Entropy Guided Dynamic Patch Segmentation for Time Series Transformers

Sachith Abeywickrama[1,2] **Emadeldeen Eldele**[3*] **Min Wu**[2] **Xiaoli Li**[4] **Chau Yuen**[1]

[1]Nanyang Technological University, Singapore
[2]Institute of Advanced Intelligence and Computing (IAIC), A*STAR, Singapore
[3]Department of Computer Science, Khalifa University, UAE
[4]Singapore University of Technology and Design, Singapore

Abstract

Patch-based transformers have emerged as efficient, improved architectures for long-horizon time series modeling. Yet existing approaches rely on *temporally-agnostic* patch construction, in which arbitrary starting positions and fixed lengths fracture temporal coherence by splitting natural transitions across patch boundaries. This naive segmentation often disrupts short-term dependencies and weakens representation learning. We propose a novel **Entro**py-Guided Dynamic **P**atch **E**ncoder (`EntroPE`), as a temporally informed framework that dynamically detects transition points via conditional entropy and dynamically places patch boundaries. This preserves temporal structure while retaining the computational benefits of patching. `EntroPE` consists of 2 key modules, namely an **Entropy-Guided Dynamic Patcher (EDP)** that applies information-theoretic criteria to locate natural temporal shifts and determine patch boundaries, and an **Adaptive Patch Encoder (APE)** that employs pooling and cross-attention to capture intra-patch dependencies and produce fixed-size latent representations. Extensive experiments on long-term forecasting, classification, and anomaly detection demonstrate that the proposed method improves both accuracy and efficiency, establishing entropy-guided dynamic patching as a promising new paradigm for time series modeling. Code is available at: `https://github.com/Sachithx/EntroPE`.

1 Introduction

Time series analysis is fundamental to numerous critical applications, including electricity load forecasting [13], financial market prediction [12, 41], healthcare monitoring [31], and environmental surveillance [32]. However, accurately modeling time-series data remains challenging due to inherent noise, complex temporal dependencies, and irregular patterns across multiple time horizons [19, 35]. The transformer architecture [36] has revolutionized language and sequence modeling across domains through self-attention mechanisms, leading to significant advances in time series analysis. Despite their promise in learning long-term temporal relationships, time series transformers remain suboptimal due to unique complexities such as non-stationarity, seasonality, and varying temporal granularities [40, 48].

Recent advances in time series transformers include Informer [51], Autoformer [44], and FEDformer [52], which address long-sequence modeling challenges through sparse attention mechanisms, decomposition techniques, and improved positional encodings while operating on point-wise inputs. Triformer [6] and PatchTST [27] introduced a patch-based input representation by dividing raw sequences into fixed-length patches, achieving substantial improvements in computational efficiency and performance. This demonstrated the critical impact of input tokenization strategies on transformer

*Corresponding author. `emadeldeenahmed.eldele@ku.ac.ae`

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

performance in time series. However, subsequent approaches have predominantly adopted similar *temporally agnostic* patching schemes, which may not fully capture the temporal coherence and statistical properties inherent in time-series data.

These patching strategies treat time as temporally homogeneous, partitioning sequences at arbitrary points without regard to the underlying temporal structure. *Temporally-agnostic* patching introduces 2 fundamental challenges. **(1)** They create inconsistent input representations between training and inference phases. Training observes patches starting at diverse temporal offsets, which result in patches that begin at various temporal positions relative to underlying patterns (e.g., a patch might start mid-trend in one sample but at a trend beginning in another). However, during inference, patches are extracted from predetermined positions in the input sequence, creating a systematic distribution shift in patch configurations. **(2)** Further, predetermined patch boundaries arbitrarily segment coherent temporal structures, ignoring frequent underlying dynamics (see Figure 1(**A**)). For example, a frequently occurring gradual trend change or seasonal transition may be split across multiple patches, breaking the natural temporal dependencies that are crucial for accurate pattern recognition.

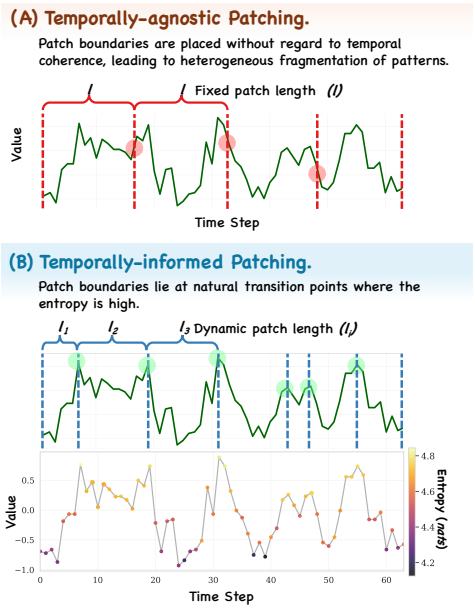


Figure 1: **Comparison of patching strategies.** (a) Temporally-agnostic fixed-length patching places boundaries at arbitrary positions, fragmenting coherent temporal patterns. (b) Our temporally informed approach places boundaries at entropy peaks, where the conditional entropy scale indicates natural transition points, thereby preserving temporal coherence within patches.

These challenges directly impact model performance. Fragmented temporal patterns create incomplete intra-patch representations, separating semantically related time points across patches and processing them independently. Additionally, the train-inference mismatch reduces the model's ability to generalize, as it must extrapolate to patch configurations with different statistical properties than those encountered during training. This is particularly detrimental for capturing rapid transitions and fine-grained temporal dynamics that require consistent temporal context.

We argue that patch boundaries should align with points of elevated **predictive uncertainty**, where the past provides insufficient information to determine the future. Formally, such moments are characterized by high conditional entropy of the next observation given historical context. Segmenting time at these entropy peaks localizes regime changes at patch boundaries while yielding patches whose interiors remain predictively coherent. Motivated by this principle, we introduce `EntroPE`, an **Entro**py-Guided Dynamic **P**atch **E**ncoder, which integrates *uncertainty-aware* boundary detection as a core architectural component. Unlike prior approaches that fix patch length or rely on heuristic change-point detectors, `EntroPE` derives segmentation from information-theoretic predictive structure, ensuring patch boundaries align with natural temporal transitions (Figure 1(**B**)). We then employ adaptive encoding mechanisms to process these variable-length patches while preserving intra-patch dependencies. This temporally-informed approach addresses both the train-inference mismatch and boundary fragmentation issues inherent in *temporally-agnostic* patching methods. The main contributions of this paper are:

- We propose `EntroPE`, an information-theoretic framework for dynamic patching in time series transformers. It isolates conditional predictive entropy as a causal boundary signal for time-series patching and shows it outperforms existing alternative segmentation criteria.
- We quantitatively show that conditional-entropy-induced boundaries produce 2.8× more internally homogeneous patches than fixed patching across 6 datasets, with 97.2% of detected boundaries coinciding with high-uncertainty positions.

- We show that this segmentation principle generalizes across long-term forecasting, classification, and anomaly detection, with efficient computational overhead from the lightweight entropy model.

2 Related Work

Time-Series Transformers and Patch Representations. Informer [51], Autoformer [44], and FEDformer [52] improve long-sequence forecasting through sparse attention, decomposition, and frequency-domain modeling, respectively. Patch representations offer a complementary approach by grouping observations into local segments. Following patch-based modeling in vision [9], Triformer [6] and PatchTST [27] demonstrate the benefits of temporal patching for efficient forecasting. Subsequent work explores cross-variable dependencies [50, 38], patch-independent self-supervised learning [18], and convolutional or MLP-based patch processing [33, 34].

Multi-Scale and Structured Representations. Several methods enrich patch representations across temporal scales. MSPatch [4] combines multiple patch resolutions, while Pathformer [5] adaptively routes computation across multi-scale Transformer branches. MOIRAI [43] uses multiple patch-size projections to accommodate different sampling frequencies. Rough Transformers [26] instead use multi-view path signatures to encode local and global temporal structure. These approaches illustrate how the scale and representation of temporal segments influence modeling efficiency and accuracy.

Adaptive Patching and Segmentation. Adaptation can operate on patch representations, selection, or boundaries. AdaPatch [21] addresses non-stationarity through patch-level normalization and reconstruction, while HDMixer [14] learns extendable patches. APN [24] learns time-aware patch boundaries and aggregation weights for irregular observations. SRSNet [46] selects and reassembles informative candidate patches, and TimeMosaic [8] adapts patch granularity to temporal heterogeneity. TimeSqueeze [1] combines a state-space encoder with dynamic segmentation, placing boundaries where consecutive-value changes exceed a threshold scaled by local signal power. ReinPatch [47] learns discrete patch-boundary decisions through reinforcement learning, jointly optimizing a patching policy and the downstream sequence model using Group Relative Policy Gradient under a minimum compression constraint. These methods provide mechanisms for adapting temporal resolution; `EntroPE` investigates learned predictive uncertainty as the boundary-selection signal.

Entropy-Guided Boundary Selection. The Byte Latent Transformer (BLT) [29] uses next-byte predictive entropy from a small autoregressive model to form variable-length patches for language modeling. In time-series forecasting, EAPformer [20] uses temporal entropy to adjust patch boundaries and combines this segmentation with multivariate dependency modeling. `EntroPE` adapts conditional predictive entropy to continuous time series using a separately trained, frozen autoregressive estimator. The entropy estimator and patch encoder receive quantized input tokens, and the forecasting head produces continuous predictions. To examine the contribution of the boundary criterion, we compare learned predictive entropy with empirical window entropy and alternative boundary signals under a matched downstream architecture.

3 Preliminaries

3.1 Problem Formulation

Given a multivariate time series $X = [x^{(1)}, \ldots, x^{(C)}] \in \mathbb{R}^{C \times L}$ with C channels and look-back length L, we address forecasting, classification, and anomaly detection. Following the channel-independence principle [27], our architecture f_ϕ processes each channel independently to obtain $\mathbf{z}^{(c)} = f_\phi(x^{(c)})$, producing $Z = [\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(C)}] \in \mathbb{R}^{C \times d}$ that is projected to a task-specific output. The pipeline consists of entropy-guided dynamic patching, adaptive patch encoding, a global transformer over patches, and a fusion decoder, illustrated in Figure 2.

