 | 0.233 | **0.312** | 0.229 | 0.322 | 0.277 | 0.371 | 0.299 | 0.387 | 0.266 | 0.372 |

**Table 1** demonstrates that DeRNN achieves state-of-the-art or highly competitive performance across seven benchmarks with a fixed look-back window of 96. While specialized baseline models exhibit marginal advantages on specific datasets (e.g., iTransformer on Electricity and Traffic), DeRNN demonstrates superior overall robustness and achieves the lowest errors on the majority of the metrics. Unlike Transformer-based models that often struggle with overfitting in ultra-long sequences or MLP-based architectures that lack sufficient non-linear expressiveness, DeRNN leverages a trend-fluctuation decoupling strategy to effectively balance global trend preservation with local dynamic modeling. This design ensures superior accuracy in complex scenarios, exemplified by a 4.4% MSE reduction on the ETTh1 dataset ($T = 720$) compared to the previous best method.

### 4.3 Prediction Visualization

**Fig. 3** provides a qualitative comparison of forecasting results on the ETTh1 dataset. DeRNN exhibits superior alignment with the ground truth compared to state-of-the-art baselines, particularly in capturing sharp peaks and preserving temporal phase. While MLP-based models like TSMixer and DLinear suffer from significant amplitude attenuation—failing to reach the extrema of the true series—and Transformer variants like CrossFormer exhibit noticeable phase shifts, DeRNN effectively models both global trends and intricate local fluctuations. This visualization corroborates the quantitative improvements, demonstrating the model's robustness in handling complex, non-stationary dynamics.

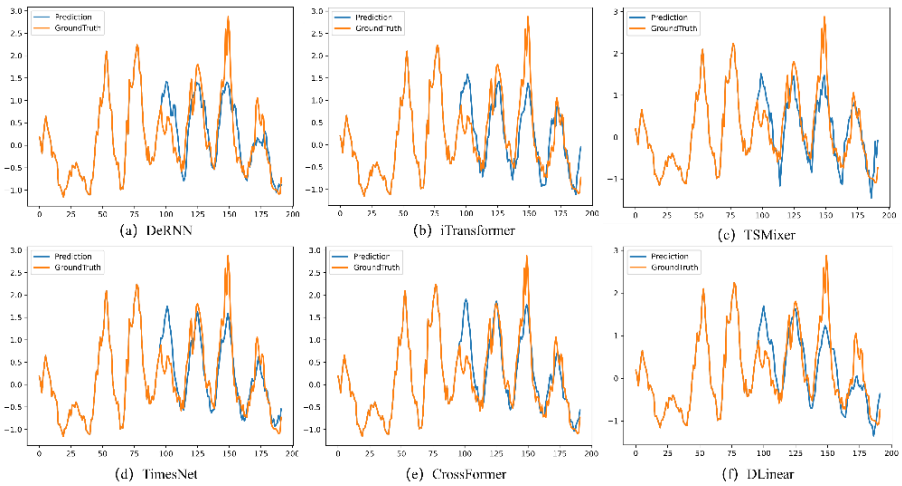


**Fig. 3.** Visualization of forecasting results on the ETTh1 dataset.

### 4.4 Ablation Study

To verify the contribution of each individual component, we conducted an ablation study on the ETTh1 and Weather datasets, averaging the metrics over all prediction horizons. As shown in **Table 2**, the full DeRNN architecture consistently achieves the lowest errors compared to all variants. Specifically, the removal of either the Linear Branch or the GRU Branch leads to distinct performance degradation, confirming the validity of our decoupling strategy in handling global trends and local non-linear dynamics respectively. Notably, omitting RevIN results in the most severe deterioration, underscoring the critical role of reversible normalization in mitigating the impact of non-stationary distribution shifts.

