ORANGETREE: A Linear and Tree-based Time Series Forecasting Model Supporting Multiple Input and Output Lengths

Yiqi Tang[1,2], Zhichen Lai[3], Yuxuan Yang[1], Dalin Zhang[4], Jinpeng Chen[5], Gang Chen[1], Huan Li[1,2]✉

[1]State Key Laboratory of Blockchain and Data Security, Zhejiang University
[2]Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security
[3]College of Computer and Data Science, Fuzhou University
[4]Space Information Research Institute, Hangzhou Dianzi University
[5]School of Computer Science, Beijing University of Posts and Telecommunications

{tang_yiqi, yangyuxuan, cg, lihuan.cs}@zju.edu.cn
lzc@fzu.edu.cn, zhangdalin@hdu.edu.cn, jpchen@bupt.edu.cn

Abstract

Time series forecasting (TSF) models suffer from a critical rigidity: the inability to handle Multiple Input and Output Lengths (MIOL) within a single architecture. Current workarounds, such as padding or autoregression, often incur computational redundancy or suffer from error accumulation. We propose ORANGETREE, a linear and tree-based architecture that decouples model parameters from sequence lengths. It employs a Segment Tree Encoder and an Inverse Tree Decoder to translate multi-length input history into multi-length output predictions within a unified framework. Crucially, a Range Weighter and a Feature Fuser bridge these components, dynamically selecting and fusing relevant ranges into optimal contexts for forecasting. Experiments on six benchmarks confirm ORANGETREE achieves the SOTA performance. In MIOL settings, it matches the accuracy of length-fixed models without substantial accuracy degradation or a substantial increase in inference latency typical of heuristic methods. Code is available at `https://github.com/Tang-Yiqi/OrangeTree`.

1 Introduction

Time series forecasting (TSF) is a cornerstone of modern decision systems, underpinning critical domains like energy, transportation, and finance [1]. While the transition from statistical baselines to deep architectures [2], ranging from Transformers-based [3, 4] to Linear-based models [5, 6], has yielded substantial accuracy gains, the field remains entrenched in a paradigm of *structural rigidity*.

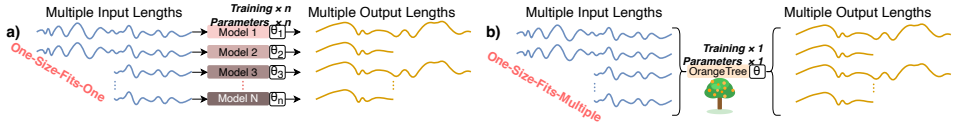


Figure 1: Paradigm Shift in TSF: (a) The traditional rigid approach requires training distinct models for each input-output configuration. (b) The proposed ORANGETREE paradigm enables a single model to natively support MIOL during inference.

The defining limitation of SOTA architectures is their inability to handle **Multiple Input and Output Lengths (MIOL)** natively. As shown in Figure 1(a), the mainstream paradigm enforces a "One-

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Size-Fits-One" constraint: input and output lengths must be permanently fixed at initialization. This rigidity creates a severe friction between model design and real-world deployment. In dynamic environments (Figure 1(b)), systems often need short contexts for fast inference, yet long contexts for accurate long-term planning. To bridge this gap, practitioners resort to inefficient ensembles of length-specific models, wasting storage and management overhead, or crude heuristics like padding and trimming [7], leading to computational redundancy and extraneous noise.

The root of this rigidity lies in the entanglement of model parameters with sequence lengths. Recurrent architectures, while theoretically flexible, suffer from error accumulation over long horizons [8]. Conversely, modern Linear and Transformer models use fixed-size embeddings and static prediction heads, tightly coupling parameters to sequence length. To adapt these rigid structures to MIOL, external workarounds (such as padding and trimming [7]) are required, but these lead to wasted computational resources or information loss (see our discussions in Section 2.1).

In this work, we challenge this status quo by posing a pivotal question: *Is it possible to design a TSF architecture that, through a single training process, supports multiple input and output lengths during inference?*

We answer affirmatively and propose ORANGETREE, a linear and tree-based model that delivers native flexibility for TSF. Its core intuition is to view a time series as a set of ranges rather than a fixed sequence. ORANGETREE derives its name from its ability to assess the importance of all ranges (All Range $\Rightarrow$ Orange) and to perform representation extraction and prediction using a tree structure.

ORANGETREE introduces a structural paradigm shift through four key mechanisms. First, we replace fixed linear projections with a **Segment Tree Encoder** grounded in *binary lifting*. This allows the model to sparsely encode variable-length history into multi-scale representations by sharing aggregation weights, decoupling parameters from input length. Second, inspired by Pascal's Triangle, we introduce a **Range Weighter** that scores input range importance layer-by-layer. Then, the **Feature Fuser** orchestrates the Encoder by range importance for the extraction and fusion of the top-K range representations. Finally, an **Inverse Tree Decoder** synthesizes the forecast by dynamically generating leaf nodes to match any desired output length. Our extensive evaluation shows that ORANGETREE does not merely solve the MIOL problem — it does so without substantial performance degradation. To draw an analogy: in a running competition, athletes train for specific distances, just as a short-distance runner usually cannot win a long-distance race. In machine learning, a single parameter set usually struggles to adapt to diverse scenarios (MIOL). In this work, we train an "all-around athlete" (ORANGETREE trained on MIOL) that not only rivals the "single-event champion" (ORANGETREE trained on fixed lengths) trained for each length, but also outperforms existing length-specific SOTAs.

Our contributions are summarized as follows:

- We propose ORANGETREE, a novel tree-based forecasting framework that natively supports multiple input and output lengths (MIOL) without padding or re-training.
- We design a Segment Tree Encoder and an Inverse Tree Decoder grounded in *binary lifting*. Paired with a hierarchical Range Weighter and a top-K Feature Fuser, these structures dynamically generate multi-granularity representations to construct optimal contexts for specified output lengths.
- Extensive experiments confirm that ORANGETREE establishes a new SOTA on standard benchmarks while natively resolves the MIOL challenge, without parameter inflation, computational redundancy, or severe performance degradation.

2 Preliminaries

2.1 Related Work

Time series forecasting has evolved from statistical methods to deep learning paradigms [2]. Recent advances primarily focus on forecasting accuracy and efficiency, but the ability to natively support MIOL remains rarely addressed. This section reviews representative methods and analyzes their structural limitations in multi-length modeling.

Structural Rigidity in TSF Architectures. With the proposal of the attention mechanism [9], a surge of Transformer-based TSF models have emerged. Early Transformer variants, such as Informer [3] and Pyraformer [4], focused on reducing complexity in long-term dependency modeling

through ProbSparse self-attention and hierarchical graph structures, respectively. Recent methods, like PatchTST [10] and iTransformer [11], shift the paradigm toward patch-based representation and channel-level attention to improve accuracy. During this period, the emergence of DLinear [5], an extremely simple linear model, completely broke the years-long "monopoly" of Transformer structures in TSF, proving that Linear models are also capable of TSF tasks. Subsequently, numerous Linear TSF models have emerged, such as TiDE [6], TSMixer [12], SOFTS [13], etc., improving interactions between time steps and channels. Therefore, structurally, modern TSF models can be divided into two main branches: Transformer models and Linear models.

Although Transformer and Linear models in modern TSF differ sharply in structure, they share a trait: modeling length is highly coupled with model parameters. Existing Linear models essentially rely on linear transformations like $\mathbf{Y} = \mathbf{W}\mathbf{X} + \mathbf{B}$, where the dimensions of the weight matrix $\mathbf{W}$ and bias $\mathbf{B}$ are hard-coded to specific output lengths. Similarly, Transformer models typically depend on fixed-length positional encodings and static embedding layers on the input side, while employing single or multi-layer linear forecasting heads on the output side. Furthermore, some niche architectures, such as CNN-based (e.g., TCN [14]) and State-Space-based (e.g., S-Mamba [15]), also lose the flexibility to vary the modeling length due to the necessary linear transformations on their input and output sides. Consequently, most modern TSF models cannot natively achieve MIOL.

Multiple-Length Modeling in Neural Networks. Although variable-length modeling capabilities are well-established in the NLP and CV, transferring these paradigms to the TSF domain remains challenging. In NLP, variable-length modeling is achieved via autoregressive Transformer structures [9]. However, for continuous time series, this method leads to significant cumulative errors and delays compared to non-autoregressive approaches. Similarly, in CV, dynamic resolution image modeling is often facilitated by AdaPool [16]. However, this approach may disrupt the complex structures of time series and dependencies among channels.

Thus, TSF lacks a dedicated solution for MIOL. Early methods like RNN [17] and LSTM [18] theoretically support MIOL through recurrent connections. However, inherent issues such as difficulty in modeling long-term dependencies and error accumulation [8] cause these models to lag far behind modern TSF models in forecasting accuracy. To circumvent structural rigidity, existing methods employ heuristic adapters like Padding and Autoregression [7], but these all result in varying degrees of resource waste or performance loss. Recent methods, such as HDMixer [19], use linear interpolation before input modeling, theoretically supporting multiple input lengths, but still employ fixed forecasting heads on the output side. Additionally, some Foundation Model-based methods theoretically support MIOL, like LLMTIME [20], LLM4DyG [21], and LSTPrompt [22]. However, these methods incur high computational cost and latency.

Unlike these compromise methods, ORANGETREE introduces a linear and tree-based architecture that natively decouples parameters from sequence length, providing the first end-to-end and unified MIOL solution for the TSF field.

2.2 Problem Statement

Formally, let $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ denote a multivariate time series, where $\mathbf{x}_t \in \mathbb{R}^C$ represents the observations of C variables (channels) at time step t. Given a historical look-back window of length L, denoted as $\mathbf{X}_{in} = [\mathbf{x}_{t-L+1}, \ldots, \mathbf{x}_t] \in \mathbb{R}^{L\times C}$, the canonical TSF task aims to predict the subsequent forecast horizon of length T, denoted as $\mathbf{X}_{out} = [\mathbf{x}_{t+1}, \ldots, \mathbf{x}_{t+T}] \in \mathbb{R}^{T\times C}$.

The MIOL Objective. Unlike standard TSF formulations where L and T are fixed hyperparameters binding the model architecture, the MIOL task requires learning a unified mapping $f_\theta : \mathbb{R}^{L\times C} \to \mathbb{R}^{T\times C}$. Crucially, the parameter set θ must be invariant to the lengths, i.e., $\dim(\theta) \perp \{L, T\}$.

In this work, we define the MIOL objective over a discrete set of valid configurations $\mathcal{S} = \{(L_n, T_n)\}_{n=1}^{N}$. The model is trained to minimize the cumulative error across all pairs in $\mathcal{S}$ and then supports inference for any $(L, T) \in \mathcal{S}$ without retraining. We adopt this discrete strategy for two key reasons: 1) in real-world operational environments (e.g., cloud-based forecasting services), varying demands typically cluster around specific, standardized horizons (e.g., short-term vs. long-term), thus discrete configurations cover the vast majority of real-world needs; 2) learning dependencies over a continuous length spectrum introduces excessive variance, making convergence difficult. Restricting the domain to a discrete set $\mathcal{S}$ ensures peak performance on every supported configuration.

3 The ORANGETREE Design

Figure 2 illustrates the ORANGETREE architecture, designed to decouple model parameters from sequence lengths. The framework processes multivariate time series through four linear modules:

1. ***Segment Tree Encoder*** (Section 3.1) for representation: A hierarchical encoder that projects the input series into a multi-scale tree structure. By employing shared linear layers across nodes, it extracts channel-dependent embeddings for *each* possible sub-range $[i : j]$ of the historical window, independent of the input length L.
2. ***Range Weighter*** (Section 3.2) for importance: A global context evaluator inspired by Pascal's Triangle. It operates on a channel-independent view to score the importance across time series ranges, acting as a linear attention mechanism to identify the most predictive historical ranges.
3. ***Feature Fuser*** (Section 3.3) for selection and fusion: An aggregation module bridging the Segment Tree Encoder and Range Weighter. It uses the importance scores to select the most relevant ranges from the Encoder's tree and fuses them into a compact, channel-dependent hidden state.
4. ***Inverse Tree Decoder*** (Section 3.4) for prediction: A dual-branch decoding architecture. It combines the channel-dependent (Main Decoder) and channel-independent (Residual Decoder) representations via an inverted tree, dynamically expanding depth to match the output length T.

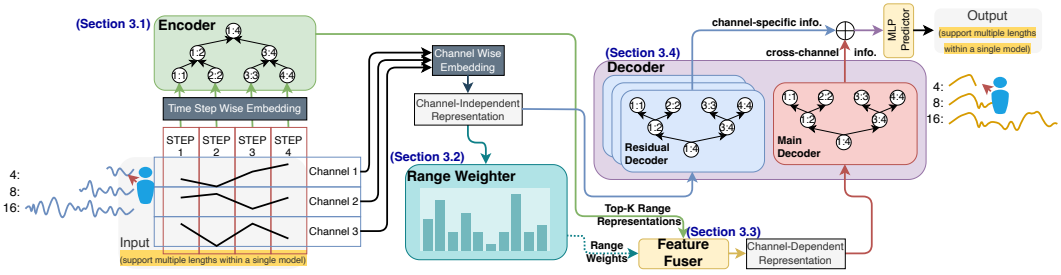


Figure 2: ORANGETREE architecture with a toy example ($C = 3, L = 4$ for an input): Encoder aggregates time steps into range representations (e.g., 1-2, 2-4, and 1-4) using a linear layer; Range Weighter and Feature Fuser compute importance across ranges and fuse the most informative features; Decoder produces variable-length forecasts by expanding parallel Main and Residual inverse trees.

The key innovation enabling MIOL is the *binary lifting* nature of both the Segment and Inverse Trees. Since the linear parameters (θ) are defined at the node level rather than the sequence level, ORANGETREE can adapt its depth and breadth for any given (L, T) pairs during inference.

3.1 Segment Tree Encoder

To address the rigidity of fixed-size embedding layers (where $\mathbf{W} \in \mathbb{R}^{D \times L}$), we propose a hierarchical architecture inspired by the classic segment tree data structure. Unlike standard linear models that treat the input history as a flat vector, our encoder projects the time series into a recursive tree structure. This design yields two critical properties: (1) *Length Invariance*, as parameters are shared across all tree nodes independent of the total sequence length L; and (2) *Multi-Scale Representation*, enabling the efficient extraction of representations for any range $[i : j]$ in $\mathcal{O}(\log L)$ time.