`EntroPE` operates in 2 phases. **First**, we pre-train a lightweight entropy model on unlabeled data (self-supervision) to estimate next-step predictive uncertainty. **Second**, we freeze this model and use it to provide patch boundaries during task-specific training; downstream representations and output predictions remain continuous throughout.

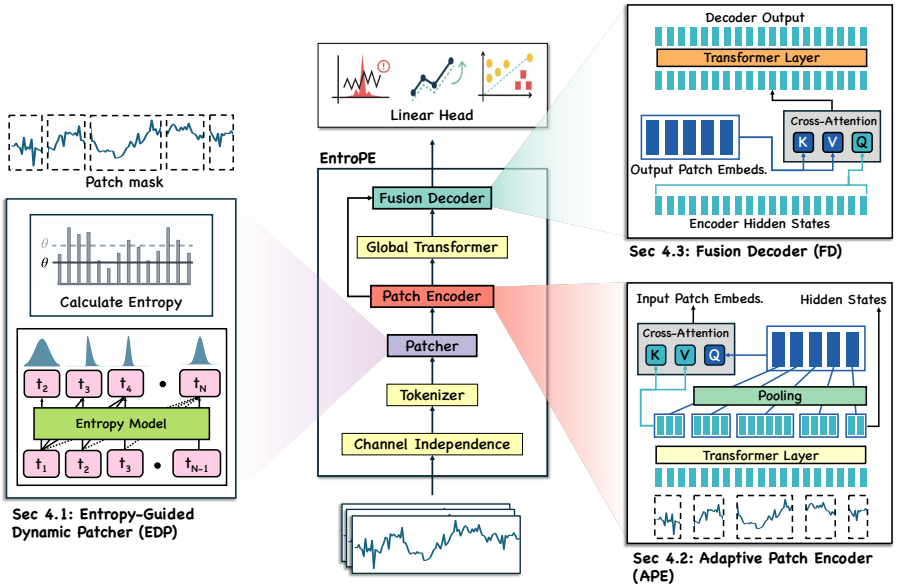


Figure 2: EntroPE architecture. (A) **EDP** uses a frozen causal Transformer to estimate next-token entropy and selects patch boundaries from entropy levels or increases. (B) **APE** aggregates intra-patch dependencies via pooling and cross-attention into fixed-size embeddings. (C) **FD** combines global patch context with local time-point representations.

3.2 Entropy Model

Quantization. For each univariate sequence $x = [x_1, \ldots, x_L]$, we apply a quantizer $q : \mathbb{R} \to \mathcal{V}$ with $|\mathcal{V}| = V$ to obtain tokens $\tau_t = q(x_t)$. Bin boundaries are determined from training-data statistics with $\epsilon = 0.005$ coverage:

$$R = \max\big(|Q_{\epsilon/2}(X_{\text{train}})|, |Q_{1-\epsilon/2}(X_{\text{train}})|\big), \tag{1}$$

and $[-R, R]$ is uniformly divided into V bins. Crucially, quantization is used *only* to estimate the entropy signal. Unlike Chronos [2], our prediction heads operate on continuous predictions and are not constrained to discrete vocabulary decoding, sidestepping the dequantization-range limitation.

Architecture and Training. A 2-layer GPT-style causal transformer [30] with 16 embedding dimension (≈16K parameters) is trained for next-token prediction via cross-entropy:

$$\theta^*_{\text{ent}} = \arg\min_{\theta_{\text{ent}}} \mathbb{E}_{X \sim \mathcal{D}_{\text{pretrain}}} \mathcal{L}_{\text{CE}}(X; \theta_{\text{ent}}). \tag{2}$$

Once converged, we freeze θ^*_{ent} and use it only to compute entropy; it is not used to generate predictions. Boundaries can be precomputed per sample. However, even computing boundaries during downstream training increased epoch duration by only about 0.1-1.0 seconds across the 6 measured datasets (Table. 21).

4 EntroPE Architecture

4.1 Entropy-Guided Dynamic Patcher (EDP)

For input $X \in \mathbb{R}^{C \times L}$ and channel c, we tokenize and compute the conditional entropy at each position:

$$H(x_t^{(c)}) = -\sum_{v \in \mathcal{V}} p_{\theta^*_{\text{ent}}}(\tau_{t+1}^{(c)} = v \mid \tau_{\le t}^{(c)}) \log p_{\theta^*_{\text{ent}}}(\tau_{t+1}^{(c)} = v \mid \tau_{\le t}^{(c)}). \tag{3}$$

Write $h_t^{(c)} = H(x_t^{(c)})$ for the predictive entropy at position t. A high value means that the frozen estimator is uncertain about the next token given the preceding tokens in channel c.

Sample-Adaptive Thresholds. Each forecasting configuration uses one of 2 boundary scores: the entropy level $h_t^{(c)}$ or its increase $\Delta h_t^{(c)} = h_t^{(c)} - h_{t-1}^{(c)}$. We compute the corresponding cutoff separately for each sample and channel, using the quantile level α:

$$\theta_{\text{abs}}^{(c)} = Q_\alpha\Big(\{h_t^{(c)}\}_{t=1}^{L}\Big), \qquad \gamma_{\text{rel}}^{(c)} = Q_\alpha\Big(\{\Delta h_t^{(c)}\}_{t=2}^{L}\Big). \tag{4}$$

Table 11 gives the mode and quantile level used for each dataset and forecast horizon.

Boundary Placement. The first observation starts a patch, so $M_1^{(c)} = 1$. For $1 \le t < L$, the selected rule determines whether a new patch starts at the next observation:

$$M_{t+1}^{(c)} = \begin{cases} \mathbf{1}[h_t^{(c)} > \theta_{\text{abs}}^{(c)}], & \text{entropy-level mode,} \\ 1, & \text{entropy-rise mode, } t = 1, \\ \mathbf{1}[\Delta h_t^{(c)} > \gamma_{\text{rel}}^{(c)}], & \text{entropy-rise mode, } t \ge 2. \end{cases} \tag{5}$$

Consecutive starts yield one-observation patches. When a maximum patch length is specified, we also split patches that exceed it. Let $1 = b_1^{(c)} < \cdots < b_{P^{(c)}+1}^{(c)} = L + 1$ denote the resulting boundaries, including the terminal endpoint. Patch j contains observations $b_j^{(c)}, \ldots, b_{j+1}^{(c)} - 1$ and has length $\ell_j^{(c)} = b_{j+1}^{(c)} - b_j^{(c)}$.

Motivation for Predictive Entropy. With fixed-length patching, shifting the start of an input window can split the same temporal pattern at a different point. `EntroPE` uses the estimator's next-token uncertainty to guide the placement of boundaries within each window. For quantized observations, the true conditional entropy lower-bounds the expected next-token negative log-likelihood (Proposition 1). Splitting a patch preserves its length-weighted mean entropy (Proposition 2). We empirically assess the practical value of the learned boundary signal through within-patch variance, boundary concentration (Section 5.3) and forecasting error under alternative segmentation rules.

4.2 Adaptive Patch Encoder (APE)

The EDP produces variable-length patches whose patch count and patch length vary across samples. The APE converts these into fixed-size embeddings via pooling-initialized cross-attention, inspired by Perceiver architecture [17] and adapted to continuous time-point embeddings (also used in BLT [29] for variable-length encoding in language modeling).

We map each quantized observation to a token embedding $\mathbf{e}_t^{(c)} = E(q(x_t^{(c)}))$. The local encoder transforms these embeddings into token states $\mathbf{h}_t^{(c)}$. For patch j, we initialize its representation by max-pooling the states at positions $t \in \mathcal{T}_j^{(c)}$, then refine it by cross-attention over those positions.

$$\mathbf{z}_j^{(c),0} = \text{MaxPool}\big(\{\mathbf{h}_t^{(c)}\}_{t\in\mathcal{T}_j^{(c)}}\big), \qquad \mathbf{z}_j^{(c),n} = \mathbf{z}_j^{(c),n-1} + W_o \sum_{i\in\mathcal{T}_j^{(c)}} \alpha_{ji}^{(n)} W_v\big(\mathbf{h}_i^{(c),n-1}\big), \tag{6}$$

with attention weights $\alpha_{ji}^{(n)}$ computed by softmax over $\mathcal{T}_j^{(c)}$. The output is patch embeddings $P^{(c)} \in \mathbb{R}^{P^{(c)}\times d_p}$ and refined token states $H^{(c)} \in \mathbb{R}^{L\times d_t}$.

For batch processing, we pad to $P_{\max} = \max_{b,c} P^{(b,c)}$ and apply attention masks $\mathcal{M}_{bc,j} = \mathbb{1}[j \le P^{(b,c)}]$ to ignore padded slots during self-attention.

4.3 Fusion Decoder (FD)

A standard transformer captures inter-patch dependencies, $\tilde{P}^{(c)} = \text{Transformer}(P^{(c)}, \mathcal{M}) \in \mathbb{R}^{P^{(c)}\times d_p}$. The fusion decoder then injects this global context back into the token-level representation via cross-attention, where token states query patch representations:

$$\tilde{H}^{(c)} = H^{(c)} + W_o\,\text{softmax}\left(\frac{W_q(H^{(c)})W_k(\tilde{P}^{(c)})^\top}{\sqrt{d_k}} + A^{(c)}\right) W_v(\tilde{P}^{(c)}). \tag{7}$$

Table 1: Multivariate forecasting results, averaged over horizons $T \in \{96, 192, 336, 720\}$ with fixed input length $L = 96$. **Red** marks the best result and **Blue** the second best. Per-horizon T forecasting results are in Appendix Table 12.