**Table 2.** Ablation Study Results on ETTh1 and ETTm1. Values represent the average MSE and MAE across four prediction lengths $T \in 96,192,336,720$.

| Model Variant | ETTh1 | | ETTm1 | |
|---|---|---|---|---|
| | MSE | MAE | MSE | MAE |
| **DeRNN** | **0.4437** | **0.4404** | **0.3876** | **0.3985** |
| w/o GRU | 0.4880 | 0.4675 | 0.4235 | 0.4153 |
| w/o Linear | 0.4513 | 0.4484 | 0.3892 | 0.3996 |
| w/o RevIN | 0.4550 | 0.4516 | 0.4050 | 0.4182 |
| w/o Patching | 0.4725 | 0.4666 | 0.3988 | 0.4114 |

### 4.5 Robustness Analysis

**Table 3** shows DeRNN yields the lowest degradation across all perturbations on the ETTh1 dataset. Unlike Transformers, which fail under 10 dropout (MSE $> 1.0$), DeRNN's Bi-GRU effectively infers missing temporal steps. Furthermore, its Trend

Anchor Track seamlessly adapts to drift noise, significantly outperforming models that cannot decouple macroscopic shifts from local fluctuations.

To assess the model's reliability in unstable industrial IoT environments, we introduce three characteristic perturbations to the input sequences using the tsaug library1. First, we apply Gaussian noise (zero-mean, $\sigma = 0.1$) to simulate the inevitable sensor thermal noise and measurement uncertainty commonly encountered in harsh industrial deployments. Second, we introduce Drift noise (maximum drift intensity of 0.5) to mimic the systemic bias caused by sensor aging or gradual calibration shifts over long-term operation. Finally, Dropout noise is applied by randomly masking 10 of the time steps to zero, which represents data packet loss due to transmission failures or network congestion in IoT communication protocols.

**Table 3.** Performance comparison of different models under three types of noise (Dropout, Gaussian, Drift) with different prediction lengths. MSE and MAE are used as metrics, and the best results are highlighted in bold.

| Noise | Length | DeRNN | | iTransformer | | DLinear | | TimesNet | | CrossFormer | | Autoformer | | TSMixer | | LightTS | | Pyraformer | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| Dropout | 96 | **0.4381** | **0.4248** | 0.4591 | 0.4410 | 0.4579 | 0.4412 | 0.4464 | 0.4377 | 0.4691 | 0.4708 | 0.5153 | 0.4824 | 0.5463 | 0.5293 | 0.5137 | 0.4853 | 0.8060 | 0.6872 |
| | 192 | **0.4900** | **0.4621** | 0.5069 | 0.4683 | 0.5028 | 0.4692 | 0.4981 | 0.4708 | 0.5178 | 0.4945 | 0.5680 | 0.5222 | 0.6411 | 0.5899 | 0.5691 | 0.5190 | 0.8930 | 0.7417 |
| | 336 | **0.5338** | **0.4831** | 0.5451 | 0.4907 | 0.5404 | 0.4935 | 0.5426 | 0.4974 | 0.6344 | 0.5871 | 0.5631 | 0.5218 | 0.7143 | 0.6373 | 0.6214 | 0.5510 | 1.0299 | 0.8192 |
| | 720 | **0.5396** | **0.5032** | 0.5500 | 0.5150 | 0.5608 | 0.5383 | 0.5688 | 0.5229 | 0.7846 | 0.6783 | 0.5914 | 0.5433 | 0.7830 | 0.6877 | 0.6773 | 0.6069 | 1.0027 | 0.8031 |
| Gaussian | 96 | **0.4881** | **0.4812** | 0.5098 | 0.4896 | 0.5069 | 0.4900 | 0.4952 | 0.4864 | 0.5245 | 0.5244 | 0.5962 | 0.5460 | 0.5918 | 0.5601 | 0.5724 | 0.5351 | 0.7227 | 0.6399 |
| | 192 | **0.5358** | **0.5044** | 0.5573 | 0.5136 | 0.5514 | 0.5138 | 0.5517 | 0.5172 | 0.5718 | 0.5353 | 0.5888 | 0.5369 | 0.6730 | 0.6078 | 0.6262 | 0.5637 | 0.8721 | 0.7164 |
| | 336 | **0.5746** | **0.5212** | 0.5940 | 0.5298 | 0.5890 | 0.5340 | 0.5902 | 0.5332 | 0.6606 | 0.5928 | 0.6167 | 0.5555 | 0.7672 | 0.6634 | 0.6757 | 0.5900 | 0.9561 | 0.7644 |
| | 720 | **0.5742** | **0.5301** | 0.5914 | 0.5449 | 0.6006 | 0.5683 | 0.6037 | 0.5500 | 0.8316 | 0.6999 | 0.5962 | 0.5616 | 0.8377 | 0.7154 | 0.7210 | 0.6378 | 1.0689 | 0.8326 |
| Drift | 96 | **0.5550** | **0.5246** | 0.5729 | 0.5353 | 0.5693 | 0.5360 | 0.5607 | 0.5351 | 0.5805 | 0.5609 | 0.6433 | 0.5883 | 0.6492 | 0.5987 | 0.6469 | 0.5791 | 0.7964 | 0.6889 |
| | 192 | **0.5937** | **0.5450** | 0.6220 | 0.5598 | 0.6140 | 0.5593 | 0.6029 | 0.5556 | 0.6245 | 0.5753 | 0.6895 | 0.6010 | 0.7362 | 0.6477 | 0.7019 | 0.6075 | 0.8971 | 0.7469 |
| | 336 | 0.6604 | 0.5810 | 0.6572 | **0.5747** | **0.6493** | 0.5777 | 0.6493 | 0.5750 | 0.7032 | 0.6226 | 0.6623 | 0.5910 | 0.8303 | 0.7009 | 0.7492 | 0.6322 | 0.9597 | 0.7749 |
| | 720 | **0.6320** | **0.5723** | 0.6499 | 0.5859 | 0.6607 | 0.6095 | 0.6601 | 0.5897 | 0.9020 | 0.7448 | 0.6390 | 0.5947 | 0.8963 | 0.7514 | 0.7918 | 0.6772 | 1.1036 | 0.8606 |