Recursive Representation Extraction. Given a multivariate input $\mathbf{X} = \{x_1, \ldots, x_L\}$ with $x_t \in \mathbb{R}^C$, the encoder builds the tree bottom-up. First, each time step is linearly projected into leaf nodes:

$$\mathbf{h}_t = \text{Linear}^{(0)}(\mathbf{x}_t) \in \mathbb{R}^D, t = 1 \sim L. \tag{1}$$

Higher-level nodes are then recursively aggregated from their children. For a parent node covering the range $[i : j]$, its representation $\mathbf{h}_{[i:j]}$ is derived from its left child $\mathbf{h}_{\text{left}}$ and right child $\mathbf{h}_{\text{right}}$ via a shared aggregation layer:

$$\mathbf{h}_{\text{parent}} = \text{Linear}^{ag}([\mathbf{h}_{\text{left}}; \mathbf{h}_{\text{right}}]) \in \mathbb{R}^D, \tag{2}$$

where $[\cdot;\cdot]$ denotes concatenation. Crucially, because $\mathbf{h}_{\text{parent}}$ maintains the same dimension D as its children, the Linear^{ag} in Eq. 2 is reused across all layers of the tree. This parameter sharing is the mechanism that decouples the model size from the input length L (see Figure 3(b)).

To represent an arbitrary range $[u : v]$, ORANGETREE identifies its *canonical covering set*, the minimal collection of disjoint, pre-computed nodes spanning the range. Figure 3(c) illustrates three retrieval patterns: direct access for complete subtrees (e.g., $[1 : 4]$ via the root), hierarchical composition for mixed ranges (e.g., $[2 : 4]$ via leaf $[2 : 2]$ and node $[3 : 4]$), and boundary-spanning aggregation (e.g., $[2 : 3]$ via leaves $[2 : 2]$ and $[3 : 3]$). Details in Appendix D.1.1.

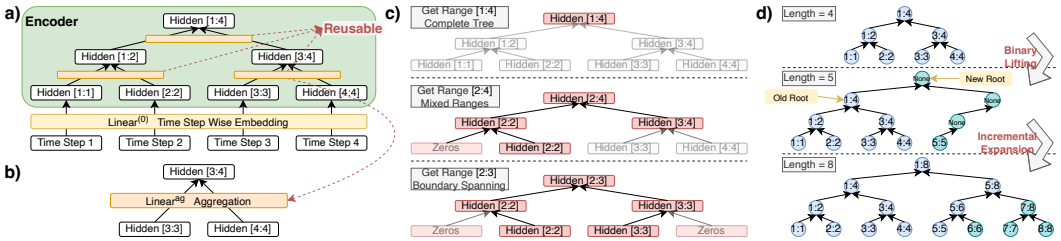


Figure 3: Mechanics of the Segment Tree Encoder. a & b) The encoder recursively aggregates time steps using a shared linear layer, ensuring length-independent parameters; c) Arbitrary ranges' representations like $[2 : 4]$ are efficiently extracted by stitching together the minimal set of pre-computed nodes; d) The tree dynamically expands via *binary lifting* to fit variable input lengths.

Dynamic Expansion via *Binary Lifting*. A core challenge in MIOL is handling L without retraining. We solve this via a *Binary Lifting* strategy. The tree is not initialized with a fixed depth; instead, it dynamically expands to the smallest depth l such that $2^l \geq L$. As shown in Figure 3(d), when the input length increases (e.g., from $L = 4$ to $L = 5$), the model simply adds new nodes and potentially a new root, reusing the existing Linear^{ag} for the new connections. This ensures the model capacity scales naturally with input length while maintaining $\mathcal{O}(L)$ space complexity.

Generalization to Multi-Branch Trees. While binary trees are efficient, deep trees can suffer from optimization difficulties (e.g., vanishing gradients). To mitigate this, we generalize the encoder to a B-branch segment tree, where each parent aggregates B children. This introduces a trade-off: larger B reduces tree depth ($\log_B L$) and facilitates gradient flow, but increases the parameter count of the aggregation layer ($\mathbb{R}^{B \cdot D} \to \mathbb{R}^D$). We treat B as a structural hyperparameter to balance expressivity and efficiency. See the hyperparameter settings for B in Appendix D.2.

3.2 Range Weighter

To distinguish informative historical patterns from noise, we introduce a global attention mechanism that evaluates the predictive importance across ranges $[i : j]$ within the input window. As shown in Figure 4, this weighter employs a resolution normalization strategy to decouple the scoring parameters from the input length L, and a hierarchical structure inspired by Pascal's Triangle to efficiently propagate global context to local ranges.

Resolution Normalization via Interpolation. To ensure the weighting is invariant to L, we first project the input $\mathbf{X} \in \mathbb{R}^{L \times C}$ into a fixed-resolution hidden space. We sample M fixed points $\{\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_M\}$ via linear interpolation, where M is a structural hyperparameter. Each channel is then converted to a channel-independent embedding:

$$\mathbf{e}_c = \text{Linear}^{\text{RM}}(\tilde{\mathbf{x}}_{:,c}) \in \mathbb{R}^D, \quad c = 1 \sim C, \tag{3}$$

where $\tilde{\mathbf{x}}_{:,c} \in \mathbb{R}^M$ denotes the interpolated series of channel c. The vector $\mathbf{e}_c$ encapsulates the global representation of the channel and serves as the channel-independent representation in Decoder (see Residual Decoder in Section 3.4).

Root Node Initialization. Since it only scores importance, we project the high-dimensional channel embedding $\mathbf{e}_c$ into a lightweight hidden space $D' \ll D$:

$$\mathbf{g}^{(c)}_{[1:L]} = \text{Linear}^{\text{root}}(\mathbf{e}_c) \in \mathbb{R}^{D'}, \tag{4}$$

which then acts as the root node of the Pascal structure, representing the hidden state of the full input range $[1 : L]$.

Pascal-Style Hierarchical Scoring. A top-down generation process inspired by Pascal's Triangle derives weights for all ranges. From $[1 : L]$, we recursively generate hidden states for smaller

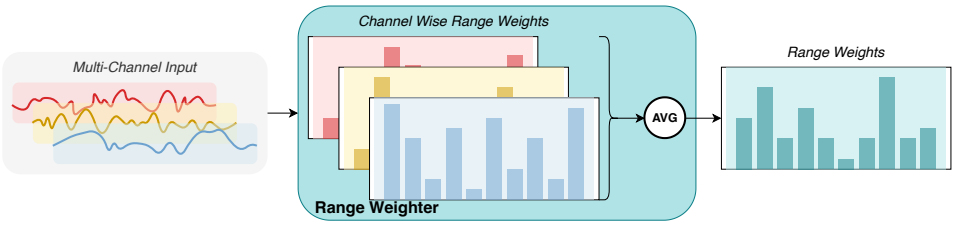


Figure 4: Overview of the Range Weighter.

ranges by "shaving off" time steps from the boundaries. Specifically, the representation for $[i:j]$ is aggregated from its two "parent" ranges, $[i-1:j]$ (extended left) and $[i:j+1]$ (extended right):

$$\mathbf{g}_{[i:j]}^{(c)} = \text{Linear}^{\text{pascal}}([\mathbf{g}_{[i-1:j]}^{(c)}; \mathbf{g}_{[i:j+1]}^{(c)}]) \in \mathbb{R}^{D'}. \tag{5}$$

This recursive formula allows for assessing a local range's importance relative to its broader context. Finally, range scalar scores are averaged across channels to obtain a unified global weight:

$$w_{[i:j]} = \frac{1}{C}\sum_{c=1}^{C} w_{[i:j]}^{(c)}, w_{[i:j]}^{(c)} = \text{Linear}^{\text{vote}}(\mathbf{g}_{[i:j]}^{(c)}). \tag{6}$$

An example is provided in Appendix D.1.2. These weights are subsequently used by the Feature Fuser to select the most predictive ranges from the Encoder.

3.3 Feature Fuser

The fuser serves as a sparse filter, aggregating the most predictive historical ranges. As illustrated in Figure 5, guided by the importance weights $\{w_{[i:j]}\}$ from the Range Weighter, it selects the top-K (see the hyperparameter settings for K in Appendix D.2) ranges and retrieves their corresponding representations $\{\mathbf{h}_k\}_{k=1}^{K}$ from the Segment Tree Encoder (Section 3.1). The fused channel-dependent representation is:

$$\mathbf{h}_{\text{fused}} = \text{GELU}\left(\sum_{k=1}^{K} \text{Linear}^{\text{RG}}(w_k \mathbf{h}_{\text{range-}k})\right) \in \mathbb{R}^{D}, \tag{7}$$

where w_k and $\mathbf{h}_{\text{range-}k}$ are the weight and representation of the k-th selected range, respectively.

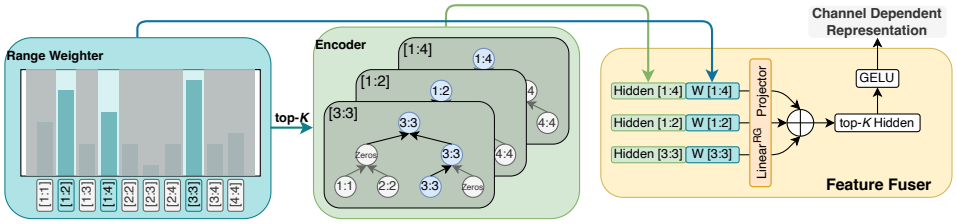


Figure 5: Architecture of the Feature Fuser, combining the Range Weighter and Encoder to produce a channel-dependent representation.

3.4 Inverse Tree Decoder

Mirroring the encoder's hierarchical structure, our decoder maps the compressed hidden state back to a desired forecast output length T by dynamically expanding a root representation into T individual time steps. The decoder adopts a dual-branch structure (Figure 6(a)), consisting of a channel-dependent Main Decoder and a channel-independent Residual Decoder. This design effectively balances cross-channel information with channel-specific information, an approach validated by existing studies [11, 13] as well as our ablation studies (Section 4.4). Unlike those fixed-size forecasting heads $\mathbb{R}^D \Rightarrow \mathbb{R}^T$, our Decoder decouples model parameters from T by defining the transformation at the node level rather than the sequence level.

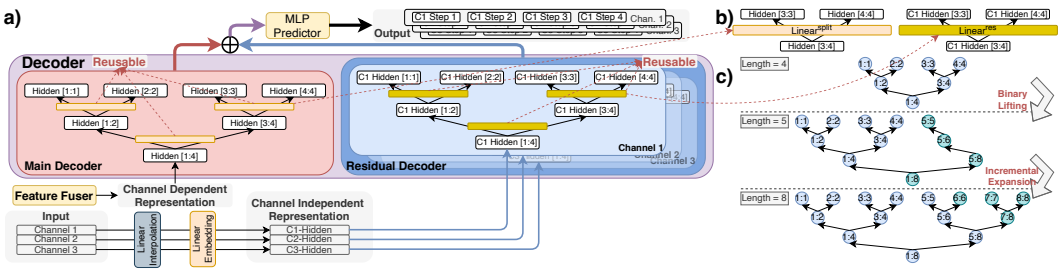


Figure 6: Structure of the Inverse Tree Decoder. a) Main architecture; b) Representation decomposition method; c) Variable output length.

Main Decoder: Cross-Channel Generation. The Main Decoder is responsible for generating the shared temporal dynamics common to all channels. Starting from the fused representation $\mathbf{z}_{\text{parent}}^{(0)} = \mathbf{h}_{\text{fused}} \in \mathbb{R}^D$ (Eq. 7), it recursively splits the parent node into two child nodes at each layer:

$$\left[\mathbf{z}_{\text{left}}^{(l)}; \mathbf{z}_{\text{right}}^{(l)}\right] = \text{Linear}^{\text{split}}\left(\mathbf{z}_{\text{parent}}^{(l-1)}\right) \in \mathbb{R}^{2D}. \tag{8}$$

Crucially, the splitting layer $\text{Linear}^{\text{split}} : \mathbb{R}^D \to \mathbb{R}^{2D}$ is shared across all layers (Figure 6(b)). By iterating this process for $H = \lceil \log_2 T \rceil$ layers, we generate T leaf nodes $\{\mathbf{z}_1^{(H)}, \ldots, \mathbf{z}_T^{(H)}\}$ that capture cross-channel interactions for each future time step.

Residual Decoder: Channel-Specific Generation. Complementing the Main Decoder, the Residual Decoder independently reconstructs the unique representations of each channel. It takes the channel-specific embedding $\mathbf{e}_c$ (Eq. 3) as its root $\mathbf{r}_c^{(0)}$ and applies an independent splitting process:

$$\left[\mathbf{r}_{c,\text{left}}^{(l)}; \mathbf{r}_{c,\text{right}}^{(l)}\right] = \text{Linear}^{\text{res}}\left(\mathbf{r}_{c,\text{parent}}^{(l-1)}\right) \in \mathbb{R}^{2D}. \tag{9}$$

This yields T channel-specific leaf nodes $\{\mathbf{r}_{c,1}^{(H)}, \ldots, \mathbf{r}_{c,T}^{(H)}\}$. Since this branch processes each channel in isolation, it preserves the unique distributional properties of individual channels.

Variable-Length Output Synthesis. To support variable output lengths T, both decoders utilize a *binary lifting* strategy for expansion. As shown in Figure 6(c), the tree depth H is dynamically adjusted to accommodate the requested length T without structural re-initialization. The final forecast for time step t in channel c combines the shared and specific information:

$$\mathbf{o}_{c,t} = (\mathbf{z}_t^{(H)} + \mathbf{r}_{c,t}^{(H)}) \in \mathbb{R}^D,\ c = 1 \sim C; t = 1 \sim T. \tag{10}$$

Here, $\mathbf{z}_t^{(H)}$ injects cross-channel information, while $\mathbf{r}_{c,t}^{(H)}$ provides channel-specific information. A lightweight MLP predictor then projects this state to the final scalar prediction:

$$\hat{y}_{c,t} = \text{MLP}(\mathbf{o}_{c,t}), \quad c = 1 \sim C; t = 1 \sim T. \tag{11}$$

Multi-Branch Extension. Analogous to the Encoder, the Decoder can be generalized from a binary tree to a B-branch tree. This allows for shallower trees when modeling long horizons, mitigating gradient propagation issues. The branching factor B is a structural hyperparameter that must be consistent across the Main and Residual branches but can differ from the Encoder's configuration. See the hyperparameter settings for B in Appendix D.2.