Model	ETTh1		ETTh2		ETTm1		ETTm2		Weather		Solar		Electricity		Traffic		**Overall**	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
EntroPE	**0.427**	**0.431**	**0.369**	**0.402**	**0.379**	**0.394**	0.285	0.332	**0.247**	**0.277**	**0.239**	0.275	0.193	0.284	0.498	0.298	**0.330**	0.336
TimeMosaic	**0.437**	**0.426**	**0.375**	**0.393**	0.389	**0.384**	0.283	**0.319**	0.257	**0.273**	0.268	0.269	0.199	0.276	0.485	0.290	0.336	**0.329**
CALF	0.443	0.435	0.379	0.402	0.403	0.397	0.285	0.324	0.257	0.277	**0.242**	**0.243**	0.192	**0.271**	**0.467**	**0.285**	**0.333**	**0.329**
iTransformer	0.457	0.449	0.396	0.414	0.421	0.419	0.293	0.335	0.261	0.281	0.244	**0.268**	**0.182**	**0.271**	**0.442**	**0.289**	0.337	0.341
TimeMixer	0.483	0.454	0.397	0.413	0.389	0.401	**0.282**	0.326	**0.250**	0.278	0.250	0.287	**0.186**	0.276	0.495	0.310	0.341	0.343
FilterTS	0.446	0.434	0.381	0.402	0.389	0.398	**0.282**	**0.324**	0.258	0.282	0.274	0.299	0.200	0.284	0.487	0.326	0.339	0.344
TSLANet	0.464	0.449	0.381	0.406	**0.383**	0.398	0.286	0.330	0.256	0.279	0.263	0.284	0.201	0.284	0.489	0.307	0.340	0.342
TimeFFM	0.459	0.439	0.375	0.402	0.402	0.397	0.290	0.335	0.273	0.292	0.375	0.386	0.224	0.310	0.669	0.432	0.383	0.374
HDMixer	0.452	0.438	0.383	0.405	0.398	0.402	0.284	0.329	0.256	0.285	0.313	0.336	0.199	0.301	0.543	0.379	0.353	0.359
TimeBase	0.450	0.438	0.394	0.415	0.438	0.432	0.295	0.338	0.256	0.285	0.402	0.411	0.247	0.337	0.729	0.392	0.401	0.381
FEDformer	0.443	0.460	0.446	0.457	0.455	0.460	0.308	0.353	0.320	0.371	0.254	0.363	0.218	0.333	0.606	0.374	0.381	0.396
PatchTST	0.461	0.455	0.395	0.416	0.396	0.409	0.294	0.338	0.256	0.279	0.252	0.294	0.201	0.292	0.476	0.304	0.341	0.348
DLinear	0.462	0.458	0.569	0.522	0.404	0.408	0.355	0.402	0.266	0.319	0.328	0.399	0.211	0.299	0.627	0.388	0.403	0.399
TimesNet	0.477	0.465	0.414	0.428	0.404	0.414	0.298	0.335	0.262	0.288	0.345	0.365	0.251	0.339	0.652	0.358	0.388	0.374

Here $A^{(c)}$ is the token-to-patch attention mask, with zero for allowed pairs and $-\infty$ otherwise. With a decoder window of one patch, each token attends to its own patch representation, which already incorporates inter-patch context from the global Transformer. The decoder further processes the fused token states with a local Transformer block.

For forecasting, we flatten the decoder output, apply a linear projection, and reverse the instance normalization to obtain predictions on the original scale. We give the classification and anomaly-detection heads and their training objectives in Appendix B.4.

5 Experiments

We evaluate EntroPE on long-term forecasting, classification, and anomaly detection tasks. We use the same backbone architecture across all 3 tasks; only the task-specific head changes. Section 5.1 presents the main forecasting results; Section 5.2 demonstrates generalization to classification and anomaly detection; Section 5.3 empirically validates the entropy mechanism; and Section 5.4 isolates and analyzes each architectural component. Unless otherwise stated, all experimental results are averaged across 3 random seeds. Full implementation settings, dataset descriptions, and extended result tables are presented in the Appendix (A,B,C).

5.1 Long-Term Forecasting

Datasets. To examine forecasting performance we conduct comprehensive evaluations on 8 widely-used long-term multivariate forecasting datasets, namely, the ETT family (ETTh1, ETTh2, ETTm1, ETTm2) that encompasses a range of scenarios in energy transfer technology at hourly and 15-minute resolution, Weather that offers insights into 21 meteorological variables over time, Electricity (ECL) featuring electricity consumption data from 321 clients, Solar-Energy that captures 10-minute power production records from 137 photovoltaic plants, and Traffic that records hourly road occupancy from 862 freeway sensors in the San Francisco Bay Area. ECL, Solar-Energy, and Traffic extend the evaluation to the high-dimensional, strongly periodic regime, with an order of magnitude more channels than the ETT family and weather. Table 6 presents detailed dataset descriptions.

Baselines. We benchmark against 13 state-of-the-art baselines covering 6 different categories. (1) Transformer-based models: TimeMosaic [8], iTransformer [25], PatchTST [27], FEDformer [52]; (2) CNN-based model: TimesNet [45], TSLANet [11]; (3) MLP-based models: TimeMixer [37], HDMixer [14], DLinear [48]; (4) Lightweight models: TimeBase [15]; (5) LLM-aligned and foundation models: Time-FFM [23], CALF [22]; (6) Filter-based model: FilterTS [39]. (Further details on baselines are discussed in Appendix A.4.)

Results. EntroPE achieves the best MSE performance on 6 out of 9 cases, including the overall average performance across all datasets (Table 1). Compared with PatchTST, EntroPE reduces

average MSE by approximately 3.2% while using 90.9% fewer parameters and 78.5% fewer counted MACs across the 8 datasets, retaining a channel-independent design. The other cases where EntroPE is not first are illustrative: on Electricity (321 channels), iTransformer and CALF dominate via cross-channel attention; on ETTm2, TimeMosaic uses larger models (~8M parameters versus our ≤ 0.8M). EntroPE is competitive with CALF (~180M) and TimeMosaic (~8M) at 10-225$\times$ fewer parameters, indicating that gains arise from boundary placement rather than capacity. Compared to the other adaptive patching method HDMixer, EntroPE consistently outperforms it at matched compute. Further details on efficiency and model size are demonstrated in Figure 6- 7 and Table 19.

Longer Input Context Forecasting. Figure 3 examines forecasting performance of ETTm1, Weather and Solar as the input length increases. We additionally evaluate $L = 512$ with horizons $T = 96$ and 480, and $T = 720$ on Solar, comparing dynamic and static patching at small, medium, and large model sizes. Within each comparison, we fix the training budget, and the static patch length matches the average dynamic patch length. At $512 \rightarrow 96$, dynamic patching reduces MSE across all 5 evaluated datasets and all 3 model sizes, with reported reductions of 1.9-12.7%. At longer horizons, the gains depend on the dataset and model size, with some settings favoring static patching. Complete MSE/MAE results and relative changes are provided in Appendix C.3.

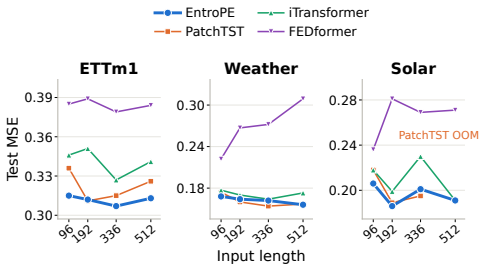


Figure 3: Test MSE for input lengths.

5.2 Classification and Anomaly Detection

Classification. On 14 UEA multivariate classification benchmarks [3] with the forecasting head replaced by a classification head. As demonstrated by Table 2, EntroPE achieves 76.01% average accuracy, ahead of TSLANet (72.65%), TS-TCC [10] (70.98%), ROCKET [7] (65.18%), PatchTST (68.53%), and GPT4TS [53] (66.72%). The dynamic patcher captures discriminative temporal patterns without architectural changes. We also include the Cricket, EigenWorms, MotorImagery, and StandWalkJump datasets from the UEA repository, which further demonstrate that EntroPE supports very long input contexts.

Table 2: Classification accuracy (%) on selected UEA multivariate datasets. **Red** marks the best result and **Blue** the second best.

Dataset	EntroPE	TSLANet	GPT4TS	TimesNet	ROCKET	CrossF.	PatchTST	MLP	TS-TCC
EthanolConcentration	29.10	30.42	25.48	27.73	**42.58**	**34.98**	28.90	33.46	32.32
FaceDetection	**69.10**	66.77	65.58	67.47	64.70	66.17	**68.96**	67.42	63.05
Handwriting	**57.40**	**57.88**	34.56	26.18	48.47	26.24	26.00	22.47	47.76
Heartbeat	**80.00**	**77.56**	36.59	74.48	69.76	76.59	76.59	73.17	77.07
JapaneseVowels	**99.20**	**99.19**	98.11	97.83	95.68	98.92	98.65	97.84	97.30
PEMS-SF	**88.60**	83.82	87.28	88.13	75.10	82.08	**88.44**	82.08	86.71
SelfRegulationSCP1	**92.90**	91.81	91.47	77.43	84.64	**92.49**	89.76	88.40	91.13
SelfRegulationSCP2	**63.10**	**61.67**	51.67	52.84	54.44	53.33	54.44	51.67	53.89
SpokenArabicDigits	**99.80**	**99.91**	99.36	98.36	99.20	96.41	99.68	96.68	99.77
UWaveGestureLibrary	**94.40**	**91.25**	84.38	83.13	76.03	81.56	80.00	81.88	86.25
Cricket	83.33	**98.61**	94.44	91.67	**100.00**	91.67	95.83	75.00	94.44
EigenWorms	**74.81**	**72.52**	56.49	53.44	9.92	52.67	46.56	54.20	47.33
MotorImagery	59.00	59.00	**62.00**	**61.00**	52.00	50.00	59.00	47.00	50.00
StandWalkJump	**73.33**	26.67	46.67	60.00	40.00	46.67	46.67	53.33	**66.67**
Average	**76.01**	**72.65**	66.72	68.55	65.18	67.84	68.53	66.04	70.98

Anomaly Detection. As demonstrated by Table 3, on 5 standard anomaly detection benchmarks (SMD, MSL, SMAP, SWaT, PSM), EntroPE matches or exceeds baselines on F1 score. We also include ETSformer [42] and LightTS [49] as baselines. These datasets have a 100-length input context window. Therefore, we train the entropy model with a block size of 100 for anomaly detection.