## 4.6 Efficiency Analysis

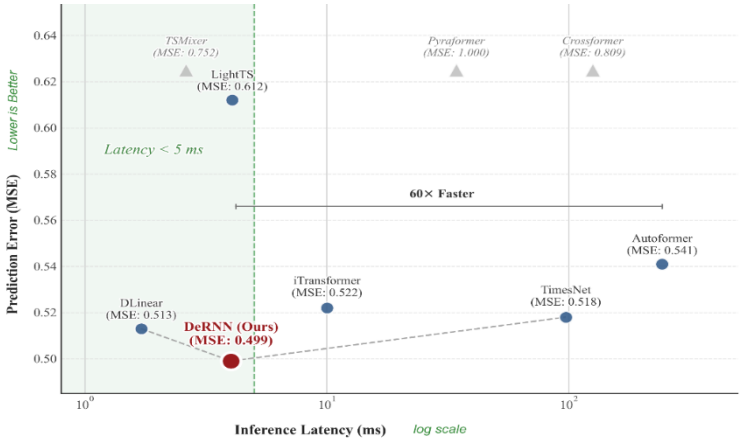


**Fig. 4.** Inference latency versus forecasting error on the ETTh1 dataset (T=720). The dashed line delineates the empirical Pareto front.

Empirical inference speed is a critical bottleneck for time-sensitive applications. As illustrated in **Fig. 4**, DeRNN establishes a new state-of-the-art Pareto balance. It strictly

[1] https://github.com/arundo/tsaug

operates within the ultra-low latency regime and yields the lowest overall error (MSE: 0.499), achieving an approximate 60 × speedup over Autoformer. Furthermore, it maintains a decisive performance margin over competitive efficient baselines such as iTransformer and TimesNet. This underscores the architectural elegance of DeRNN, demonstrating that it effectively eliminates traditional computational bottlenecks without compromising sequence modeling capability.

## 5 Conclusion

We propose DeRNN, an efficient dual-track architecture that successfully addresses the challenge of balancing long-term trends and local non-linear dynamics. Extensive experiments across seven benchmarks demonstrate that our model achieves state-of-the-art accuracy—notably achieving a 4.4% MSE reduction on the ETTh1 dataset—while maintaining significantly lower memory usage and nearly 3x faster inference latency than leading Transformer-based baselines. Furthermore, its superior resistance to Gaussian, drift, and dropout noise validates its immense potential for robust deployment in real-world industrial IoT environments.