3.5 Loss Function and Model Training

Standard TSF models typically adopt either Mean Absolute Error (MAE) for outlier robustness or Mean Squared Error (MSE) for high-sensitivity penalization [23]. To leverage the strengths of both, we draw on DGraFormer [24] and adopt a hybrid loss:

$$\mathcal{L} = \alpha \cdot \mathcal{L}_{\text{MAE}} + (1 - \alpha) \cdot \mathcal{L}_{\text{MSE}}, \quad \alpha \in [0, 1]. \tag{12}$$

Empirically, we observe $\alpha = 0.8$ yields the optimal trade-off (See Appendix D.2.4 for experimental results.). During training, we optimize the parameters by minimizing $\mathcal{L}$ over the discrete set of valid input-output configurations $\mathcal{S} = \{(L_k, T_k)\}$.

4 Experiments

We evaluate the efficacy and versatility of ORANGETREE across two primary experimental phases: (i) **Fixed-Input Forecasting** (Section 4.2), which encompasses both standard fixed-length benchmarking (S1) and multi-output flexibility (S2). (ii) **Full MIOL Forecasting** (Section 4.3), validating the model's native capability to handle multiple input and output lengths simultaneously (S3). We further conduct ablation studies to quantify each component's contribution (Section 4.4), report parameter counts, and describe representation reuse (Section 4.5).

4.1 Experimental Setup

Implementation Details and Training Protocol. All experiments are implemented in PyTorch 2.5 on a server with an NVIDIA RTX 5000 Ada GPU. Details in Appendix D.4.

Datasets and Metrics. We utilize six widely adopted benchmarks: ETTh1, ETTh2, ETTm1, ETTm2 (electricity domain), Weather (meteorological domain), and Solar (energy domain). Following standard conventions [11], we apply a 6:2:2 train-validation-test split for the ETT datasets and a 7:1:2 split for Weather and Solar. We report performance using MSE and MAE.

We compare ORANGETREE against 11 competitive baselines from four families [1]: (1) **Linear-based**: DLinear [5], TiDE [6], TSMixer [12], and SOFTS [13]; (2) **Transformer-based**: PatchTST [10], iTransformer [11], DGraFormer [24], TQNet [25], and CASA [26]; and (3) **State-Space-based**: S-Mamba [15]; (4) **Recurrent-Connection-based**: LSTM [18].

4.2 Fixed-Input Forecasting (Settings S1 and S2)

We adhere to the standard evaluation protocol, fixing the input lookback at $L = 96$ and evaluating performance across four distinct prediction horizons $T \in \{96, 192, 336, 720\}$. To investigate the trade-off between specialization and flexibility, we compare two training strategies:

- **Idealized Specialist (ORANGETREE-IS)**: We train independent ORANGETREE models for each specific output length T. This matches the standard baseline approach and maximizes specialization for the setting S1.
- **Horizon-Unified (ORANGETREE-HU)**: We train a single ORANGETREE model to jointly minimize error across all four horizons simultaneously. This validates the model's native capability to support variable outputs without retraining (supporting the setting S2).

Table 1: Multivariate forecasting performance ($L = 96$, averaged over 4 output horizons $T \in \{96, 192, 336, 720\}$). We compare ORANGETREE-IS and all baselines (trained specifically for each horizon T, following Setting S1) against ORANGETREE-HU (a single unified model trained jointly to support all 4 horizons, following Setting S2). **Bold** and underlined values indicate the best and second-best results, respectively. Full results are provided in Appendix G.

	ORANGETREE -IS (S1)		ORANGETREE -HU (S2)		DGraF. (2025)		CASA (2025)		TQNet (2025)		S-Mamba (2025)		SOFTS (2024)		iTrans. (2024)		PatchTST (2023)		TSMixer (2023)		TiDE (2023)		DLinear (2023)		LSTM (1997)	
Dataset	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	**0.365**	0.379	0.366	**0.377**	0.379	0.394	0.386	0.394	0.377	0.393	0.398	0.405	0.399	0.405	0.401	0.408	0.397	0.406	0.448	0.447	0.409	0.411	0.403	0.407	1.181	0.825
ETTm2	**0.272**	**0.316**	0.275	0.319	0.284	0.326	0.276	0.319	0.277	0.323	0.288	0.332	0.287	0.330	0.288	0.332	0.287	0.330	0.289	0.333	0.358	0.404	0.350	0.401	2.562	1.262
ETTh1	**0.428**	**0.425**	**0.428**	0.427	0.442	0.432	0.438	0.433	0.441	0.434	0.455	0.450	0.449	0.443	0.454	0.448	0.453	0.446	0.463	0.452	0.541	0.507	0.456	0.452	1.212	0.843
ETTh2	**0.366**	**0.394**	0.372	0.396	0.372	0.398	0.373	0.398	0.378	0.402	0.381	0.405	0.373	0.400	0.383	0.406	0.385	0.410	0.402	0.417	0.611	0.550	0.559	0.515	3.283	1.431
Weather	**0.234**	**0.261**	0.239	0.265	0.249	0.271	0.243	0.267	0.242	0.269	0.251	0.276	0.255	0.278	0.258	0.278	0.256	0.279	0.256	0.278	0.270	0.320	0.265	0.317	0.406	0.401
Solar	0.204	**0.239**	0.207	0.242	0.216	**0.239**	0.222	0.244	**0.198**	0.256	0.240	0.272	0.229	0.256	0.233	0.262	0.236	0.266	0.261	0.297	0.347	0.417	0.330	0.401	0.498	0.436

As shown in Table 1, the horizon-specific ORANGETREE-IS establishes a new SOTA. Crucially, the horizon-unified ORANGETREE-HU incurs negligible performance loss (and occasionally exceeds ORANGETREE-IS), despite optimizing for multiple output lengths simultaneously. Although LSTM [18] supports arbitrary lengths, their performance significantly lags behind other modern models. This resilience confirms that our Inverse Tree Decoder effectively decouples model parameters from output

[1] We exclude Foundation Model-based methods (e.g., LLMTime [20] and LSTPrompt [22]) to focus on *end-to-end models*. This is because a single Foundation Model's resource footprint is sufficient to deploy hundreds of the standard TSF baselines and the ORANGETREE. Such a comparison runs counter to the core objective of this work. All baselines are detailed in Appendix C.

length, enabling flexible deployment without substantial accuracy degradation typical of heuristic adaptations.

4.3 Full MIOL Forecasting (Setting S3)

To validate the model's adaptability in dynamic deployment environments, we evaluate performance under the Full MIOL setting, where both input lengths L and output lengths T vary during inference.

We compare our natively unified ORANGETREE-MIOL against two categories of baselines: (1) ORANGETREE-IS: a separate model is trained specifically for every distinct (L, T) pair. (2) Heuristic Adapters: Traditional strategies that force fixed-size models to handle variable lengths without training multiple times. For input adaptation, we apply *Trimming* or *Padding*. For output adaptation, we employ *Autoregression* or *Trimming*. This yields four combinatorial heuristics: TRIM-REG, TRIM-TRIM, PAD-REG, and PAD-TRIM. We provide more details in Appendix H.

Table 2 details the trade-offs across accuracy (MSE/MAE), inference latency, and memory usage. Remarkably, the unified ORANGETREE-MIOL matches the accuracy of the specialized ORANGETREE-IS, effectively eliminating the need for multi-model maintenance without substantial performance degradation. In contrast, heuristics incur severe penalties. The input-side TRIM-$\star$ methods achieve low latency but incur significant accuracy loss due to discarding the first half of the input. Meanwhile, the output-side $\star$-REG methods reduce memory usage at the cost of accuracy and higher latency (from iterative forecasting). Among heuristics, PAD-TRIM achieves the best forecasting accuracy by avoiding both input truncation and error accumulation. However, it incurs substantially higher latency and memory usage due to modeling at the maximum length, whereas ORANGETREE-MIOL incurs no such costs. Applying these heuristics to previous SOTA models yields consistent conclusions.[2]

Table 2: Performance trade-offs in the Full MIOL setting (averaged over $L \in \{48, 96\}$ and $T \in \{48, 96\}$). We compare the unified ORANGETREE-MIOL against the idealized specialist ORANGETREE-IS (trained for each (L, T) pair, as the baseline) and four heuristic adapters. *Time* and *MEM* denote inference latency (s) and memory usage (GB). Full results and the results across 8 pairs from $L = \{48, 96, 144\}$ and $T = \{96, 192, 336, 720\}$ are provided in Appendix H.3.

	ORANGETREE -IS (S1)				ORANGETREE -MIOL (S3)				ORANGETREE +TRIM-REG				ORANGETREE +TRIM-TRIM				ORANGETREE +PAD-REG				ORANGETREE +PAD-TRIM			
Dataset	MSE	MAE	Time	MEM	MSE	MAE	Time	MEM	MSE	MAE	Time	MEM	MSE	MAE	Time	MEM	MSE	MAE	Time	MEM	MSE	MAE	Time	MEM
ETTh1	0.353	0.374	0.248	9.71	**0.356**	**0.375**	0.244	9.71	0.370	0.383	0.210	6.48	0.362	0.383	0.121	12.74	0.370	0.389	0.633	6.69	0.359	0.386	0.465	12.91
ETTh2	0.256	0.314	0.261	9.68	**0.258**	**0.315**	0.257	9.66	0.268	0.322	0.214	6.45	0.265	0.316	0.113	12.65	0.263	0.331	0.628	6.90	0.259	0.319	0.414	12.95
ETTm1	0.337	0.354	0.262	9.67	**0.343**	**0.357**	0.270	9.67	0.446	0.412	0.231	6.49	0.403	0.389	0.110	12.64	0.407	0.397	0.720	6.72	0.370	0.374	0.403	12.91
ETTm2	0.153	0.237	0.251	9.76	**0.155**	**0.237**	0.261	9.72	0.167	0.256	0.221	6.47	0.169	0.256	0.125	12.68	0.169	0.262	0.695	6.77	0.167	0.259	0.389	12.91
Weather	0.145	0.182	0.258	13.95	0.147	**0.184**	0.252	13.97	0.192	0.229	0.242	9.39	0.168	0.205	0.116	18.48	0.147	0.189	0.672	9.48	**0.146**	**0.184**	0.400	18.64
Solar	0.243	0.286	0.265	16.83	**0.240**	0.285	0.268	16.91	0.602	0.586	0.257	11.34	0.302	0.336	0.122	22.45	0.244	0.291	0.685	11.39	0.244	**0.281**	0.412	22.49

4.4 Ablation Study

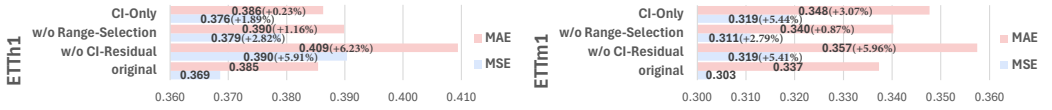


Figure 7: Ablation results with $L = 96, T = 96$ (full results and analysis in Appendix D.3).

To quantify the contribution of individual components, we evaluate 3 variants against the original model (figures in Appendix D.3): (1) w/o CI-Residual: Removes the channel-independent residual branch. This tests whether explicitly modeling shared global trends complements the channel-specific tree representations. (2) w/o Range-Selection: Bypasses the Range Weighter and Feature Fuser, utilizing only the Segment Tree's root node. This verifies the necessity of our multi-scale aggregation compared to using a single global representation. (3) CI-Only: Retains only the channel-independent

[2]As detailed in Appendix I, applying these heuristics to SOTA models (e.g., CASA, DGraFormer) results in even sharper performance drops, confirming that ORANGETREE's resilience stems from its native architectural elasticity (Segment Tree + Inverse Tree) rather than simple data augmentation.

residual structure. This baseline quantifies the performance gain attributed to our tree-based channel-dependent modeling. Figure 7 shows the full ORANGETREE yields the lowest error.

4.5 Parameter Efficiency and Representation Reuse

Parameter efficiency. To illustrate the parameter-sharing benefit of ORANGETREE's native MIOL support, we select four representative baselines: the SOTA models CASA (Transformer-based) and SOFTS (Linear-based), and the classic models PatchTST (Transformer-based) and DLinear (Linear-based). These baselines do not natively support MIOL, and their official implementations use the standard input length $L = 96$ and provide no configurations for other input lengths. We therefore compare them with ORANGETREE-HU, which shares one model across multiple output lengths at the same input length.

As illustrated in Figure 8, we compare both single-model parameter counts and the total required to cover all output lengths. For a single model, ORANGETREE-HU has fewer parameters than CASA and PatchTST, while its comparison with SOFTS varies across datasets. When deploying models for all output lengths, ORANGETREE-HU requires far fewer total parameters than CASA, PatchTST, and SOFTS, while also achieving better average MSE and MAE (Table 1). These results show that ORANGETREE-HU offers both better forecasting performance and an advantage in parameter count over these strong baselines. Although DLinear, an extremely lightweight classic Linear model, has fewer total parameters than ORANGETREE-HU, it lags far behind ORANGETREE and other SOTA models in forecasting performance.

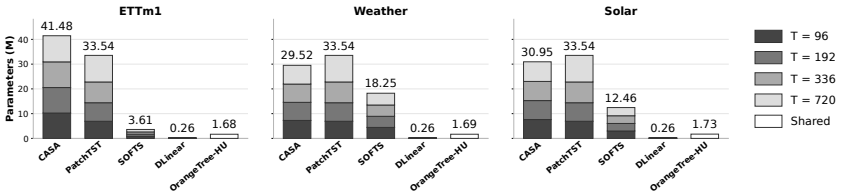


Figure 8: Parameter counts on ETTm1, Weather, and Solar using official baseline hyperparameters with input length $L = 96$ and output lengths $T \in \{96, 192, 336, 720\}$.

Representation reuse. ORANGETREE's native MIOL support not only enables parameter sharing but also allows representation reuse during inference. Length-fixed baselines train a separate model for each (L, T) pair, so they cannot share one model across output lengths even when the input length is the same. As illustrated in Figure 9, a single ORANGETREE model covers multiple trained input-output length pairs (L, T) without switching models, and the representation of each input is shared across its output lengths. For the same input of length L, the Representation Module runs once. The Decoder reuses the representation to generate forecasts for each output length T.

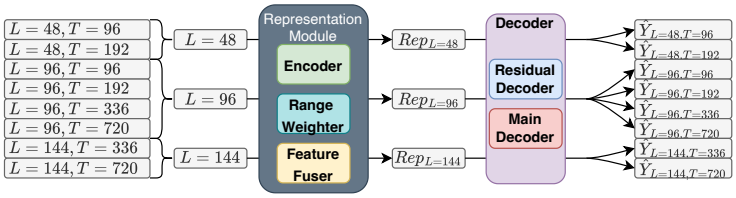


Figure 9: Representation reuse for different input lengths L and output lengths T in one ORANGETREE model.