5.3 Empirical Validation of Entropy-Guided Segmentation

Table 3: Anomaly detection F1 score (%) on 5 benchmark datasets. **Red** marks the best result and **Blue** the second best.

Dataset	EntroPE	TSLANet	GPT4TS	TimesNet	PatchTST	ETSformer	FEDformer	LightTS	DLinear
SMD	**87.10**	**87.91**	86.89	84.61	84.62	83.13	85.08	82.53	77.10
MSL	**85.30**	83.32	82.45	81.84	78.70	**85.03**	78.57	78.95	84.88
SMAP	**77.93**	**75.96**	72.88	69.39	68.82	69.50	70.76	69.21	69.26
SWaT	**94.30**	92.80	**94.23**	93.02	85.72	84.91	93.19	93.33	87.52
PSM	**97.70**	**97.73**	97.13	97.34	96.08	91.76	97.23	97.15	93.55
Average	**88.47**	**87.54**	86.72	85.24	82.79	82.87	84.97	84.23	82.46

Table 4: Patch-level diagnostics on 500 test samples.

Dataset	Variance ratio ↓ Dynamic	Variance ratio ↓ Static	Gain	Boundary conc.
ETTh1	0.195 ± 0.10	0.516 ± 0.20	2.6×	0.974 ± 0.02
ETTh2	0.223 ± 0.22	0.481 ± 0.26	2.2×	0.977 ± 0.02
ETTm1	0.066 ± 0.06	0.208 ± 0.13	3.1×	0.971 ± 0.02
ETTm2	0.111 ± 0.11	0.307 ± 0.22	2.8×	0.978 ± 0.02
Electricity	0.172 ± 0.09	0.484 ± 0.26	2.8×	0.957 ± 0.01
Weather	0.052 ± 0.08	0.175 ± 0.21	3.4×	0.977 ± 0.02
Mean	**0.137**	**0.362**	**2.8×**	**0.972**

We test whether entropy-guided boundaries produce the patch-level properties that motivate the design. On 500 test samples per dataset, we measure 2 quantities. (1) The *intra-patch variance ratio* is the average within-patch variance divided by the total sequence variance; lower values indicate predictively coherent patch interiors. (2) The *boundary entropy concentration* is the fraction of detected boundaries that coincide with the top-α highest-entropy positions. Across all 6 datasets, dynamic patches are $2.8\times$ more internally homogeneous than fixed patches, and 97.2% of boundaries fall at high-entropy positions (Table 4). The mechanism behaves as the design intends: predictable regimes are grouped into patches, while high-uncertainty transitions become boundaries.

5.4 Ablation Studies

Boundary Signal Analysis. A core question is whether *conditional predictive entropy* is meaningfully different from simpler change-point heuristics. We hold the architecture fixed (all variants share the same APE, global transformer, and FD) and replace only the boundary signal, comparing 6 alternatives on ETTm1, ETTh2, and Weather with 3 different input context lengths $L \in \{96, 336, 512\}$. Figure 4 shows that conditional predictive entropy outperforms variance change-point, frequency spectral shift, CUSUM, local difference, EAPformer-style empirical Shannon entropy [20], and static boundaries under matched conditions.

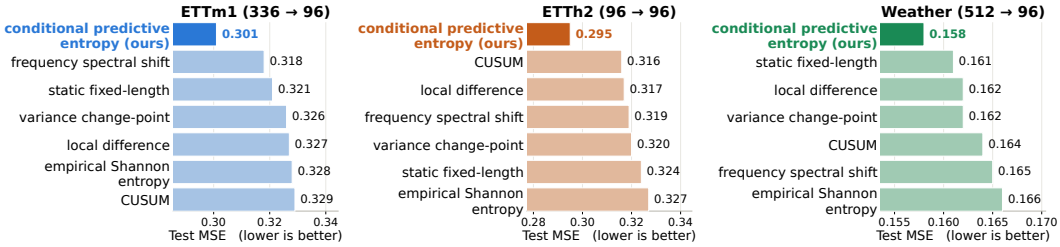


Figure 4: Boundary-signal ablation on ETTm1, ETTh2 and Weather different input lengths.

Architectural Components Analysis. We progressively remove the EDP, APE, and FD on 4 datasets at prediction horizons $T \in \{336, 720\}$. The full architecture wins 7 of 8 cases (Table 5); removing the EDP (replacing dynamic patches with static ones) confirms that boundary placement contributes more than encoder design or decoder fusion. This ablation experiment used a single seed MSE on each run.

Entropy Model Training and Threshold Analysis. In Figure 5, each panel varies one control on the ETT-family datasets at input length $L = 96$; the y-axis is the change in test MSE against the reference configuration, averaged over $T \in 96, 336$ (negative is better). The reference is a vocabulary-256 entropy model trained for 10 epochs with the per-dataset percentile threshold, so the curves pass through zero at 10 epochs in (a) and at vocabulary 256 in (b). Extending training to 20 or

Table 5: Component ablation (MSE ↓). **Red** marks the best result and **Blue** the second best.

Dataset	T	Full Dyn.	−EDP Stat(1)	−EDP Stat(8)	−EDP Stat(16)	−EDP−APE Pool+FD	−EDP−APE−FD Pool+Flat
ETTh1	336	**0.429**	**0.425**	0.441	0.444	0.438	0.519
	720	**0.439**	**0.460**	0.469	0.477	0.461	0.527
ETTh2	336	**0.355**	**0.428**	0.435	0.439	0.439	0.462
	720	**0.397**	**0.448**	0.470	0.463	0.466	0.456
ETTm1	336	**0.393**	**0.401**	0.409	0.415	0.402	0.427
	720	**0.445**	**0.451**	0.459	0.452	0.457	0.483
Weather	336	**0.258**	**0.261**	0.265	0.270	0.262	0.262
	720	**0.341**	**0.343**	0.353	0.359	0.347	0.346

50 epochs yields no consistent forecasting gain. Most vocabulary settings change MSE by at most about 2%, with the largest increase near 3% on ETTm2 at $V = 64$. The effect of a fixed absolute threshold depends on the dataset. For example, a threshold of 5.0 nats reduces MSE by about 2% on ETTh1 but increases it on the other 3 datasets. For the entropy-rise rule, all tested fixed thresholds yield higher MSE than the percentile reference, although the gap stays below 2% on all 4 datasets at 0.7 and 1.0 nats.

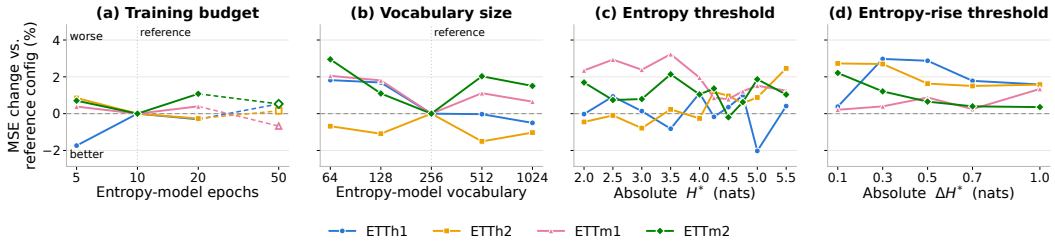


Figure 5: Sensitivity of EntroPE to the entropy model and its thresholding rules.

Efficiency Analysis. The quantile threshold α acts as a computational control knob: lower values yield finer segmentation, higher values produce coarser patches. Figure 6 compares forecasting error and computational cost across the 4 ETT datasets. EntroPE achieves the lowest ETT-family average MSE among the plotted models at roughly 42 million counted MACs per sample. Models such as HDMixer and FilterTS use fewer MACs but have higher average MSE, placing EntroPE at the low-error end of the observed Pareto frontier.

Figure 7 shows at forecast horizon $T = 96$ and lookbacks are ETTm1 $L = 336$; Weather $L = 512$; Solar $L = 192$. Numbered points are static patching at fixed patch length $P \in 1, 2, 4, 6, 8, 12$; the highlighted point is EntroPE's entropy-based dynamic patching (threshold $= 0.95$). Timing covers the full forward pass, including entropy estimation and boundary generation; the static variants bypass

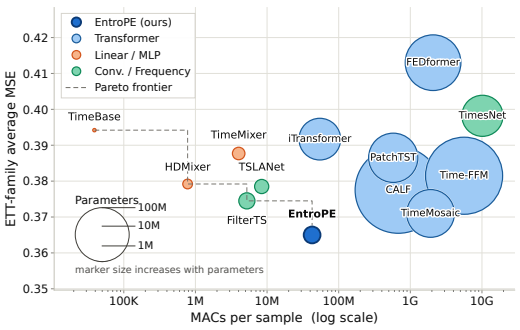


Figure 6: ETT-family average MSE versus counted MACs per sample. Bubble size indicates parameter count.

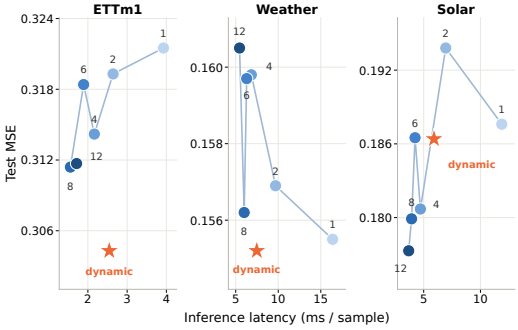


Figure 7: Test MSE versus inference latency per sample for dynamic and fixed-length patching at $T = 96$.

the entropy model entirely, so the comparison is conservative with respect to dynamic patching. On ETTm1 and Weather, dynamic patching achieves lower test MSE than all 6 tested fixed patch lengths and runs faster than the finest fixed segmentation ($P = 1$). On Solar, the fixed patches of length $P = 12$ achieve both lower MSE and lower latency than dynamic patching.