## References

1. Das, Abhimanyu, Weihao Kong, Andrew Leach, Shaan K Mathur, Rajat Sen, and Rose Yu. 2023. “Long-Term Forecasting with TiDE: Time-Series Dense Encoder.” Transactions on Machine Learning Research. https://openreview.net/forum?id=pCbC3aQB5W.
2. Agarwal, Sanskar, Shivam Sharma, Kazi Newaj Faisal, and Rishi Raj Sharma. 2025. “Time-Series Forecasting Using SVMD-LSTM: A Hybrid Approach for Stock Market Prediction.” Journal of Probability and Statistics 2025 (1): 9464938.
3. Ge, Chang, Jie Yan, Weiye Song, et al. 2025. “Middle-Term Wind Power Forecasting Method Based on Long-Span NWP and Microscale Terrain Fusion Correction.” Renewable Energy 240: 122123.
4. Das, Manas Kamal, C Christopher Columbus, and E Elakiya. 2025. “A Temporal Attention-Based SARIMA–BiLSTM Residual Learning Model Tuned by Grey Wolf Optimizer for Parallel Urban Traffic Forecasting.” IEEE Access.
5. Chen, Si-An, Chun-Liang Li, Sercan O Arik, Nathanael Christian Yoder, and Tomas Pfister. 2023. “TSMixer: An All-MLP Architecture for Time Series Forecast-Ing.” Transactions on Machine Learning Research. https://openreview.net/forum?id=wbpxTuXgm0.
6. Jiang, Jihua, Bin Jiang, Yong Wang, and Duoqian Miao. 2025. “MLANet: An Efficient Recurrent Neural Network for Long-Term Time Series Forecasting.” Knowledge-Based Systems, 113971.
7. Liu, Shizhan, Hang Yu, Cong Liao, et al. 2022. “Pyraformer: Low-Complexity Pyramidal Attention for Long-Range Time Series Modeling and Forecasting.” International Conference on Learning Representations. https://openreview.net/forum?id=0EXmFzUn5I.
8. Zhang, Yunhao, and Junchi Yan. 2023. “CrossFormer: Transformer Utilizing Cross-Dimension Dependency for Multivariate Time Series Forecasting.” The Eleventh International Conference on Learning Representations. https://openreview.net/forum?id=vSVLM2j9eie.

9. Liu, Yong, Tengge Hu, Haoran Zhang, et al. 2024. “iTransformer: Inverted Transformers Are Effective for Time Series Forecasting.” The Twelfth International Conference on Learning Representations. https://openreview.net/forum?id=JePfAI8fah.
10. Zeng, Ailing, Muxi Chen, Lei Zhang, and Qiang Xu. 2023. “Are Transformers Effective for Time Series Forecasting?” Proceedings of the AAAI Conference on Artificial Intelligence 37: 11121–28.
11. Zhang, T, Y Zhang, W Cao, et al. 2022. “Less Is More: Fast Multivariate Time Series Forecasting with Light Sampling-Oriented Mlp Structures. arXiv 2022.” arXiv Preprint arXiv:2207.01186.
12. Lin, Shengsheng, Weiwei Lin, Wentai Wu, Feiyu Zhao, Ruichao Mo, and Haotong Zhang. 2025. “Segrnn: Segment Recurrent Neural Network for Long-Term Time Series Forecasting.” IEEE Internet of Things Journal.
13. Wu, Haixu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. 2021. “Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting.” Advances in Neural Information Processing Systems 34: 22419–30.
14. Wu, Haixu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. 2023. “TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis.” International Conference on Learning Representations.
15. Qiu, Xiangfei, Xingjian Wu, Yan Lin, Chenjuan Guo, Jilin Hu, and Bin Yang. 2025. “Duet: Dual Clustering Enhanced Multivariate Time Series Forecasting.” Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining v. 1, 1185–96.