5 Conclusion

We introduce ORANGETREE, a linear and tree-based architecture decoupling model parameters from sequence length. ORANGETREE achieves state-of-the-art results on fixed-horizon benchmarks and natively resolves the Multiple Input and Output Length (MIOL) challenge, without parameter inflation, computational redundancy, or severe performance degradation. [3]

[3] Limitation and future work are discussed in Appendix E.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62402420), Fundamental Research Funds for Central Universities (Grant No. 226202400109), and Beijing Natural Science Foundation (No. L233034). Huan Li is the corresponding author of this paper.

References

[1] Konstantinos Benidis, Syama Sundar Rangapuram, Valentin Flunkert, Yuyang Wang, Danielle Maddix, Caner Turkmen, Jan Gasthaus, Michael Bohlke-Schneider, David Salinas, Lorenzo Stella, et al. Deep learning for time series forecasting: Tutorial and literature survey. *ACM Computing Surveys*, 55(6):1–36, 2022.

[2] Yushan Jiang, Kanghui Ning, Zijie Pan, Xuyang Shen, Jingchao Ni, Wenchao Yu, Anderson Schneider, Haifeng Chen, Yuriy Nevmyvaka, and Dongjin Song. Multi-modal time series analysis: A tutorial and survey. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 6043–6053, 2025.

[3] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11106–11115, 2021.

[4] Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X Liu, and Schahram Dustdar. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *International Conference on Learning Representations*, 2022.

[5] 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.

[6] Abhimanyu Das, Weihao Kong, Andrew Leach, Shaan K Mathur, Rajat Sen, and Rose Yu. Long-term forecasting with tide: Time-series dense encoder. *Transactions on Machine Learning Research*, 2023.

[7] Agnieszka Bier, Agnieszka Jastrzebska, and Pawel Olszewski. Variable-length multivariate time series classification using rocket: A case study of incident detection. *IEEE Access*, 10:95701–95715, 2022.

[8] Xiaobao Song, Liwei Deng, Hao Wang, Yaoan Zhang, Yuxin He, and Wenming Cao. Deep learning-based time series forecasting. *Artificial Intelligence Review*, 58(1):23, 2024.

[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[10] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations*, 2023.

[11] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2024.

[12] Vijay Ekambaram, Arindam Jati, Nam Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. Tsmixer: Lightweight mlp-mixer model for multivariate time series forecasting. In *Proceedings of the 29th ACM SIGKDD conference on knowledge discovery and data mining*, pages 459–469, 2023.

[13] Lu Han, Xu-Yang Chen, Han-Jia Ye, and De-Chuan Zhan. Softs: Efficient multivariate time series forecasting with series-core fusion. *Advances in Neural Information Processing Systems*, 37:64145–64175, 2024.

[14] Shaojie Bai. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[15] Zihan Wang, Fanheng Kong, Shi Feng, Ming Wang, Xiaocui Yang, Han Zhao, Daling Wang, and Yifei Zhang. Is mamba effective for time series forecasting? *Neurocomputing*, 619:129178, 2025.

[16] Alexandros Stergiou and Ronald Poppe. Adapool: Exponential adaptive pooling for information-retaining downsampling. *IEEE Transactions on Image Processing*, 32:251–266, 2022.

[17] Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

[18] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[19] Qihe Huang, Lei Shen, Ruixin Zhang, Jiahuan Cheng, Shouhong Ding, Zhengyang Zhou, and Yang Wang. Hdmixer: Hierarchical dependency with extendable patch for multivariate time series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 12608–12616, 2024.

[20] Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew G Wilson. Large language models are zero-shot time series forecasters. *Advances in Neural Information Processing Systems*, 36:19622–19635, 2023.

[21] Zeyang Zhang, Xin Wang, Ziwei Zhang, Haoyang Li, Yijian Qin, and Wenwu Zhu. Llm4dyg: can large language models solve spatial-temporal problems on dynamic graphs? In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4350–4361, 2024.

[22] Haoxin Liu, Zhiyuan Zhao, Jindong Wang, Harshavardhan Kamarthi, and B Aditya Prakash. Lstprompt: Large language models as zero-shot time series forecasters by long-short-term prompting. *CoRR*, 2024.

[23] Juan Terven, Diana-Margarita Cordova-Esparza, Julio-Alejandro Romero-González, Alfonso Ramírez-Pedraza, and EA Chávez-Urbiola. A comprehensive survey of loss functions and metrics in deep learning. *Artificial Intelligence Review*, 58(7):195, 2025.

[24] Han Yan, Dongliang Chen, Guiyuan Jiang, Bin Wang, Lei Cao, Junyu Dong, and Yanwei Yu. Dgraformer: Dynamic graph learning guided multi-scale transformer for multivariate time series forecasting. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, pages 3516–3524, 2025.

[25] Shengsheng Lin, Haojun Chen, Haijie Wu, Chunyun Qiu, and Weiwei Lin. Temporal query network for efficient multivariate time series forecasting. In *Forty-second International Conference on Machine Learning*, 2025.

[26] Minhyuk Lee, HyeKyung Yoon, and MyungJoo Kang. Casa: Cnn autoencoder-based score attention for efficient multivariate long-term time-series forecasting. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, 2025.

A Impact Statement

This work contributes to sustainable and accessible AI. By proving that a linear architecture can rival heavy Transformers, ORANGETREE drastically reduces the computational and storage footprint required for high-performance forecasting. This efficiency facilitates deployment on resource-constrained devices, directly supporting Green AI goals by lowering energy consumption. We do not foresee any negative societal impacts arising from this research.

B Ethics Statement

We have ensured that all datasets used in the experiments are publicly available, and we have carefully considered the ethical implications of our work. Therefore, we believe that our research does not present any ethical or moral risks.

C Detailed Descriptions of All Baselines

C.1 Linear Models

- DLinear [5]: It follows the paradigm of seasonal-trend decomposition. It employs two linear layers to predict the seasonal and trend parts separately, which are then summed to produce the final forecast.
- TiDE [6]: It utilizes a Time-series Dense Encoder architecture. TiDE encodes historical time series across all channels using an MLP and then decodes them into future values with another MLP.
- TSMixer [12]: Through hierarchical patch alignment and cross-channel online calibration, it enhances long-sequence and multivariate modeling capabilities, and maintains low computational cost.
- SOFTS [13]: Through the STAR module with an "aggregate-distribute-fuse" mechanism, it achieves channel-independent and channel-dependent modeling, thereby improving performance in multivariate TSF.

C.2 Transformer Models

- PatchTST [10]: It uses a patching mechanism to reduce computational and storage costs, enabling the same window length to capture longer TS dependencies. PatchTST adopts a channel-independent architecture to reduce cross-channel interference. Simultaneously, it also designs a masked reconstruction task to assist the model in learning temporal representations.
- iTransformer [11]: It inverts the operating dimensions of the Transformer's attention and FFN, embedding each channel as a token, thereby using attention to explicitly model cross-channel correlations, and using the FFN to learn intra-channel representations.
- DGraFormer [24]: It employs dynamic correlation-aware graph learning to capture correlations across different time windows, and simultaneously uses a multi-scale temporal transformer to extract temporal patterns from sequences of different scales.
- TQNet [25]: This technique employs periodically shifted learnable vectors as the Q in the attention mechanism to capture global cross-channel patterns, while generating K and V from the original input data to encode local correlations.
- CASA [26]: It adopts a multi-layer CNN-based autoencoder structure, aiming to replace the computation of K and Q in the traditional attention structure, thereby improving prediction performance and reducing complexity.

C.3 State-Space Models

- S-Mamba [15]: It first embeds each timestep of each channel through a linear layer, then utilizes a bidirectional Mamba layer to extract correlations between channels. Finally, it generates the prediction results through a linear layer.

C.4 Recurrent-Connection Models

- LSTM [18]: It is an improved variant of RNN [17] that introduces a memory cell and gating mechanisms to learn both short-term and long-term dependencies. It natively supports arbitrary input and output lengths via its recurrent connections.

D Model Details

D.1 Computation Details

D.1.1 Hierarchical Representation Extraction in the Encoder

Regardless of which range's representation within the historical series is to be extracted, the process ultimately requires aggregation to the root node, as shown in Figure 10. Several examples are illustrated below:

- Getting the representation of the range $[1:4]$: The representation for range $[1:4]$ has been pre-computed, and since $[1:4]$ is the root node, we use it directly, as shown in Figure 10 (a).
- Getting the representation of the range $[2:4]$: We need node $[2:2]$ and node $[3:4]$ to compose this range. However, node $[2:2]$ and node $[3:4]$ cannot be aggregated directly because they belong to different layers. To bring node $[2:2]$ and node $[3:4]$ to the same layer, we need to propagate the representation of node $[2:2]$ to its parent node and pad the nodes that participate in the computation but are not selected with zeros, as shown in Figure 10 (b). The formula is:

$$\mathbf{h}_{[2:4]} = \text{Linear}^{\text{ag}}\left(\left[\text{Linear}^{\text{ag}}\left(\left[\mathbf{h}_{\text{zeros}}\,;\mathbf{h}_{[2:2]}\right]\right);\mathbf{h}_{[3:4]}\right]\right) \in \mathbb{R}^{D}\,. \tag{13}$$

- Getting the representation of the range $[2:3]$: Similarly, as shown in Figure 10 (c), the formula is:

$$\mathbf{h}_{[2:3]} = \text{Linear}^{\text{ag}}\left(\left[\text{Linear}^{\text{ag}}\left(\left[\mathbf{h}_{\text{zeros}}\,;\mathbf{h}_{[2:2]}\right]\right);\text{Linear}^{\text{ag}}\left(\left[\mathbf{h}_{[3:3]};\mathbf{h}_{\text{zeros}}\right]\right)\right]\right) \in \mathbb{R}^{D}\,. \tag{14}$$

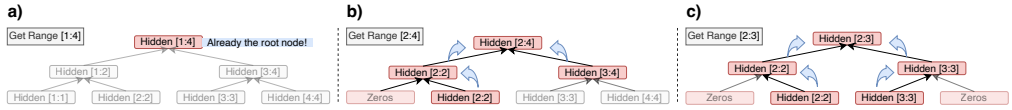


Figure 10: Example: Encoder Aggregation Process.

D.1.2 Weight Calculation in the Range Weighter

When calculating the weight for a specific range, first, the hidden states of that range across all channels are retrieved, as shown in Figure 11 (a). Then, this hidden state is projected through a shared linear layer to obtain the vote (a scalar) of each channel for that range, as shown in Figure 11 (b). Finally, the votes from all channels for that range are summed to obtain the final weight of the range, as shown in Figure 11 (c).

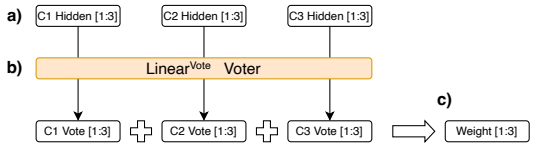


Figure 11: Weight calculation in the Range Weighter (Taking range [1:3] as an example).

D.2 Hyperparameters

D.2.1 The top-K ranges of Feature Fuser

Adjusting top-K typically does not directly fix a specific K. This is because, in the case of MIOL, the number of ranges required for representation differs between longer and shorter inputs. Therefore, we

use a K-Rate to calculate the K for a given input length. For an input of length L, the total number of ranges is:

$$N_{range} = \frac{(L+1) \times L}{2}. \tag{15}$$

The parameter K in top-K is:

$$K = N_{range} \times K\text{-Rate}. \tag{16}$$

D.2.2 Branch Number of Encoder and Decoder

For convenience, we usually synchronize the adjustment of the branch number (B) in both the Encoder and Decoder. A larger B increases the parameter size, which allows the model to capture details. Conversely, a smaller B reduces parameter size and can enhance model robustness. Therefore, selecting an appropriate B during training is essential.

D.2.3 Hyperparameter Settings

Table 3 summarizes the core hyperparameters of ORANGETREE and their search spaces or fixed values. We tune B and K separately for each dataset and input-output length pair (L, T). We search the branch number B over $\{2, 3, 4, 5\}$. For top-K selection, we search either K-Rate over $\{0.1, 0.01, 0.001\}$ or K over $\{4, 16, 64\}$. The hidden size D and the Range Weighter's hidden size D' are fixed at 384 and 32, respectively. The hybrid loss weight is set to $\alpha = 0.8$, assigning weights of 0.8 to MAE and 0.2 to MSE.

Table 3: Core hyperparameters of ORANGETREE.

Hyperparameter	**Meaning**	**Value**
B	Branch Number	Search in $\{2, 3, 4, 5\}$
K	Top-K Selection	Search K-Rate in $\{0.1, 0.01, 0.001\}$ or K in $\{4, 16, 64\}$
D	Hidden Size	384
D'	Hidden Size for the Range Weighter	32
α	$\mathcal{L} = \alpha \cdot \text{MAE} + (1-\alpha) \cdot \text{MSE}$	0.8

D.2.4 Hybrid Loss

The function of the hybrid loss is:

$$\mathcal{L} = \alpha \cdot \mathcal{L}_{\text{MAE}} + (1-\alpha) \cdot \mathcal{L}_{\text{MSE}}, \quad \alpha \in [0, 1]. \tag{17}$$

As shown in Table 4, setting $\alpha = 0.8$ consistently yields the most stable and optimized MSE+MAE improvements across datasets.

Table 4: Effect of α on model performance ($L = 96, T = 96$).

Dataset	$\alpha = 0.0$			$\alpha = 0.2$			$\alpha = 0.4$			$\alpha = 0.6$			$\alpha = 0.8$			$\alpha = 1.0$		
	MSE	MAE	MSE+MAE	MSE	MAE	MSE+MAE	MSE	MAE	MSE+MAE	MSE	MAE	MSE+MAE	MSE	MAE	MSE+MAE	MSE	MAE	MSE+MAE
ETTh1	0.376	0.395	0.771	0.372	0.392	0.764	0.368	0.388	0.756	0.370	0.388	0.758	0.369	0.385	**0.754**	0.382	0.389	0.771
ETTm1	0.301	0.350	0.651	0.299	0.344	0.643	0.299	0.341	**0.640**	0.302	0.340	0.642	0.303	0.337	**0.640**	0.311	0.337	0.648
Weather	0.158	0.219	0.377	0.157	0.205	0.362	0.154	0.202	0.356	0.156	0.198	0.354	0.149	0.187	**0.336**	0.154	0.196	0.350

D.3 Ablation

To validate the contribution of each component in ORANGETREE, we design three architectural variants as illustrated in Figure 12. Each variant systematically removes or isolates specific components to quantify their impact on forecasting accuracy.