Qualitative Analysis. Figure 8 illustrates how patch placement varies across and within input windows. The upper panels show the input signal and the lower panels show estimated next-token entropy in nats. Alternating shaded and unshaded regions mark successive patches. The dashed red line indicates the absolute entropy threshold. At $L = 96$, the ETTm1 and Weather examples contain 6 and 7 patches, with mean lengths of 16 and 14 time steps, respectively, while the Solar example contains 7 patches with a mean length of 14. The longer Solar example ($L = 336$) shows how patch spacing changes within a single window. Several boundaries cluster around the burst of activity near time step 200, where estimated predictive entropy is high. Boundaries are sparse across the long, nearly flat intervals where entropy is low. This is a strong example of how computation is allocated based on flat (low information density) regions.

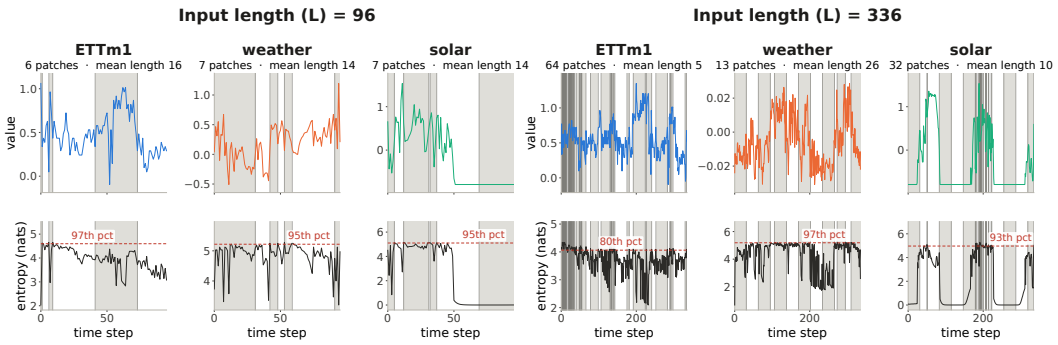


Figure 8: Entropy-guided patch placement on example windows from ETTm1, Weather, and Solar.

6 Limitations

EntroPE's benefits depend on the dataset, forecast horizon, and model size. Improvements are small and mixed on M4-Monthly ($36 \rightarrow 18$), while static patching achieves lower MSE on Solar for the small model at horizons 480 and 720 and the large model at horizon 720 (Tables 13-14). Boundary placement is also sensitive to noise: on ETTh1, Jaccard similarity to clean-signal boundaries decreases from 0.63 at 40 dB SNR to 0.28 at 2 dB, despite a stable boundary count (Table 24). The dataset-specific entropy estimator may require retraining under domain shifts, and channel-independent segmentation can miss coordinated multivariate transitions. Furthermore, recurring boundaries on strongly periodic datasets suggest that simpler period-based segmentation may provide comparable benefits. Future work includes noise-robust and transferable entropy estimators, cross-channel boundary coordination, and mechanisms that adaptively select between dynamic, fixed, and period-based segmentation according to signal characteristics and computational constraints.

7 Conclusion

We introduced EntroPE, a dynamic patch encoding framework that uses entropy-guided boundary detection to identify natural temporal transitions in time series. By placing patch boundaries at information-rich transitions rather than fixed intervals, EntroPE preserves temporal coherence while enabling efficient variable-length processing. This information-theoretic approach improves performance across downstream tasks with practical computational benefits. Beyond empirical gains, EntroPE demonstrates the value of respecting intrinsic temporal structure, opening new directions in adaptive context-aware sequence modeling.

Acknowledgments

We thank the anonymous reviewers for their helpful feedback. Sachith Abeywickrama acknowledges support from the Agency for Science, Technology and Research (A*STAR), Singapore, through the Singapore International Graduate Award (SINGA). This research is supported by the Ministry of Education, Singapore, under the MOE Tier 1 grant (SKI 2021_08_03), and by the Kwan Im Thong Hood Cho Temple Chair Professorship. This research is also supported by the KU Research Center for Advanced Intelligent Systems (KU-AIS) at Khalifa University of Science and Technology.

References

[1] Sravan Kumar Ankireddy, Nikita Seleznev, Nam H Nguyen, Yulun Wu, Senthil Kumar, Furong Huang, and C Bayan Bruss. Timesqueeze: Dynamic patching for efficient time series forecasting. *arXiv preprint arXiv:2603.11352*, 2026.

[2] Abdul Fatir Ansari, Lorenzo Stella, Ali Caner Türkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda-Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Bernie Wang. Chronos: Learning the language of time series. *Trans. Mach. Learn. Res.*, 2024, 2024.

[3] Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. The UEA multivariate time series classification archive, 2018. *arXiv preprint arXiv:1811.00075*, 2018.

[4] Yizhi Cao, Zijian Tian, Wenjie Guo, and Xinggao Liu. MSPatch: A multi-scale patch mixing framework for multivariate time series forecasting. *Expert Syst. Appl.*, 273:126849, 2025. doi: 10.1016/J.ESWA.2025.126849.

[5] Peng Chen, Yingying Zhang, Yunyao Cheng, Yang Shu, Yihang Wang, Qingsong Wen, Bin Yang, and Chenjuan Guo. Pathformer: Multi-scale transformers with adaptive pathways for time series forecasting. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[6] Razvan-Gabriel Cirstea, Chenjuan Guo, Bin Yang, Tung Kieu, Xuanyi Dong, and Shirui Pan. Triformer: Triangular, variable-specific attentions for long sequence multivariate time series forecasting. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 1994–2001. ijcai.org, 2022. doi: 10.24963/IJCAI.2022/277.

[7] Angus Dempster, François Petitjean, and Geoffrey I. Webb. ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Min. Knowl. Discov.*, 34(5):1454–1495, 2020. doi: 10.1007/S10618-020-00701-Z.

[8] Kuiye Ding, Fanda Fan, Chunyi Hou, Zheya Wang, Lei Wang, Zhengxin Yang, and Jianfeng Zhan. TimeMosaic: Temporal heterogeneity guided time series forecasting via adaptive granularity patch and segment-wise decoding. In *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pages 20790–20798. AAAI Press, 2026. doi: 10.1609/AAAI.V40I25.39218.

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[10] Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, Chee Keong Kwoh, Xiaoli Li, and Cuntai Guan. Time-series representation learning via temporal and contextual contrasting. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 2352–2359. ijcai.org, 2021. doi: 10.24963/IJCAI.2021/324.

[11] Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, and Xiaoli Li. TSLANet: Rethinking transformers for time series representation learning. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pages 12409–12428. PMLR / OpenReview.net, 2024.

[12] Thomas Fischer and Christopher Krauss. Deep learning with long short-term memory networks for financial market predictions. *Eur. J. Oper. Res.*, 270(2):654–669, 2018. doi: 10.1016/J.EJOR.2017.11.054.

[13] Alberto Gasparin, Slobodan Lukovic, and Cesare Alippi. Deep learning for time series forecasting: The electric load case. *CAAI Trans. Intell. Technol.*, 7(1):1–25, 2022. doi: 10.1049/CIT2.12060.

[14] 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 *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2024, February 20-27, 2024, Vancouver, Canada*, pages 12608–12616. AAAI Press, 2024. doi: 10.1609/AAAI.V38I11.29155.

[15] Qihe Huang, Zhengyang Zhou, Kuo Yang, Zhongchao Yi, Xu Wang, and Yang Wang. TimeBase: The power of minimalism in efficient long-term time series forecasting. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025.

[16] Songtao Huang, Zhen Zhao, Can Li, and Lei Bai. TimeKAN: Kan-based frequency decomposition learning architecture for long-term time series forecasting. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025.

[17] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and João Carreira. Perceiver: General perception with iterative attention. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 4651–4664. PMLR, 2021.

[18] Seunghan Lee, Taeyoung Park, and Kibok Lee. Learning to embed time series patches independently. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[19] Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A*, 379(2194):20200209, 2021.

[20] Jiahao Ling, Xuan Yang, Shimin Gong, and Bo Gu. EAPformer: Entropy-aware patch transformer for multivariate long-term time series forecasting. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management, CIKM 2025, Seoul, Republic of Korea, November 10-14, 2025*, pages 1850–1860. ACM, 2025. doi: 10.1145/3746252.3761055.

[21] Kun Liu, Zhongjie Duan, Cen Chen, Yanhao Wang, Dawei Cheng, and Yuqi Liang. AdaPatch: Adaptive patch-level modeling for non-stationary time series forecasting. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management, CIKM 2025, Seoul, Republic of Korea, November 10-14, 2025*, pages 1882–1891. ACM, 2025. doi: 10.1145/3746252.3761360.

[22] Peiyuan Liu, Hang Guo, Tao Dai, Naiqi Li, Jigang Bao, Xudong Ren, Yong Jiang, and Shu-Tao Xia. CALF: aligning llms for time series forecasting via cross-modal fine-tuning. In *Thirty-Ninth AAAI Conference on Artificial Intelligence, Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence, Fifteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2025, Philadelphia, PA, USA, February 25 - March 4, 2025*, pages 18915–18923. AAAI Press, 2025. doi: 10.1609/AAAI.V39I18.34082.

[23] Qingxiang Liu, Xu Liu, Chenghao Liu, Qingsong Wen, and Yuxuan Liang. Time-FFM: Towards lm-empowered federated foundation model for time series forecasting. In *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

[24] Xvyuan Liu, Xiangfei Qiu, Xingjian Wu, Zhengyu Li, Chenjuan Guo, Jilin Hu, and Bin Yang. Rethinking irregular time series forecasting: A simple yet effective baseline. In *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pages 23873–23881. AAAI Press, 2026. doi: 10.1609/AAAI.V40I28.39563.

[25] 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, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[26] Fernando Moreno-Pino, Alvaro Arroyo, Harrison Waldon, Xiaowen Dong, and Álvaro Cartea. Rough transformers: Lightweight and continuous time series modelling through signature patching. In *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

[27] 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, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[28] Wenzhe Niu, Zongxia Xie, Yanru Sun, Wei He, Man Xu, and Chao Hao. LangTime: A language-guided unified model for time series forecasting with proximal policy optimization. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025.

[29] Artidoro Pagnoni, Ramakanth Pasunuru, Pedro Rodríguez, John Nguyen, Benjamin Muller, Margaret Li, Chunting Zhou, Lili Yu, Jason E. Weston, Luke Zettlemoyer, Gargi Ghosh, Mike Lewis, Ari Holtzman, and Srini Iyer. Byte latent transformer: Patches scale better than tokens. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 9238–9258. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.ACL-LONG.453.

[30] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[31] Ben Y. Reis and Kenneth D. Mandl. Time series modeling for syndromic surveillance. *BMC Medical Informatics Decis. Mak.*, 3:2, 2003. doi: 10.1186/1472-6947-3-2.

[32] Richard L Smith. Extreme value analysis of environmental time series: an application to trend detection in ground-level ozone. *Statistical Science*, pages 367–377, 1989.

[33] Artyom Stitsyuk and Jaesik Choi. xPatch: Dual-stream time series forecasting with exponential seasonal-trend decomposition. In *Thirty-Ninth AAAI Conference on Artificial Intelligence, Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence, Fifteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2025, Philadelphia, PA, USA, February 25 - March 4, 2025*, pages 20601–20609. AAAI Press, 2025. doi: 10.1609/AAAI.V39I19.34270.

[34] Peiwang Tang and Weitai Zhang. Unlocking the power of patch: Patch-based MLP for long-term time series forecasting. In *Thirty-Ninth AAAI Conference on Artificial Intelligence, Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence, Fifteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2025, Philadelphia, PA, USA, February 25 - March 4, 2025*, pages 12640–12648. AAAI Press, 2025. doi: 10.1609/AAAI.V39I12.33378.

[35] José F. Torres, Dalil Hadjout, Abderrazak Sebaa, Francisco Martínez-Álvarez, and Alicia Troncoso. Deep learning for time series forecasting: A survey. *Big Data*, 9(1):3–21, 2021. doi: 10.1089/BIG.2020.0159.

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017.

[37] Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y. Zhang, and Jun Zhou. TimeMixer: Decomposable multiscale mixing for time series forecasting. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[38] Xue Wang, Tian Zhou, Qingsong Wen, Jinyang Gao, Bolin Ding, and Rong Jin. CARD: channel aligned robust blend transformer for time series forecasting. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[39] Yulong Wang, Yushuo Liu, Xiaoyi Duan, and Kai Wang. FilterTS: Comprehensive frequency filtering for multivariate time series forecasting. In *Thirty-Ninth AAAI Conference on Artificial Intelligence, Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence, Fifteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2025, Philadelphia, PA, USA, February 25 - March 4, 2025*, pages 21375–21383. AAAI Press, 2025. doi: 10.1609/AAAI.V39I20.35438.

[40] Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. Transformers in time series: A survey. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*, pages 6778–6786. ijcai.org, 2023. doi: 10.24963/IJCAI.2023/759.

[41] Sithumi Wickramasinghe, Bikramjit Das, and Dorien Herremans. Smart timing for mining: A deep learning framework for bitcoin hardware roi prediction. *arXiv preprint arXiv:2512.05402*, 2025.

[42] Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. Etsformer: Exponential smoothing transformers for time-series forecasting. *arXiv preprint arXiv:2202.01381*, 2022.

[43] Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pages 53140–53164. PMLR / OpenReview.net, 2024.

[44] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 22419–22430, 2021.

[45] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. TimesNet: Temporal 2d-variation modeling for general time series analysis. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[46] Xingjian Wu, Xiangfei Qiu, Hanyin Cheng, Zhengyu Li, Jilin Hu, Chenjuan Guo, and Bin Yang. Enhancing time series forecasting through selective representation spaces: A patch perspective. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025.

[47] Yulun Wu, Sravan Kumar Ankireddy, Samuel Sharpe, Nikita Seleznev, Dehao Yuan, Hyeji Kim, and Nam H Nguyen. Dynamic tokenization via reinforcement patching: End-to-end training and zero-shot transfer. *arXiv preprint arXiv:2603.26097*, 2026.

[48] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 11121–11128. AAAI Press, 2023. doi: 10.1609/AAAI.V37I9.26317.

[49] Tianping Zhang, Yizhuo Zhang, Wei Cao, Jiang Bian, Xiaohan Yi, Shun Zheng, and Jian Li. Less is more: Fast multivariate time series forecasting with light sampling-oriented mlp structures. *arXiv preprint arXiv:2207.01186*, 2022.

[50] Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[51] 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 *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 11106–11115. AAAI Press, 2021. doi: 10.1609/AAAI.V35I12.17325.

[52] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 27268–27286. PMLR, 2022.

[53] Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. One fits all: Power general time series analysis by pretrained LM. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

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: We evaluate `EntroPE` on eight forecasting datasets, 14 UEA classification datasets, and five anomaly-detection benchmarks. The matched-architecture boundary-signal comparison in Section 5.4 and the patch statistics in Table 4 test the proposed segmentation criterion.

2. **Limitations**

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

   Answer: [Yes]

   Justification: Section 6 describes settings in which fixed patches match or outperform adaptive patches, along with limitations of channel-independent segmentation. Appendix F.3 quantifies how noise and transfer to another dataset affect boundary locations.

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: Appendix B.5 states and proves both propositions. Proposition 1 bounds expected next-token log loss for a fixed history, and Proposition 2 shows that splitting a patch preserves total entropy and yields at least one child whose mean entropy is at most the parent mean.

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: Appendix B specifies the tokenizer, causal entropy estimator, boundary rules, and task-specific heads and objectives. For the main forecasting comparison, Appendix C.1 gives the training protocol and selected configurations, and Table 12 gives per-horizon scores.

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: The EntroPE repository provides data access instructions, checkpoints, and training and evaluation commands for all eight forecasting datasets.

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: Appendix A.1 lists the forecasting splits, and Appendix B.2 gives the entropy estimator's optimizer and stopping rule. Tables 10 and 11 give the shared settings and the configurations selected on validation data for each dataset and horizon.

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: Table 16 reports sample standard deviations across seeds 2021–2023 after averaging the four horizons within each seed, with a single run on Traffic. Table 17 separately reports per-horizon variability across five seeds for a fixed `EntroPE` configuration, while Table 4 reports variability across 500 test samples per dataset.

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: Appendix C.1 identifies the GPU models, and Appendix E.3 reports downstream training time per epoch on six forecasting datasets. We provide selected inference latency measurements, but do not provide per-experiment memory requirements or an estimate of total GPU-hours.

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: We used existing benchmark data for model development and evaluation. The study involved no participant recruitment or collection of new personal data, and the Broader Impacts discussion addresses the risk that interpreting patch boundaries as operational alarms could be misleading.

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: The Broader Impacts discussion identifies potential benefits for energy forecasting and industrial monitoring. It explains how noisy measurements can trigger false alarms and how independent channel processing can miss faults visible only across sensors.

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: The release includes time-series forecasting code and checkpoints trained for numerical prediction, along with access to existing benchmarks. These assets do not present the high-risk release scenarios addressed by this question.

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: Appendix A describes the benchmark datasets and credits the baseline methods, and the repository acknowledges upstream implementations.

13. **New assets**

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

    Answer: [Yes]

    Justification: The repository README documents the released forecasting code and checkpoints, including environment setup, data retrieval, entropy-model training, and commands for training and evaluating each forecasting dataset.

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, recruitment, and consent?

    Answer: [N/A]

    Justification: We evaluated models on previously released benchmark datasets. We did not crowdsource labels, recruit participants, or give participants instructions.

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 equivalent approvals were obtained?

    Answer: [N/A]

    Justification: This study uses existing benchmark records and involved no new participant recruitment, interaction, or intervention.

16. **Declaration of LLM usage**

    Question: Does the paper describe the use of LLMs if they are used in the research process, writing process, or method?

    Answer: [Yes]

    Justification: The entropy estimator is a small causal transformer trained on quantized time-series observations. The authors describe their use of ChatGPT for manuscript preparation in the Use of Large Language Models disclosure.

Appendix

Sections A and B describe the benchmarks and implementation. Section C presents the forecasting configurations and results. Sections D and E report seed sensitivity and computational cost. Section F examines how the segmentation behaves across samples, thresholds, noise levels, and datasets.

A Datasets and baselines

A.1 Forecasting Datasets

Table 6 summarizes the 8 datasets in the main forecasting comparison. ETTh1 and ETTh2 record transformer temperature and load hourly, while ETTm1 and ETTm2 record them every 15 minutes. Weather contains 21 meteorological variables. Electricity (ECL) records hourly consumption for 321 clients. Solar records generation at 137 photovoltaic plants every 10 minutes, and Traffic records hourly road occupancy at 862 freeway sensors. The additional monthly forecasting experiment is described in Section C.3.

Table 6: Forecasting datasets in the main comparison. Split sizes are listed in training, validation, and test order.

Dataset	Dim	Reported split sizes (train, val., test)	Frequency	Domain
ETTh1	7	(8545, 2881, 2881)	Hourly	Energy Infrastructure
ETTh2	7	(8545, 2881, 2881)	Hourly	Energy Infrastructure
ETTm1	7	(34465, 11521, 11521)	15 min	Energy Infrastructure
ETTm2	7	(34465, 11521, 11521)	15 min	Energy Infrastructure
Weather	21	(36792, 5271, 10540)	10 min	Meteorological Monitoring
Electricity (ECL)	321	(18317, 2633, 5261)	Hourly	Electricity Consumption
Solar	137	(36697, 5257, 10513)	10 min	Solar Power Generation
Traffic	862	(12185, 1757, 3509)	Hourly	Road Occupancy

A.2 Classification Datasets

We evaluate classification on 14 multivariate datasets from the UEA archive, following TSLANet [11]. The collection includes Cricket, EigenWorms, MotorImagery, and StandWalkJump. Dataset descriptions are available at `https://www.timeseriesclassification.com/`.