As shown in Figure 13, the full ORANGETREE consistently yields the lowest error. The performance degradation in w/o Range-Selection confirms that global representation alone is insufficient,

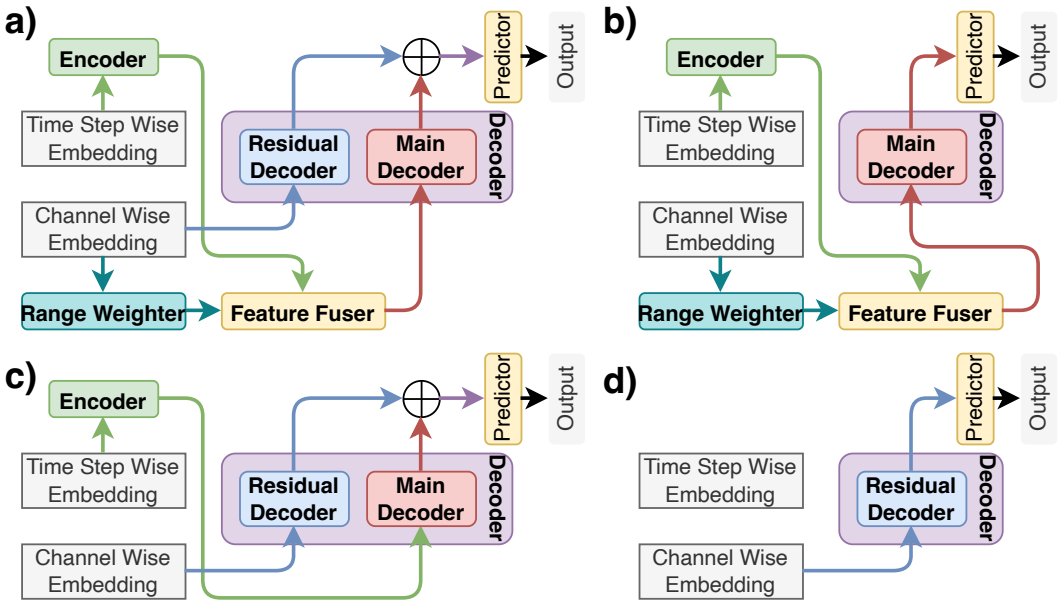


Figure 12: Ablation study: a) original structure; b) w/o CI-Residual; c) w/o Range-Selection; and d) CI-Only.

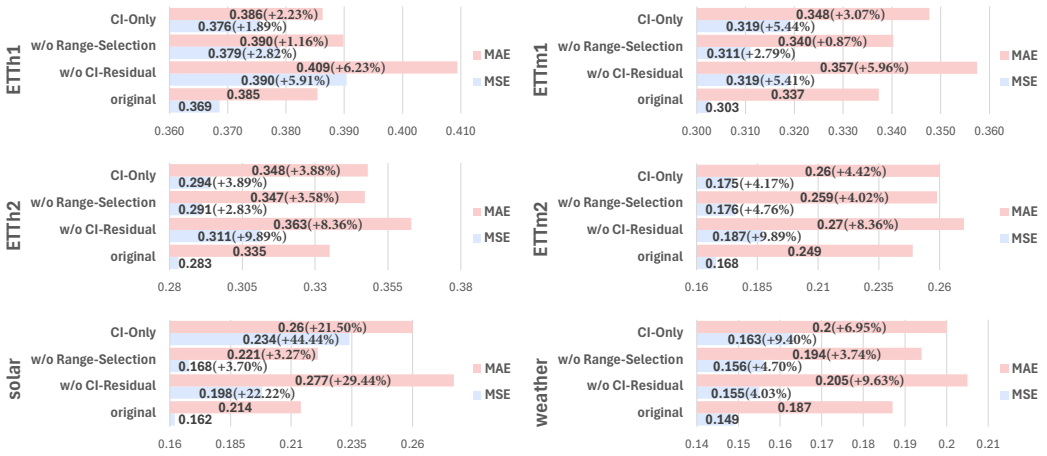


Figure 13: Ablation results ($L = 96$, $T = 96$).

validating our top-K selection mechanism for capturing localized temporal shifts. Furthermore, the significant drop in w/o CI-Residual underscores the importance of the dual-branch design, proving that balancing shared channel dynamics with independent residual correction is critical for robust forecasting.

D.4 Training Details

D.4.1 Hardware and Software

All experiments are conducted on a consistent computing environment to ensure reproducibility. Table 5 summarizes the complete hardware and software specifications.

Table 5: Hardware and software configuration.

Component	Specification
CPU	Intel(R) Xeon(R) w9-3495X
GPU	NVIDIA RTX 5000 Ada (32GB, Ada Lovelace architecture)
NVIDIA Driver	Version 570.195.03
CUDA	Version 12.8
OS	Ubuntu 24.04.3 LTS
Deep Learning Framework	PyTorch 2.5.1+cu121

D.4.2 Variable-Length Training

For scenarios involving variable lengths (Settings S2 and S3), we adopt a training strategy where sequence lengths are uniform within each batch but shuffled across batches. Specifically, we define different input-output length pairs as (L, T) Pair$_n$, where each (L, T) Pair contains a slightly varying number of samples, as shown in Figure 14(a). Subsequently, we create independent batches for each (L, T) Pair (with DropLast $=$ True during training and DropLast $=$ False during validation and testing), as shown in Figure 14(b). Finally, we shuffle all batches and feed them into ORANGETREE for training, as shown in Figure 14(c).

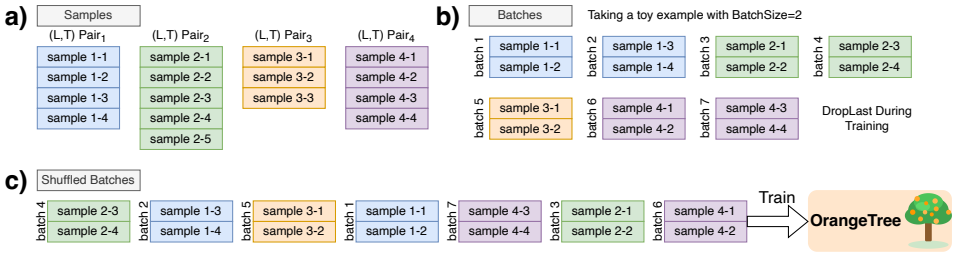


Figure 14: A toy example of variable-length training.

E Limitation and Future Work

Although ORANGETREE theoretically supports arbitrary horizons, we observe a practical "Generalist-Specialist Gap" between fully-arbitrary-length and length-specialized training.

As shown in Table 6, the performance drop of MIOL (S3) compared to IS (S1) is negligible. However, under Arbitrary Length training, a significant performance degradation appears, though it is not entirely out of control. This is the so-called "Generalist-Specialist Gap". We highlight this gap as a strategic avenue for achieving extremely elastic temporal context windows in future time series forecasting models.

Table 6: Performance comparison between MIOL and Arbitrary Length. IS (S1) trains one model per length as the baseline. MIOL (S3) is trained only on $L, T \in \{48, 96\}$. Arbitrary Length trains on continuous lengths $L, T \in [48, 96]$.

Dataset	L-T	IS (S1)		MIOL (S3)		Arbitrary Length	
		MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	48-48	0.338	0.362	0.339	0.363	0.381	0.408
	48-96	0.376	0.389	0.381	0.390	0.431	0.439
	96-48	0.328	0.361	0.331	0.361	0.372	0.390
	96-96	0.369	0.385	0.371	0.387	0.421	0.418
	AVG	0.353	0.374	0.356	0.375	0.401	0.414
ETTm1	48-48	0.392	0.383	0.412	0.388	0.580	0.481
	48-96	0.382	0.380	0.379	0.380	0.519	0.469
	96-48	0.269	0.314	0.273	0.317	0.324	0.357
	96-96	0.303	0.337	0.306	0.341	0.356	0.379
	AVG	0.337	0.354	0.343	0.357	0.445	0.422

F Random Seed Robustness Test

Unless otherwise specified, all experiments in this paper use the same fixed random seed. To assess the robustness of ORANGETREE to random seed selection, we additionally evaluate ORANGETREE-IS using five different random seeds on ETTh1, ETTm1, and Weather. Table 7 reports the standard deviation and average MSE/MAE across seeds, together with the results of SOTA baselines. ETTh1, a relatively small dataset, exhibits larger standard deviations than ETTm1 and Weather. Nevertheless, its average MSE and MAE across the five seeds remain lower than those of the SOTA baselines.

Table 7: Forecasting performance across five random seeds ($L = 96$, averaged over output lengths $T \in \{96, 192, 336, 720\}$). **Bold** and underlined values indicate the best and second-best results when comparing ORANGETREE-IS AVG with the baselines.

	ORANGETREE-IS STD		ORANGETREE-IS AVG		DGraFormer (2025)		CASA (2025)		TQNet (2025)		S-Mamba (2025)	
Dataset	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	0.008	0.007	**0.432**	**0.428**	0.442	0.432	0.438	0.433	0.441	0.434	0.455	0.450
ETTm1	0.001	0.001	**0.366**	**0.379**	0.379	0.394	0.386	0.394	0.377	0.393	0.398	0.405
Weather	0.001	0.001	**0.234**	**0.262**	0.249	0.271	0.243	0.267	0.242	0.269	0.251	0.276

G Full Experimental Results for Fixed-Input Forecasting

Table 8 summarizes the forecasting performance across six benchmark datasets. By comparing the horizon-specific ORANGETREE-IS against the horizon-unified ORANGETREE-HU and current SOTA baselines, we draw two critical conclusions.

Superiority of the Segment Tree Architecture. In the standard fixed-horizon setting (S1), ORANGETREE-IS establishes a new SOTA, achieving the lowest average MSE on 5 out of 6 datasets (the second-best in the remaining dataset). Notably, on ETTh1 and ETTm1, ORANGETREE-IS outperforms complex Transformer-based models like DGraFormer and TQNet. This validates that our Segment Tree Encoder captures multi-scale temporal dependencies more effectively than fixed-window attention mechanisms, while the Pascal-based Range Weighter acts as a precise filter to suppress noise in volatile series.

Effectiveness of Parameter Decoupling (S1 vs. S2). A key contribution of this work is the ability to decouple model parameters from output length T. The ORANGETREE-HU variant, which optimizes a single set of parameters for all four horizons simultaneously (S2), demonstrates remarkable resilience. As shown in Table 8, the performance gap between the specialized ORANGETREE-IS and the unified ORANGETREE-HU is negligible — often less than 0.005 in MSE. In specific cases like ETTm1, ORANGETREE-HU even marginally outperforms the specialized models. This confirms that the Inverse Tree Decoder successfully generalizes across planning horizons without the "catastrophic forgetting" or parameter dilution typically observed when traditional models are forced to handle variable outputs with heuristic adapters.

Table 8: Multivariate forecasting performance ($L = 96$, $T \in \{96, 192, 336, 720\}$). We compare ORANGETREE-IS and all baselines (trained specifically for each horizon, Setting S1) against ORANGETREE-HU (a single unified model trained for all horizons, Setting S2). **Bold** and underlined values indicate the best and second-best results, respectively.