A.3 Anomaly Detection Datasets

We follow the anomaly detection protocol of TSLANet [11] and use input windows of 100 observations (Table 7). SMD and PSM contain server-monitoring data. MSL and SMAP contain spacecraft telemetry, and SWaT records sensor measurements from a water treatment testbed.

Table 7: Anomaly detection datasets, reported partition sizes, and input window lengths. Partition sizes are listed in training, validation, and test order.

Dataset	Dim	Reported split sizes (train, val., test)	Length	Domain
SMD	38	(566724, 141681, 708420)	100	Server Machine
MSL	55	(44653, 11664, 73729)	100	Spacecraft
SMAP	25	(108146, 27037, 427617)	100	Spacecraft
SWaT	51	(396000, 99000, 449919)	100	Infrastructure
PSM	25	(105984, 26497, 87841)	100	Server Machine

A.4 Forecasting Baselines

Table 8 lists the 13 baselines in the main forecasting comparison and the feature most relevant to each model.

Table 8: Forecasting baselines grouped by architecture.

Model	Category	Key Features / Innovations
TimeMosaic [8]	Transformer	patch granularity based on temporal heterogeneity and motif reuse.
iTransformer [25]	Transformer	Inverted attention across feature and time dimensions.
PatchTST [27]	Transformer	Patch-based embedding with channel-independence.
FEDformer [52]	Transformer (freq.-decomp.)	Fourier/Wavelet enhanced decomposition.
TimesNet [45]	CNN	Temporal 2D variation modeling with convolutional layers.
TSLANet [11]	CNN + Lightweight	Lightweight and pretraining module for enhanced feature representation
TimeMixer [37]	MLP	Multiscale mixing, separation of trend and seasonal signals.
HDMixer [14]	MLP	Hierarchical dependency with extendable patches.
DLinear [48]	Linear/MLP	Channel-independent trend/seasonal decomposition.
Time-FFM [23]	Foundation + Language	Pre-trained foundation model for time series.
TimeBase [15]	Lightweight	Resource-efficient, practical forecasting baseline.
CALF [22]	LLM-aligned / Cross-modal	Cross-modal match, feature regularization, output consistency.
FilterTS [39]	Filter-based	Static global + dynamic cross-variable frequency filtering.

B Entropy Estimation, Patching, and Task Objectives

B.1 Input Normalization and Tokenization

Following Chronos [2], we represent normalized observations by discrete tokens. We compute the feature means and standard deviations for normalization on the training split. Let X_{train} denote the normalized training values used to estimate the range of the tokenizer. With $\epsilon = 0.005$, we set

$$R = \max\left(\left|Q_{\epsilon/2}(X_{\text{train}})\right|, \left|Q_{1-\epsilon/2}(X_{\text{train}})\right|\right).$$

We use uniform bins over $[-R, R]$ and assign out-of-range values to the nearest end bin. For ordered bin edges $b_1, \ldots, b_{B-1}$, writing $b_0 = -\infty$ and $b_B = +\infty$ gives

$$q(x) = j \quad \text{if } b_{j-1} \le x < b_j, \qquad j \in \{1, \ldots, B\}. \tag{8}$$

If c_j denotes a bin center, its representative value is $d(j) = c_j$. In the forecasting model, the quantized tokens supply inputs to the entropy estimator and to the patch encoder. The forecasting head produces continuous values from the learned representations

B.2 Next-Token Training and Predictive Entropy

For each dataset, we train a causal model on sequences of quantized observations $\tau_{1:L}$. Its next-token cross-entropy loss on one sequence is

$$\mathcal{L}_{\text{CE}} = -\sum_{t=1}^{L-1} \log p_\theta(\tau_{t+1} \mid \tau_{\le t}). \tag{9}$$

We use AdamW with a learning rate of 10^{-3}, weight decay of 0.05, gradient clipping at 1.0, and a batch size of 128. Training runs for up to 50 epochs, with early stopping after five epochs without improvement in validation loss. Table 9 gives the forecasting estimator configuration.

For a discrete distribution p, Shannon entropy is $-\sum_v p(v) \log p(v)$. After training, we freeze the model and compute per-position predictive entropy in nats,

$$H_t = -\sum_{v \in \mathcal{V}} p_\theta(v \mid \tau_{\le t}) \log p_\theta(v \mid \tau_{\le t}).$$

A large H_t means that the model assigns probability across several possible next tokens.

B.3 Boundary Rules and Inputs Longer Than One Block

The setting in Table 11 determines which score we threshold. With *Mono.* set to 0, we compare H_t with the α-quantile of entropy values in the sample. With *Mono.* set to 1, we compare the increase $\Delta H_t = H_t - H_{t-1}$ with

$$\gamma_{\text{rel}} = Q_\alpha\left(\{\Delta H_t\}_{t=2}^{L}\right). \tag{10}$$

Table 9: Configuration of the entropy estimator used in forecasting.

Hyperparameter	Value
Number of layers (n_layer)	2
Number of heads (n_head)	4
Embedding dimension (n_embd)	16
Dropout	0.1
Bias	False
Vocabulary size ($vocab_size$)	256
Block size ($block_size$)	96

A score above the selected threshold starts a new patch after that position. The quantile α is dimensionless. The resulting cutoff is measured in nats.

The same frozen estimator can score inputs of different lengths. For inputs longer than its 96-observation block, it processes non-overlapping blocks. In the released forecasting implementation, each sample and channel is right-padded independently to a multiple of the block size, and scores at padded positions are discarded. The estimator is reused across forecasting lookbacks and classification sequence lengths.

B.4 Task Heads and Training Losses

Let $\tilde{H}^{(c)}$ denote the fused representation for channel c. We use a task-specific linear head and train the forecasting, classification, and reconstruction models with the objectives below.

Forecasting. For each channel, the head predicts T future values.

$$\hat{Y}^{(c)} = g_{\text{fc}}(\text{Flatten}(\tilde{H}^{(c)})) \in \mathbb{R}^T, \qquad \mathcal{L}_{\text{forecast}} = \frac{1}{CT}\sum_{c=1}^{C}\sum_{t=1}^{T}(Y_{c,t} - \hat{Y}_{c,t})^2.$$

Classification. The head projects the concatenated channel representations to K class logits,

$$z = g_{\text{cls}}\Big(\text{Flatten}([\tilde{H}^{(1)}, \ldots, \tilde{H}^{(C)}])\Big) \in \mathbb{R}^K, \qquad \mathcal{L}_{\text{class}} = -\log \frac{\exp(z_y)}{\sum_{k=1}^{K}\exp(z_k)},$$

where y is the observed class.

Anomaly Detection. The reconstruction head predicts the L input values of each channel.

$$\hat{X}^{(c)} = g_{\text{rec}}(\text{Flatten}(\tilde{H}^{(c)})) \in \mathbb{R}^L, \qquad \mathcal{L}_{\text{anomaly}} = \frac{1}{CL}\sum_{c=1}^{C}\sum_{t=1}^{L}(X_{c,t} - \hat{X}_{c,t})^2.$$

Downstream Training. We freeze the entropy estimator θ^*_{ent} and optimize the token embeddings, adaptive patch encoder, global Transformer, fusion decoder, and task head. For these parameters Φ, the training objective is

$$\Phi^* = \arg\min_{\Phi} \mathbb{E}_{(X,Y)\sim\mathcal{D}_{\text{task}}}[\mathcal{L}_{\text{task}}(X, Y; \Phi \mid \theta^*_{\text{ent}})].$$

B.5 Information-Theoretic Motivation

Proposition 1 (Conditional entropy and next-token log loss)**.** *For a fixed history $h = \tau_{\le t}$, let $p_\star(\cdot \mid h)$ denote the true next-token distribution. The expected negative log-likelihood of any predictive distribution $q(\cdot \mid h)$ is at least $H(p_\star(\cdot \mid h))$, with equality exactly when $q(\cdot \mid h) = p_\star(\cdot \mid h)$.*

Proof. The expected negative log-likelihood equals $H(p_\star(\cdot \mid h)) + D_{\text{KL}}(p_\star(\cdot \mid h)\|q(\cdot \mid h))$. The divergence is nonnegative and vanishes exactly when the 2 distributions coincide. □

Proposition 2 (Effect of splitting a patch on its mean entropy)**.** *For fixed per-position entropies h_t, let a patch of length n have mean entropy μ. Splitting it into two nonempty patches of lengths n_1, n_2 and means μ_1, μ_2 preserves total entropy. At least one child mean is at most μ. The smaller child mean is strictly below μ if and only if the two child means differ.*

Proof. The sum over positions gives $n\mu = n_1\mu_1 + n_2\mu_2$, where $n = n_1 + n_2$. Thus μ is a weighted average of the two child means and lies strictly between them when they differ. □

Proposition 1 motivates predictive entropy as a score for next-token uncertainty. `EntroPE` estimates this score from a learned distribution. Proposition 2 describes what a split alone implies. The within-patch variance diagnostics and forecasting comparisons evaluate whether the selected boundaries are useful in practice.

Remark (Continuous forecasting outputs)**.** *The patch encoder embeds quantized input tokens. Its forecasting head projects the resulting hidden representations to real-valued predictions, so the output range is independent of the input bin centers.*

C Forecasting Configurations and Results

C.1 Main Comparison and Training Protocol

For the main comparison, we use $L = 96$ and $T \in \{96, 192, 336, 720\}$. Table 10 gives the shared settings. Table 11 gives the configuration selected on the validation set for each dataset and horizon. For these configurations, we report the mean over three training seeds. Experiments used NVIDIA RTX 4090, RTX A5000, GeForce RTX 4090 D, and RTX 6000 Ada-16Q GPUs. We describe the separate seed experiments and historical results in Section D.