		ORANGETREE -IS (S1)		ORANGETREE -HU (S2)		DGraF. (2025)		CASA (2025)		TQNet (2025)		S-Mamba (2025)		SOFTS (2024)		iTrans. (2024)		PatchTST (2023)		TSMixer (2023)		TiDE (2023)		DLinear (2023)		LSTM (1997)	
Dataset	T	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	96	0.303	0.337	**0.300**	**0.336**	0.326	0.366	0.313	0.358	0.311	0.353	0.333	0.368	0.325	0.361	0.334	0.368	0.329	0.365	0.323	0.363	0.364	0.387	0.345	0.372	1.010	0.764
	192	0.355	0.366	**0.350**	**0.365**	0.358	0.376	0.375	0.382	0.356	0.378	0.376	0.390	0.377	0.391	0.380	0.394	0.376	0.392	0.450	0.451	0.374	0.387	0.380	0.389	1.221	0.851
	336	0.381	0.388	**0.377**	**0.387**	0.387	0.397	0.397	0.398	0.390	0.401	0.408	0.413	0.426	0.420	0.400	0.410	0.407	0.413	0.532	0.515	0.410	0.411	0.413	0.413	1.182	0.849
	720	**0.422**	0.424	0.436	**0.421**	0.446	0.436	0.459	0.437	0.452	0.440	0.475	0.448	0.466	0.447	0.491	0.459	0.475	0.453	0.485	0.459	0.487	0.461	0.474	0.453	1.312	0.835
	Avg	**0.365**	0.379	0.366	**0.377**	0.379	0.394	0.386	0.394	0.377	0.393	0.398	0.405	0.399	0.405	0.401	0.408	0.397	0.406	0.448	0.447	0.409	0.411	0.403	0.407	1.181	0.825
ETTm2	96	**0.168**	**0.249**	0.173	0.254	0.173	0.253	0.173	0.252	0.173	0.256	0.179	0.263	0.180	0.261	0.180	0.264	0.184	0.264	0.182	0.266	0.207	0.305	0.193	0.292	1.699	0.996
	192	**0.234**	**0.295**	0.237	0.296	0.243	0.302	0.240	0.298	0.238	0.298	0.250	0.309	0.246	0.306	0.250	0.309	0.246	0.306	0.249	0.309	0.290	0.364	0.284	0.362	2.492	1.210
	336	**0.292**	**0.331**	0.296	0.334	0.315	0.348	0.296	0.334	0.301	0.340	0.312	0.349	0.319	0.352	0.311	0.348	0.308	0.346	0.309	0.347	0.377	0.422	0.369	0.427	2.477	1.205
	720	**0.392**	**0.389**	0.395	0.392	0.404	0.401	0.395	0.392	0.397	0.396	0.411	0.406	0.405	0.401	0.412	0.407	0.409	0.402	0.416	0.408	0.558	0.524	0.554	0.522	3.581	1.635
	Avg	**0.272**	**0.316**	0.275	0.319	0.284	0.326	0.276	0.319	0.277	0.323	0.288	0.332	0.287	0.330	0.288	0.332	0.287	0.330	0.289	0.333	0.358	0.404	0.350	0.401	2.562	1.262
ETTh1	96	0.369	**0.385**	0.372	0.391	**0.367**	0.393	0.376	0.397	0.371	0.393	0.386	0.405	0.381	0.399	0.386	0.405	0.394	0.406	0.401	0.412	0.479	0.464	0.386	0.400	1.150	0.831
	192	**0.419**	**0.418**	0.422	0.420	0.438	0.426	0.427	0.424	0.428	0.426	0.443	0.437	0.435	0.431	0.441	0.436	0.440	0.435	0.452	0.442	0.525	0.492	0.437	0.432	1.263	0.878
	336	0.465	**0.440**	**0.462**	**0.440**	0.479	0.442	0.469	0.445	0.476	0.446	0.489	0.468	0.480	0.452	0.487	0.458	0.491	0.462	0.492	0.463	0.565	0.515	0.481	0.459	1.237	0.822
	720	**0.458**	**0.456**	**0.458**	**0.456**	0.484	0.467	0.479	0.468	0.487	0.470	0.502	0.489	0.499	0.488	0.503	0.491	0.487	0.479	0.507	0.490	0.594	0.558	0.519	0.516	1.200	0.841
	Avg	**0.428**	**0.425**	**0.428**	0.427	0.442	0.432	0.438	0.433	0.441	0.434	0.455	0.450	0.449	0.443	0.454	0.448	0.453	0.446	0.463	0.452	0.541	0.507	0.456	0.452	1.212	0.843
ETTh2	96	**0.283**	**0.335**	0.289	0.337	0.300	0.344	0.290	0.339	0.295	0.343	0.296	0.348	0.297	0.347	0.297	0.349	0.288	0.340	0.319	0.361	0.400	0.440	0.333	0.387	3.363	1.419
	192	**0.366**	**0.386**	0.372	0.388	0.382	0.397	**0.366**	0.388	0.367	0.393	0.376	0.396	0.373	0.394	0.380	0.400	0.376	0.395	0.402	0.410	0.528	0.509	0.477	0.476	3.438	1.417
	336	0.407	0.418	0.409	0.422	**0.388**	**0.415**	0.416	0.425	0.417	0.427	0.424	0.431	0.410	0.426	0.428	0.432	0.440	0.451	0.444	0.446	0.643	0.571	0.594	0.541	2.971	1.339
	720	**0.410**	0.435	0.417	0.437	0.418	0.436	0.420	0.439	0.433	0.446	0.426	0.444	0.411	**0.433**	0.427	0.445	0.436	0.453	0.441	0.450	0.874	0.679	0.831	0.657	3.359	1.549
	Avg	**0.366**	**0.394**	0.372	0.396	0.372	0.398	0.373	0.398	0.378	0.402	0.381	0.405	0.373	0.400	0.383	0.406	0.385	0.410	0.402	0.417	0.611	0.550	0.559	0.515	3.283	1.431
Weather	96	**0.149**	**0.187**	0.151	0.193	0.168	0.207	0.155	0.197	0.157	0.200	0.165	0.210	0.166	0.208	0.174	0.214	0.176	0.217	0.166	0.210	0.202	0.261	0.196	0.255	0.263	0.296
	192	**0.200**	**0.240**	0.204	0.244	0.213	0.249	0.206	0.245	0.206	0.245	0.214	0.252	0.217	0.253	0.221	0.254	0.221	0.256	0.215	0.256	0.242	0.298	0.237	0.296	0.348	0.352
	336	**0.258**	**0.285**	0.261	0.287	0.270	0.291	0.265	0.286	0.262	0.287	0.274	0.297	0.282	0.300	0.278	0.296	0.275	0.296	0.287	0.300	0.287	0.335	0.283	0.335	0.425	0.415
	720	**0.329**	**0.333**	0.339	0.337	0.345	0.338	0.347	0.340	0.344	0.342	0.350	0.345	0.356	0.351	0.358	0.347	0.352	0.346	0.355	0.348	0.351	0.386	0.345	0.381	0.587	0.541
	Avg	**0.234**	**0.261**	0.239	0.265	0.249	0.271	0.243	0.267	0.242	0.269	0.251	0.276	0.255	0.278	0.258	0.278	0.256	0.279	0.256	0.278	0.270	0.320	0.265	0.317	0.406	0.401
Solar	96	**0.162**	**0.214**	0.175	0.223	0.184	0.219	0.187	0.218	0.173	0.233	0.205	0.244	0.200	0.230	0.203	0.237	0.205	0.246	0.221	0.275	0.312	0.399	0.290	0.378	0.314	0.266
	192	**0.196**	0.235	0.200	0.240	0.211	**0.234**	0.223	0.243	0.199	0.257	0.237	0.270	0.229	0.253	0.233	0.261	0.237	0.267	0.268	0.306	0.339	0.416	0.320	0.398	0.235	0.245
	336	0.231	0.256	0.218	**0.247**	0.234	0.250	0.237	0.257	**0.211**	0.263	0.258	0.288	0.243	0.269	0.248	0.273	0.250	0.276	0.272	0.294	0.368	0.430	0.353	0.415	0.670	0.615
	720	0.227	**0.252**	0.235	0.257	0.235	0.254	0.239	0.258	**0.209**	0.270	0.260	0.288	0.245	0.272	0.249	0.275	0.252	0.275	0.281	0.313	0.370	0.425	0.356	0.413	0.773	0.620
	Avg	0.204	**0.239**	0.207	0.242	0.216	**0.239**	0.222	0.244	**0.198**	0.256	0.240	0.272	0.229	0.256	0.233	0.262	0.236	0.266	0.261	0.297	0.347	0.417	0.330	0.401	0.498	0.436

H Full MIOL (Multiple Input and Output Lengths) Scenario

In this section, we validate the model's adaptability in dynamic deployment environments where both lookback windows L and forecast horizons T vary during inference ($L, T \in \{48, 96\}$).

H.1 Baselines and Heuristic Adapters

To rigorously evaluate ORANGETREE, we compare it against two baseline categories: the idealized specialist and standard heuristic adaptations. We use ORANGETREE-MIOL to refer to our native unified model that support multiple input and output lengths.

1. Idealized Specialist (ORANGETREE-IS). This represents a theoretical performance upper bound, where a separate, dedicated a model is trained for every distinct (L, T) configuration.

2. Heuristic Adapters. For fixed-size models lacking native flexibility, we employ heuristic strategies to force variable-length support (illustrated in Figure 15).

- **Input Adaptation**: *Padding* aligns all sequences to the maximum length $L_{\max}$ by appending zeros (Figure 15(a)), while *Trimming* truncates sequences to the minimum length $L_{\min}$ (Figure 15(b)).

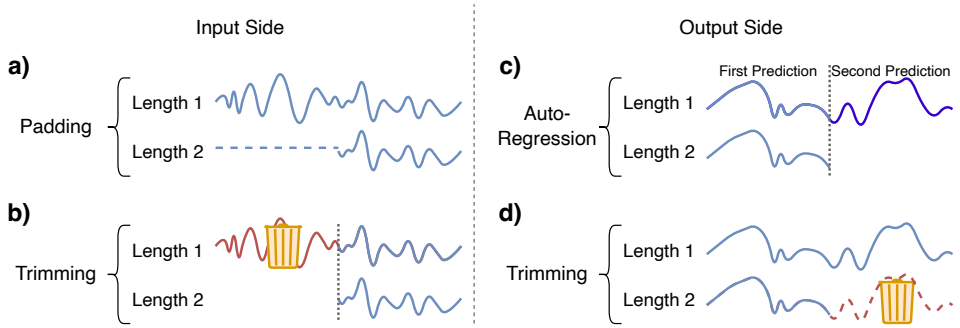


Figure 15: Visual illustration of heuristic adaptation strategies. (a) Input Padding; (b) Input Trimming; (c) Output Autoregression; (d) Output Trimming.

- **Output Adaptation**: *Autoregression* generates the shortest horizon $T_{\min}$ first and iteratively expands to $T_{\max}$ (Figure 15(c)). *Trimming* predicts the maximum horizon $T_{\max}$ and discards excess steps to match T_{target} (Figure 15(d)).

These strategies are combined to form four baselines: TRIM-REG, TRIM-TRIM, PAD-REG, and PAD-TRIM.

H.2 Theoretical Comparison

As summarized in Table 9, these heuristics introduce inherent trade-offs. Input padding and output trimming generate computational and memory redundancy ("Computational waste"), while input trimming discards potentially valuable historical context ("Information loss"). Output autoregression, while memory-efficient, suffers from error accumulation and high latency. In contrast, ORANGETREE provides native support without these structural penalties.

Table 9: Theoretical comparison of MIOL adaptation strategies versus Native Support.

Method	Input Side Trade-off	Output Side Trade-off
Padding	Computational/Memory Redundancy	–
Trimming	Information Loss	Computational/Memory Redundancy
Autoregression	–	Error Accumulation & High Latency
ORANGETREE (Ours)	**Native Support (Zero Redundancy)**	**Native Support (Zero Redundancy)**

H.3 Empirical Results and Analysis

Table 10 presents the granular performance results (Accuracy, Latency, Memory) across four input-output length combinations.

Native Elasticity Matches Specialized Training. A critical finding is that ORANGETREE-MIOL achieves forecasting accuracy roughly identical to the brute-force ORANGETREE-IS baseline. For example, on ETTh1, the average MSE gap between the native MIOL model (0.356) and the specialized IS ensemble (0.353) is negligible. This demonstrates that ORANGETREE successfully learns a length-invariant representation, allowing a single model to replace multiple specialized models without substantial performance degradation.

Superior Efficiency over Padding Heuristics. Adapters that rely on padding (e.g., PAD-REG) incur severe computational penalties. As shown in the ETTh1 results, the PAD-REG strategy causes inference latency to spike to 0.633s, nearly **2.5× slower** than ORANGETREE-MIOL (0.244s). This confirms that our native variable-length support eliminates the computational redundancy of processing padded zeros, making it significantly more suitable for high-throughput deployment.

Robustness against Trimming Artifacts. Adapters that rely on trimming suffer from catastrophic information loss, particularly in volatile datasets. On the Solar dataset, the TRIM-REG heuristic results in an MSE of 0.602—a massive degradation compared to ORANGETREE-MIOL's 0.240. This highlights the novelty of our Range Weighter and Feature Fuser: unlike simple trimming, our mechanism intelligently selects the most informative historical ranges, ensuring stability even when input contexts vary dynamically.

Table 10: Detailed performance trade-offs in the Full MIOL Scenario across four input-output length combinations ($L, T \in \{48, 96\}$). We compare the native ORANGETREE-MIOL against the ORANGETREE-IS (Idealized Specialist, trained specifically for each (L, T) pair, as the baseline) and four heuristic inference strategies. *Time* and *Mem* denote inference latency (seconds) and memory usage (GB), measured with batch sizes of 2048 (ETT), 1024 (Weather), and 192 (Solar).

		ORANGETREE -IS (S1)				ORANGETREE -MIOL (S3)				ORANGETREE +TRIM-REG				ORANGETREE +TRIM-TRIM				ORANGETREE +PAD-REG				ORANGETREE +PAD-TRIM			
Dataset	*L/T*	MSE	MAE	Time	MEM	MSE	MAE	Time	MEM	MSE	MAE	Time	MEM	MSE	MAE	Time	MEM	MSE	MAE	Time	MEM	MSE	MAE	Time	MEM
ETTh1	48-48	0.338	0.362	0.109	6.43	**0.339**	**0.363**	0.106	6.47	0.347	0.367	0.110	6.48	0.341	0.370	0.124	12.74	0.367	0.394	0.380	6.68	0.351	0.386	0.475	12.90
	48-96	0.376	0.389	0.111	12.63	**0.381**	**0.390**	0.111	12.68	0.393	0.398	0.312	6.48	0.383	0.396	0.119	12.74	0.399	0.413	0.872	6.70	0.391	0.412	0.455	12.91
	96-48	0.328	0.361	0.379	6.89	0.331	0.361	0.376	6.75	0.347	0.367	0.111	6.48	0.341	0.370	0.119	12.74	0.338	0.362	0.395	6.68	**0.326**	**0.360**	0.468	12.90
	96-96	0.369	0.385	0.392	12.88	0.371	0.387	0.383	12.95	0.393	0.398	0.305	6.49	0.383	0.396	0.120	12.75	0.375	0.388	0.884	6.70	**0.369**	**0.386**	0.460	12.91
	AVG	0.353	0.374	0.248	9.71	**0.356**	**0.375**	0.244	9.71	0.370	0.383	0.210	6.48	0.362	0.383	0.121	12.74	0.370	0.389	0.633	6.69	0.359	0.386	0.465	12.91
ETTh2	48-48	0.230	0.294	0.114	6.47	0.230	0.296	0.110	6.44	0.234	0.300	0.108	6.44	0.231	**0.294**	0.111	12.65	**0.225**	0.310	0.395	6.90	0.231	0.304	0.408	12.95
	48-96	0.291	0.336	0.108	12.67	0.298	0.340	0.111	12.64	0.301	0.344	0.311	6.45	0.299	**0.338**	0.113	12.65	**0.288**	0.354	0.864	6.91	0.302	0.353	0.409	12.96
	96-48	0.220	0.291	0.375	6.68	**0.218**	0.290	0.395	6.68	0.234	0.300	0.108	6.44	0.231	0.294	0.113	12.65	0.234	0.305	0.382	6.89	0.220	**0.288**	0.420	12.95
	96-96	0.283	0.335	0.447	12.91	0.285	0.334	0.411	12.88	0.301	0.344	0.329	6.45	0.299	0.338	0.113	12.65	0.304	0.354	0.870	6.91	**0.283**	**0.330**	0.417	12.95
	AVG	0.256	0.314	0.261	9.68	**0.258**	**0.315**	0.257	9.66	0.268	0.322	0.214	6.45	0.265	0.316	0.113	12.65	0.263	0.331	0.628	6.90	0.259	0.319	0.414	12.95
ETTm1	48-48	0.392	0.383	0.105	6.47	**0.412**	**0.388**	0.131	6.45	0.456	0.411	0.125	6.48	0.419	0.393	0.109	12.64	0.530	0.455	0.457	6.72	0.462	0.419	0.405	12.91
	48-96	0.382	0.380	0.120	12.65	**0.379**	**0.380**	0.131	12.65	0.436	0.413	0.338	6.49	0.387	0.385	0.109	12.64	0.459	0.423	0.999	6.73	0.428	0.411	0.399	12.92
	96-48	0.269	0.314	0.409	6.68	**0.273**	**0.317**	0.418	6.69	0.456	0.411	0.122	6.48	0.419	0.393	0.109	12.64	0.304	0.345	0.447	6.71	0.278	0.322	0.400	12.91
	96-96	0.303	0.337	0.415	12.88	**0.306**	**0.341**	0.401	12.89	0.436	0.413	0.337	6.49	0.387	0.385	0.111	12.65	0.336	0.364	0.977	6.73	0.310	0.344	0.408	12.91
	AVG	0.337	0.354	0.262	9.67	**0.343**	**0.357**	0.270	9.67	0.446	0.412	0.231	6.49	0.403	0.389	0.110	12.64	0.407	0.397	0.720	6.72	0.370	0.374	0.403	12.91
ETTm2	48-48	0.137	0.226	0.112	6.54	**0.139**	**0.228**	0.116	6.48	0.148	0.247	0.128	6.46	0.149	0.243	0.120	12.67	0.169	0.277	0.428	6.76	0.158	0.258	0.380	12.91
	48-96	0.179	0.255	0.115	12.67	**0.179**	**0.255**	0.120	12.68	0.186	0.270	0.319	6.47	0.188	0.268	0.120	12.68	0.192	0.281	0.955	6.78	0.204	0.296	0.396	12.92
	96-48	0.128	0.219	0.380	6.89	**0.129**	**0.216**	0.389	6.76	0.148	0.247	0.119	6.46	0.149	0.243	0.136	12.68	0.137	0.233	0.433	6.76	0.131	0.226	0.383	12.91
	96-96	0.168	0.249	0.395	12.95	**0.171**	**0.247**	0.420	12.96	0.186	0.270	0.319	6.47	0.188	0.268	0.124	12.68	0.177	0.258	0.964	6.77	0.174	0.255	0.398	12.91
	AVG	0.153	0.237	0.251	9.76	**0.155**	**0.237**	0.261	9.72	0.167	0.256	0.221	6.47	0.169	0.256	0.125	12.68	0.169	0.262	0.695	6.77	0.167	0.259	0.389	12.91
Weather	48-48	0.140	0.176	0.104	9.31	0.142	**0.176**	0.113	9.32	0.171	0.204	0.119	9.38	0.148	0.182	0.125	18.47	**0.140**	0.182	0.413	9.47	0.144	0.183	0.395	18.64
	48-96	0.172	0.214	0.108	18.46	0.178	**0.218**	0.114	18.48	0.214	0.254	0.366	9.39	0.188	0.229	0.112	18.48	**0.175**	0.222	0.933	9.49	0.180	0.223	0.391	18.65
	96-48	0.119	0.151	0.424	9.44	0.119	0.153	0.398	9.45	0.171	0.204	0.117	9.38	0.148	0.182	0.113	18.48	0.120	0.156	0.413	9.47	**0.114**	**0.146**	0.407	18.64
	96-96	0.149	0.187	0.395	18.59	0.150	0.190	0.383	18.61	0.214	0.254	0.366	9.39	0.188	0.229	0.112	18.48	0.152	0.195	0.928	9.49	**0.145**	**0.185**	0.406	18.64
	AVG	0.145	0.182	0.258	13.95	0.147	**0.184**	0.252	13.97	0.192	0.229	0.242	9.39	0.168	0.205	0.116	18.48	0.147	0.189	0.672	9.48	**0.146**	**0.184**	0.400	18.64
Solar	48-48	0.325	0.361	0.127	11.24	0.297	0.330	0.117	11.33	0.546	0.565	0.117	11.33	**0.283**	0.318	0.122	22.45	0.285	0.304	0.416	11.37	0.295	**0.303**	0.415	22.49
	48-96	0.344	0.372	0.112	22.36	0.350	0.373	0.123	22.45	0.658	0.607	0.395	11.35	**0.321**	0.355	0.122	22.45	0.342	**0.344**	0.954	11.40	0.361	0.358	0.401	22.50
	96-48	0.139	0.198	0.401	11.28	**0.144**	**0.210**	0.417	11.37	0.546	0.565	0.117	11.34	0.283	0.318	0.122	22.45	0.160	0.242	0.415	11.37	0.148	0.222	0.413	22.48
	96-96	0.162	0.214	0.419	22.42	**0.168**	**0.225**	0.414	22.49	0.658	0.607	0.398	11.36	0.321	0.355	0.122	22.46	0.188	0.274	0.954	11.40	0.172	0.239	0.417	22.49
	AVG	0.243	0.286	0.265	16.83	**0.240**	0.285	0.268	16.91	0.602	0.586	0.257	11.34	0.302	0.336	0.122	22.45	0.244	0.291	0.685	11.39	0.244	**0.281**	0.412	22.49

To fully demonstrate ORANGETREE's robust MIOL capability under challenging length ranges, we expanded the combinations from 4 to 8 pairs including standard long-term horizons:

$$L \in \{48, 96, 144\}, \quad T \in \begin{cases} \{96, 192\}, & L = 48 \\ \{96, 192, 336, 720\}, & L = 96 \\ \{336, 720\}, & L = 144 \end{cases}. \tag{18}$$

As shown in Table 11, despite the immensely increased task difficulty, ORANGETREE consistently preserves peak performance and occasionally outperforms the ORANGETREE-ISs which are independently trained models.

Table 11: Multivariate forecasting performance comparing ORANGETREE-IS (S1) and ORANGETREE-MIOL (S3) across 8 different (L, T) pairs. **Bold** indicates the better result.

			ORANGETREE -IS (S1)		ORANGETREE -MIOL (S3)					ORANGETREE -IS (S1)		ORANGETREE -MIOL (S3)	
Dataset	L	T	MSE	MAE	MSE	MAE	Dataset	L	T	MSE	MAE	MSE	MAE
	48	96	0.376	**0.389**	**0.375**	0.392		48	96	**0.179**	**0.255**	0.180	0.262
	48	192	0.437	0.425	**0.425**	**0.422**		48	192	0.247	**0.304**	**0.246**	0.305
	96	96	0.369	**0.385**	**0.364**	0.389		96	96	**0.168**	**0.249**	0.171	0.255
	96	192	0.419	0.418	**0.409**	**0.417**		96	192	**0.234**	**0.295**	**0.234**	0.296
ETTh1	96	336	0.465	0.440	**0.438**	**0.436**	ETTm2	96	336	**0.292**	**0.331**	0.295	0.335
	96	720	0.458	0.456	**0.440**	**0.454**		96	720	**0.392**	**0.389**	0.398	0.395
	144	336	0.458	0.435	**0.432**	**0.433**		144	336	**0.283**	**0.327**	0.287	0.331
	144	720	0.456	0.456	**0.439**	**0.453**		144	720	**0.376**	**0.383**	0.388	0.390
	Avg		0.430	0.426	**0.415**	**0.425**		Avg		**0.271**	**0.317**	0.275	0.321
	48	96	**0.291**	**0.336**	0.303	0.346		48	96	**0.172**	**0.214**	0.180	0.222
	48	192	**0.381**	**0.389**	0.393	0.398		48	192	**0.220**	**0.256**	0.225	0.261
	96	96	**0.283**	**0.335**	0.290	0.339		96	96	**0.149**	**0.187**	0.152	0.194
	96	192	**0.366**	**0.386**	0.375	0.390		96	192	**0.200**	**0.240**	0.201	0.241
ETTh2	96	336	**0.407**	**0.418**	0.419	0.425	Weather	96	336	**0.258**	0.285	**0.258**	**0.284**
	96	720	**0.410**	**0.435**	0.424	0.440		96	720	**0.329**	**0.333**	0.335	0.335
	144	336	**0.400**	**0.417**	0.405	0.418		144	336	0.257	0.285	**0.251**	**0.279**
	144	720	**0.409**	**0.434**	0.420	0.437		144	720	0.328	0.333	**0.326**	**0.330**
	Avg		**0.368**	**0.394**	0.379	0.399		Avg		**0.239**	**0.267**	0.241	0.268
	48	96	0.382	0.380	**0.380**	**0.376**		48	96	0.344	0.372	**0.318**	**0.352**
	48	192	**0.422**	**0.404**	0.431	**0.404**		48	192	0.314	0.337	**0.303**	**0.329**
	96	96	0.303	**0.337**	**0.299**	0.338		96	96	**0.162**	**0.214**	0.177	0.221
	96	192	0.355	0.366	**0.343**	**0.364**		96	192	**0.196**	**0.235**	0.201	0.238
ETTm1	96	336	0.381	**0.388**	**0.376**	**0.388**	Solar	96	336	0.231	0.256	**0.223**	**0.253**
	96	720	**0.422**	0.424	0.433	**0.421**		96	720	**0.227**	**0.252**	0.241	0.262
	144	336	0.380	0.387	**0.369**	**0.384**		144	336	0.225	**0.241**	**0.220**	0.249
	144	720	**0.421**	0.422	0.422	**0.416**		144	720	**0.224**	**0.250**	0.236	0.260
	Avg		0.383	0.389	**0.382**	**0.386**		Avg		**0.240**	**0.270**	**0.240**	0.271

I Full MIOL Scenario for Other TSF Models

I.1 Benchmarking Against Adapted SOTA Models

To verify that the limitations of heuristic adapters are not specific to ORANGETREE, we extend our evaluation to recent state-of-the-art architectures, specifically CASA [26] and DGraFormer [24]. Since these models lack native support for variable input/output lengths, we apply the four heuristic strategies defined in Appendix H (e.g., TRIM-REG, PAD-TRIM) and quantify their performance stability.

For each TSF model, we first establish an "IS" (idealized specialist) baseline by training specialized models for every (L, T) pair to determine optimal theoretical performance. We then evaluate the performance using either native support (ORANGETREE-MIOL for ORANGETREE) or heuristic adapters (for SOTA). The key metric is the **relative degradation rate**: the percentage increase in MSE/MAE compared to the specialized -IS baseline, indicating the "flexibility tax" incurred by the model.

Table 12 reveals a sharp contrast in architectural resilience:

- **Native Stability vs. Heuristic Fragility**: ORANGETREE-MIOL demonstrates exceptional stability, typically exhibiting only a minor performance drop compared to its fixed baseline (with an average MSE increase of merely 0.75%). In contrast, SOTA models suffer severe degradation under heuristic adaptation. Notably, in some cases, heuristic methods even triggered an MSE surge exceeding +200%, rendering the models practically unusable for MIOL forecasting.
- **Incompatibility of Rigid Architectures**: While models like DGraFormer and CASA perform well in fixed settings, they lack the structural elasticity to handle multiple lengths. Imposing variable lengths on rigid attention mechanisms that rely on precise positional encoding cripples their temporal modeling capability.
- **Architectural Advantage**: The results conclusively show that ORANGETREE's performance stems from its design. By decoupling model parameters from sequence length, ORANGETREE avoids the "structural mismatch" that plagues other models when processed via padding or autoregression. This positions ORANGETREE as a robust solution for applications requiring flexible support for multiple input-output lengths (MIOL) configurations.

Table 12: Comparative analysis of ORANGETREE-MIOL (native support) versus SOTA models (CASA, DGraFormer) adapted via heuristics. We evaluate performance across randomized input-output configurations ($L, T \in \{48, 96\}$). The IS column represents the "idealized specialist" baseline (one model trained per (L, T) pair). For other columns, values in red indicate performance degradation (MSE/MAE increase) relative to the IS baseline, while blue indicates improvement.

		OrangeTree				DGraFormer										CASA									
		IS (S1)		MIOL (S3)		IS		Trim-Reg		Trim-Trim		Pad-Reg		Pad-Trim		IS		Trim-Reg		Trim-Trim		Pad-Reg		Pad-Trim	
Dataset	L/T	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	48-48	0.338	0.362	0.339	0.363	0.340	0.370	0.339	0.368	0.334	0.368	0.362	0.394	0.357	0.392	0.342	0.377	0.344	0.376	0.340	0.377	0.360	0.397	0.354	0.387
		-	-	+0.30%	+0.28%	-	-	-0.32%	-0.44%	-1.83%	-0.45%	+6.42%	+6.60%	+5.04%	+6.07%	-	-	+0.66%	-0.02%	-0.41%	+0.04%	+5.48%	+5.51%	+3.57%	+2.80%
	48-96	0.376	0.389	0.381	0.390	0.371	0.395	0.376	0.396	0.372	0.393	0.392	0.415	0.399	0.423	0.383	0.404	0.385	0.406	0.380	0.401	0.396	0.420	0.393	0.413
		-	-	+1.33%	+0.26%	-	-	+1.18%	+0.43%	+0.08%	-0.36%	+5.72%	+5.09%	+7.43%	+7.11%	-	-	+0.39%	+0.54%	-0.90%	-0.68%	+3.34%	+3.91%	+2.68%	+2.20%
	96-48	0.328	0.361	0.331	0.361	0.334	0.371	0.339	0.368	0.334	0.368	0.332	0.368	0.329	0.368	0.340	0.381	0.344	0.376	0.340	0.377	0.338	0.380	0.334	0.377
		-	-	+0.91%	0.00%	-	-	+1.44%	-0.80%	-0.10%	-0.81%	-0.62%	-0.69%	-1.50%	-0.96%	-	-	+1.08%	-1.11%	+0.01%	-1.05%	-0.57%	-0.24%	-1.92%	-0.94%
	96-96	0.369	0.385	0.371	0.387	0.367	0.393	0.376	0.396	0.372	0.393	0.366	0.392	0.367	0.391	0.376	0.397	0.385	0.406	0.380	0.401	0.374	0.403	0.370	0.398
		-	-	+0.54%	+0.52%	-	-	+2.35%	+0.83%	+1.23%	+0.04%	-0.30%	-0.17%	+0.01%	-0.43%	-	-	+2.31%	+2.25%	+1.00%	+1.00%	-0.42%	+1.47%	-1.69%	+0.34%
	AVG	0.353	0.374	0.356	0.375	0.353	0.382	0.357	0.382	0.353	0.381	0.363	0.392	0.363	0.393	0.360	0.389	0.364	0.391	0.360	0.389	0.367	0.400	0.363	0.394
		-	-	+0.78%	+0.27%	-	-	+1.18%	+0.03%	-0.12%	-0.39%	+2.82%	+2.70%	+2.81%	+2.96%	-	-	+1.12%	+0.44%	-0.07%	-0.17%	+1.94%	+2.66%	+0.66%	+1.10%
ETTh2	48-48	0.230	0.294	0.230	0.296	0.234	0.297	0.234	0.298	0.236	0.298	0.248	0.318	0.246	0.316	0.243	0.306	0.244	0.307	0.245	0.307	0.324	0.378	0.291	0.360
		-	-	0.00%	+0.68%	-	-	-0.25%	+0.37%	+0.65%	+0.39%	+6.07%	+7.11%	+5.25%	+6.43%	-	-	+0.32%	+0.63%	+0.79%	+0.61%	+33.27%	+23.76%	+19.91%	+17.69%
	48-96	0.291	0.336	0.298	0.340	0.298	0.340	0.301	0.342	0.298	0.340	0.310	0.356	0.334	0.378	0.309	0.348	0.307	0.348	0.307	0.349	0.399	0.423	0.393	0.426
		-	-	+2.41%	+1.19%	-	-	+1.10%	+0.53%	+0.32%	+0.03%	+4.19%	+4.73%	+12.43%	+11.28%	-	-	-0.60%	0.00%	-0.40%	+0.03%	+29.19%	+21.28%	+27.26%	+22.32%
	96-48	0.220	0.291	0.218	0.290	0.223	0.294	0.234	0.298	0.236	0.298	0.225	0.295	0.224	0.296	0.238	0.309	0.244	0.307	0.245	0.307	0.237	0.308	0.238	0.309
		-	-	-0.91%	-0.34%	-	-	+4.85%	+1.17%	+5.79%	+1.18%	+1.10%	+0.06%	+0.67%	+0.46%	-	-	+2.48%	-0.50%	+2.96%	-0.53%	-0.25%	-0.41%	-0.10%	+0.11%
	96-96	0.283	0.335	0.285	0.334	0.300	0.344	0.301	0.342	0.298	0.340	0.288	0.337	0.285	0.337	0.290	0.339	0.307	0.348	0.307	0.349	0.299	0.348	0.300	0.349
		-	-	+0.71%	-0.30%	-	-	+0.26%	-0.62%	-0.51%	-1.12%	-4.04%	-2.11%	-5.00%	-2.16%	-	-	+5.77%	+2.79%	+5.98%	+2.83%	+3.09%	+2.73%	+3.30%	+2.82%
	AVG	0.256	0.314	0.258	0.315	0.264	0.319	0.267	0.320	0.267	0.319	0.268	0.326	0.273	0.332	0.270	0.326	0.275	0.328	0.276	0.328	0.315	0.364	0.305	0.361
		-	-	+0.68%	+0.32%	-	-	+1.35%	+0.33%	+1.31%	+0.07%	+1.61%	+2.36%	+3.39%	+4.03%	-	-	+2.00%	+0.76%	+2.32%	+0.76%	+16.61%	+11.88%	+13.14%	+10.88%
ETTm1	48-48	0.392	0.383	0.412	0.388	0.463	0.419	0.450	0.412	0.441	0.405	0.557	0.462	0.545	0.456	0.429	0.408	0.421	0.402	0.425	0.407	0.461	0.418	0.465	0.425
		-	-	+5.10%	+1.31%	-	-	-2.78%	-1.68%	-4.71%	-3.28%	+20.34%	+10.16%	+17.68%	+8.81%	-	-	-1.88%	-1.64%	-0.98%	-0.48%	+7.46%	+2.39%	+8.18%	+3.94%
	48-96	0.382	0.380	0.379	0.380	0.403	0.393	0.427	0.411	0.399	0.391	0.472	0.430	0.462	0.425	0.406	0.398	0.429	0.413	0.404	0.403	0.414	0.404	0.419	0.411
		-	-	-0.79%	0.00%	-	-	+5.97%	+4.61%	-0.91%	-0.47%	+17.24%	+9.43%	+14.77%	+7.95%	-	-	+5.73%	+3.90%	-0.45%	+1.35%	+2.09%	+1.59%	+3.19%	+3.36%
	96-48	0.269	0.314	0.273	0.317	0.292	0.340	0.450	0.412	0.441	0.405	0.315	0.358	0.292	0.342	0.287	0.332	0.421	0.402	0.425	0.407	0.287	0.338	0.288	0.338
		-	-	+1.49%	+0.96%	-	-	+54.30%	+21.35%	+51.24%	+19.38%	+8.04%	+5.54%	+0.20%	+0.63%	-	-	+46.69%	+20.86%	+48.05%	+22.29%	+0.06%	+1.65%	+0.11%	+1.81%
	96-96	0.303	0.337	0.306	0.341	0.326	0.366	0.427	0.411	0.399	0.391	0.349	0.377	0.325	0.362	0.313	0.358	0.429	0.413	0.404	0.403	0.324	0.361	0.325	0.364
		-	-	+0.99%	+1.19%	-	-	+30.96%	+12.40%	+22.45%	+6.94%	+7.11%	+3.09%	-0.25%	-1.21%	-	-	+37.09%	+15.41%	+29.08%	+12.58%	+3.52%	+0.93%	+3.68%	+1.74%
	AVG	0.337	0.354	0.343	0.357	0.371	0.380	0.439	0.412	0.420	0.398	0.423	0.407	0.406	0.396	0.359	0.374	0.425	0.407	0.415	0.405	0.372	0.380	0.374	0.385
		-	-	+1.78%	+0.85%	-	-	+18.23%	+8.49%	+13.29%	+4.98%	+14.17%	+7.23%	+9.51%	+4.34%	-	-	+18.49%	+8.91%	+15.54%	+8.19%	+3.60%	+1.66%	+4.18%	+2.79%
ETTm2	48-48	0.137	0.226	0.139	0.228	0.154	0.245	0.155	0.246	0.155	0.247	0.175	0.280	0.172	0.272	0.148	0.238	0.152	0.243	0.149	0.241	0.242	0.333	0.209	0.311
		-	-	+1.46%	+0.88%	-	-	+0.70%	+0.35%	+0.63%	+0.65%	+13.77%	+14.19%	+11.68%	+10.97%	-	-	+2.67%	+2.24%	+0.50%	+1.51%	+63.78%	+40.27%	+41.36%	+30.89%
	48-96	0.179	0.255	0.179	0.255	0.186	0.268	0.192	0.273	0.185	0.267	0.199	0.292	0.276	0.342	0.185	0.267	0.191	0.272	0.182	0.264	0.340	0.388	0.343	0.387
		-	-	0.00%	0.00%	-	-	+2.82%	+1.89%	-0.48%	-0.44%	+7.10%	+9.03%	+48.37%	+27.65%	-	-	+3.36%	+1.87%	-1.31%	-1.06%	+83.98%	+45.38%	+85.69%	+45.10%
	96-48	0.128	0.219	0.129	0.216	0.132	0.222	0.155	0.246	0.155	0.247	0.139	0.233	0.136	0.229	0.134	0.227	0.152	0.243	0.149	0.241	0.137	0.231	0.137	0.231
		-	-	+0.78%	-1.37%	-	-	+17.75%	+10.80%	+17.66%	+11.14%	+5.71%	+4.63%	+3.50%	+2.96%	-	-	+13.36%	+7.08%	+10.97%	+6.32%	+2.20%	+1.80%	+2.49%	+1.93%
	96-96	0.168	0.249	0.171	0.247	0.173	0.253	0.192	0.273	0.185	0.267	0.178	0.259	0.174	0.256	0.173	0.252	0.191	0.272	0.182	0.264	0.177	0.259	0.176	0.259
		-	-	+1.79%	-0.80%	-	-	+10.71%	+7.88%	+7.15%	+5.42%	+2.93%	+2.28%	+0.86%	+1.00%	-	-	+10.48%	+7.80%	+5.49%	+4.71%	+2.11%	+2.65%	+1.97%	+2.62%
	AVG	0.153	0.237	0.155	0.237	0.161	0.247	0.173	0.260	0.170	0.257	0.173	0.266	0.190	0.275	0.160	0.246	0.171	0.257	0.166	0.253	0.224	0.303	0.217	0.297
		-	-	+0.98%	-0.32%	-	-	+7.48%	+5.04%	+5.54%	+3.94%	+7.29%	+7.59%	+17.70%	+11.14%	-	-	+7.22%	+4.68%	+3.52%	+2.74%	+40.05%	+23.14%	+35.38%	+20.82%
Weather	48-48	0.140	0.176	0.142	0.176	0.150	0.183	0.166	0.197	0.153	0.185	0.150	0.199	0.157	0.204	0.138	0.179	0.144	0.181	0.141	0.183	0.149	0.206	0.148	0.201
		-	-	+1.92%	+0.27%	-	-	+11.05%	+7.87%	+2.10%	+1.10%	-0.06%	+9.01%	+5.05%	+11.76%	-	-	+4.97%	+1.15%	+2.81%	+2.02%	+8.40%	+14.55%	+7.36%	+12.22%
	48-96	0.172	0.214	0.178	0.218	0.192	0.232	0.202	0.239	0.190	0.228	0.187	0.242	0.193	0.246	0.175	0.223	0.179	0.224	0.174	0.223	0.195	0.257	0.196	0.258
		-	-	+3.16%	+2.04%	-	-	+5.20%	+3.02%	-1.11%	-1.40%	-2.74%	+4.57%	+0.72%	+6.43%	-	-	+2.43%	+0.15%	-0.50%	-0.13%	+11.43%	+15.18%	+12.25%	+15.53%
	96-48	0.119	0.151	0.119	0.153	0.130	0.168	0.166	0.197	0.153	0.185	0.130	0.168	0.134	0.173	0.127	0.167	0.144	0.181	0.141	0.183	0.125	0.168	0.126	0.168
		-	-	-0.13%	+1.37%	-	-	+27.97%	+17.08%	+17.66%	+9.73%	+0.08%	+0.14%	+3.07%	+2.89%	-	-	+13.73%	+8.99%	+11.39%	+9.92%	-1.36%	+0.69%	-0.97%	+0.86%
	96-96	0.149	0.187	0.150	0.190	0.168	0.207	0.202	0.239	0.190	0.228	0.164	0.208	0.169	0.213	0.155	0.197	0.179	0.224	0.174	0.223	0.162	0.210	0.160	0.205
		-	-	+0.67%	+1.29%	-	-	+20.27%	+15.26%	+13.05%	+10.31%	-2.20%	+0.43%	+0.66%	+2.72%	-	-	+15.59%	+13.59%	+12.29%	+13.27%	+4.78%	+6.43%	+2.97%	+4.19%
	AVG	0.145	0.182	0.147	0.184	0.160	0.197	0.184	0.218	0.171	0.206	0.158	0.204	0.163	0.209	0.149	0.192	0.162	0.203	0.158	0.203	0.158	0.210	0.157	0.208
		-	-	+1.55%	+1.28%	-	-	+15.15%	+10.35%	+7.17%	+4.62%	-1.40%	+3.57%	+2.20%	+5.93%	-	-	+8.86%	+5.76%	+6.14%	+6.00%	+6.26%	+9.63%	+5.87%	+8.65%
Solar	48-48	0.325	0.361	0.297	0.330	0.611	0.421	0.570	0.417	0.363	0.359	0.342	0.384	0.331	0.383	0.246	0.270	0.249	0.281	0.262	0.276	0.237	0.267	0.245	0.259
		-	-	-8.90%	-8.81%	-	-	-6.82%	-0.94%	-40.61%	-14.82%	-44.06%	-8.73%	-45.88%	-9.00%	-	-	+1.49%	+4.24%	+6.58%	+2.55%	-3.48%	-1.06%	-0.33%	-3.73%
	48-96	0.344	0.372	0.350	0.373	0.597	0.424	0.600	0.445	0.401	0.388	0.417	0.432	0.376	0.439	0.318	0.314	0.288	0.309	0.318	0.308	0.279	0.295	0.279	0.289
		-	-	+1.78%	+0.40%	-	-	+0.64%	+5.03%	-32.81%	-8.60%	-30.07%	+1.77%	-37.03%	+3.49%	-	-	-9.46%	-1.74%	+0.02%	-1.93%	-12.24%	-6.21%	-12.31%	-8.16%
	96-48	0.139	0.198	0.144	0.210	0.203	0.245	0.570	0.417	0.363	0.359	0.277	0.326	0.302	0.355	0.179	0.197	0.249	0.281	0.262	0.276	0.176	0.207	0.186	0.198
		-	-	+3.84%	+5.97%	-	-	+180.45%	+70.05%	+78.75%	+46.22%	+36.47%	+33.07%	+48.85%	+44.93%	-	-	+38.93%	+42.34%	+45.91%	+40.02%	-1.75%	+4.79%	+3.63%	+0.56%
	96-96	0.163	0.214	0.168	0.225	0.184	0.219	0.600	0.445	0.401	0.388	0.293	0.341	0.291	0.339	0.187	0.218	0.288	0.309	0.318	0.308	0.209	0.232	0.226	0.226
		-	-	+3.30%	+5.29%	-	-	+226.34%	+103.37%	+117.88%	+77.00%	+59.25%	+55.82%	+58.21%	+54.89%	-	-	+54.13%	+41.67%	+70.27%	+41.40%	+11.97%	+6.41%	+20.90%	+3.60%
	AVG	0.243	0.286	0.240	0.285	0.399	0.327	0.585	0.431	0.382	0.373	0.332	0.371	0.325	0.379	0.233	0.250	0.269	0.295	0.290	0.292	0.226	0.250	0.234	0.243
		-	-	-1.25%	-0.63%	-	-	+46.71%	+31.74%	-4.21%	+13.99%	-16.66%	+13.30%	-18.50%	+15.83%	-	-	+15.54%	+18.05%	+24.72%	+17.02%	-3.04%	+0.11%	+0.60%	-2.68%

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction explicitly claim that our work proposes OrangeTree, the first end-to-end TSF architecture that natively supports Multiple Input and Output Lengths (MIOL) without parameter inflation, computational redundancy, or severe performance degradation.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We provide a "Limitation and Future Work" section in the appendix E.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: We provide detailed formulations for each process, along with comprehensive comparative and ablation experiments to validate their effectiveness.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide the full source code and the software/hardware specifications used in our experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide the complete source code along with detailed documentation. All datasets used in this work are publicly available. With these materials, the main experimental results can be faithfully reproduced.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: These details are fully described in Section 4.1 (Experimental Setup) and Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We evaluate ORANGETREE-IS using five different random seeds on ETTh1, ETTm1, and Weather, and report the mean and standard deviation of MSE/MAE.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The full hardware and software configurations (type of compute workers, memory, and software environment) are detailed in Appendix B.4.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: An ethics statement is included in the paper.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: An impact statement is included in the paper.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: As stated in our ethics statement, this work does not present any negative societal impacts or potential risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have properly cited all the models and open-source datasets mentioned in the paper.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release the complete source code for OrangeTree as a new asset, with detailed software/hardware environments described in the paper and a requirements file provided in the anonymous code repository.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer:[N/A]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigor, or originality of the research.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.