For rerun baselines, we use the original authors' configurations, match the training budgets (epochs, early stopping, and timeout), and report the three-seed mean of the task metric. Forecasting uses MSE and MAE, classification uses accuracy, and anomaly detection uses F1.

Table 10: Shared settings for the main forecasting experiment at $L = 96$.

Hyperparameter	**Value**
Input	96
Global transformer layers	2
Patch encoder / fusion decoder layers	1 / 1
Attention heads	4
FFN dimension (d_{ff})	256
Entropy vocabulary size	256
Normalization	RevIN (affine)
LR schedule	One-cycle (pct_start 0.3)

Table 11: Configurations selected for the main forecasting experiment. Small, medium, and large models use local/global dimensions of $(8, 32)$, $(16, 64)$, and $(32, 128)$, respectively. α is the per-sample quantile level. Mono. selects entropy levels (0) or entropy increases (1).

Dataset	T	Size	Batch	LR	α	Dropout	Mono.
ETTh1	96	M	64	10^{-2}	0.95	0.1	0
	192	S	128	10^{-2}	0.75	0.1	1
	336	S	64	10^{-2}	0.85	0.1	0
	720	S	32	10^{-2}	0.85	0.2	0
ETTh2	96	M	32	10^{-2}	0.85	0.3	1
	192	L	128	10^{-2}	0.95	0.2	0
	336	M	128	10^{-2}	0.95	0.3	0
	720	M	128	10^{-4}	0.75	0.3	0
ETTm1	96	S	32	10^{-2}	0.85	0.1	0
	192	S	128	10^{-2}	0.95	0.1	1
	336	S	128	10^{-2}	0.95	0.1	1
	720	S	32	10^{-2}	0.85	0.1	1
ETTm2	96	S	64	10^{-2}	0.75	0.2	1
	192	M	32	10^{-2}	0.75	0.3	1
	336	S	32	10^{-2}	0.75	0.2	0
	720	L	32	10^{-3}	0.95	0.2	0
Weather	96	S	32	10^{-2}	0.95	0.2	1
	192	S	32	10^{-2}	0.85	0.1	0
	336	S	32	10^{-2}	0.95	0.2	1
	720	S	128	10^{-2}	0.85	0.1	0
ECL	96	M	64	10^{-2}	0.95	0.1	0
	192	M	32	10^{-2}	0.85	0.1	0
	336	M	64	10^{-2}	0.95	0.1	0
	720	M	64	10^{-2}	0.95	0.1	0
Traffic	96	L	8	10^{-2}	0.95	0.1	1
	192	L	8	10^{-2}	0.95	0.1	1
	336	L	8	10^{-2}	0.95	0.1	1
	720	L	8	10^{-2}	0.95	0.1	1
Solar	96	M	64	10^{-2}	0.95	0.1	0
	192	M	32	10^{-2}	0.85	0.1	0
	336	M	64	10^{-2}	0.95	0.1	0
	720	M	32	10^{-2}	0.85	0.1	1

C.2 Forecasting Errors by Dataset and Horizon

Table 12 reports the individual horizon results underlying the main forecasting comparison. Within each dataset, the average row gives the mean over the four horizons. The final column averages across the eight datasets.

Table 12: Full long-term forecasting results across all prediction horizons $T \in \{96, 192, 336, 720\}$ and their average (Avg.). Lower is better. **Best** and **second-best** results are highlighted. "All Dataset" is the mean over the eight datasets. Ties are broken by the other metric of the same dataset.

Models	Horizon	ETTh1		ETTh2		ETTm1		ETTm2		Weather		Solar		ECL		Traffic		All Dataset	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
FilterTS	96	0.388	0.396	0.292	0.341	0.325	0.362	**0.174**	**0.256**	0.173	0.217	0.230	0.272	0.174	0.261	0.466	0.319	0.278	0.303
	192	0.436	**0.423**	0.378	0.392	0.367	0.384	**0.240**	**0.299**	0.223	0.260	0.268	0.296	0.184	0.270	0.472	0.318	0.321	0.330
	336	0.479	0.446	0.423	0.429	0.400	0.405	**0.303**	**0.340**	0.279	0.300	0.298	0.314	0.199	0.285	0.488	0.325	0.359	0.356
	720	0.482	0.472	0.429	0.444	0.464	0.441	0.409	0.400	0.356	0.350	0.299	0.314	0.241	0.319	0.520	0.343	0.400	0.385
	Avg.	0.446	0.434	0.381	0.402	0.389	0.398	**0.282**	**0.324**	0.258	0.282	0.274	0.299	0.200	0.284	0.487	0.326	0.339	0.344
HDMixer	96	0.388	0.398	0.293	0.343	0.336	0.368	0.182	0.265	0.174	0.224	0.256	0.319	0.175	0.279	0.525	0.373	0.291	0.321
	192	0.443	0.429	0.379	0.395	0.375	0.386	0.246	0.306	0.225	0.264	0.322	0.328	0.187	0.291	0.519	0.375	0.337	0.347
	336	0.483	0.449	0.420	0.431	0.406	0.407	0.305	0.343	0.274	0.302	0.336	0.350	0.198	0.303	0.549	0.381	0.371	0.371
	720	0.495	0.477	0.438	0.450	0.474	0.445	0.402	0.400	0.351	0.348	0.336	0.346	0.236	0.330	0.579	0.387	0.414	0.398
	Avg.	0.452	0.438	0.383	0.405	0.398	0.402	0.284	0.329	0.256	0.285	0.313	0.336	0.199	0.301	0.543	0.379	0.353	0.359
TimeBase	96	0.394	0.402	0.301	0.348	0.386	0.408	0.195	0.278	0.173	0.220	0.355	0.389	0.222	0.315	0.767	0.404	0.349	0.346
	192	0.442	0.429	0.396	0.409	0.410	0.413	0.253	0.313	0.220	0.262	0.395	0.413	0.229	0.323	0.696	0.381	0.380	0.368
	336	0.489	0.457	0.433	0.441	0.452	0.444	0.315	0.351	0.274	0.302	0.431	0.427	0.247	0.339	0.705	0.383	0.418	0.393
	720	0.475	0.465	0.444	0.460	0.503	0.464	0.418	0.409	0.356	0.356	0.428	0.416	0.290	0.371	0.746	0.401	0.458	0.418
	Avg.	0.450	0.438	0.394	0.415	0.438	0.432	0.295	0.338	0.256	0.285	0.402	0.411	0.247	0.337	0.729	0.392	0.401	0.381
TimeMosaic	96	**0.375**	**0.387**	**0.291**	**0.332**	**0.311**	**0.339**	**0.177**	**0.252**	0.175	**0.208**	0.231	**0.246**	0.176	0.253	0.456	0.279	0.274	**0.287**
	192	0.424	**0.416**	**0.368**	**0.383**	0.368	**0.369**	0.245	**0.296**	0.222	**0.250**	0.268	0.272	0.182	0.261	0.469	0.282	0.318	**0.316**
	336	0.473	**0.440**	0.420	**0.420**	0.413	**0.397**	0.306	**0.336**	0.277	**0.291**	0.293	0.281	0.199	0.278	0.488	0.291	0.359	**0.342**
	720	**0.474**	**0.461**	**0.419**	**0.436**	0.465	**0.430**	**0.402**	**0.393**	0.354	**0.341**	0.281	**0.275**	0.239	0.310	0.525	**0.307**	**0.395**	**0.369**
	Avg.	**0.437**	**0.426**	**0.375**	**0.393**	0.389	**0.384**	0.283	**0.319**	0.257	**0.273**	0.268	0.269	0.199	0.276	0.485	0.290	0.336	**0.329**
TSLANet	96	0.380	0.402	0.299	0.350	0.322	0.360	0.178	0.262	0.174	0.215	0.232	0.267	0.177	0.262	0.470	0.300	0.279	0.302
	192	0.437	0.434	0.380	0.401	**0.361**	0.382	0.249	0.308	0.221	0.256	0.255	0.280	0.183	0.268	0.477	0.300	0.320	0.329
	336	0.488	0.456	0.418	0.426	0.394	0.406	0.310	0.347	0.276	0.297	0.281	0.294	0.200	0.286	0.488	0.305	0.357	0.352
	720	0.550	0.504	0.427	0.445	**0.456**	0.442	0.408	0.404	0.352	0.346	0.284	0.293	0.242	0.320	0.519	0.323	0.405	0.385
	Avg.	0.464	0.449	0.381	0.406	**0.383**	0.398	0.286	0.330	0.256	0.279	0.263	0.284	0.201	0.284	0.489	0.307	0.340	0.342
CALF	96	0.378	**0.394**	0.297	0.342	0.327	0.355	0.180	0.256	0.175	0.211	**0.202**	**0.220**	0.161	**0.243**	**0.431**	**0.268**	**0.269**	**0.286**
	192	0.430	0.424	0.376	**0.391**	0.376	0.381	0.246	0.300	**0.220**	0.254	**0.232**	**0.238**	0.176	**0.257**	**0.453**	**0.277**	**0.314**	**0.315**
	336	0.480	0.450	0.418	0.431	0.425	0.407	0.307	0.341	0.277	0.294	0.262	**0.257**	0.194	**0.275**	**0.476**	**0.287**	**0.355**	**0.343**
	720	0.484	0.472	0.424	0.442	0.483	0.445	0.406	0.398	0.355	0.347	0.271	**0.257**	0.235	**0.309**	**0.507**	**0.306**	0.396	**0.372**
	Avg.	0.443	0.435	0.379	0.402	0.403	0.397	0.285	0.324	0.257	0.277	**0.242**	**0.243**	0.192	**0.271**	**0.467**	**0.285**	**0.333**	**0.329**
DLinear	96	0.398	0.412	0.351	0.403	0.346	0.374	0.195	0.295	0.197	0.259	0.287	0.374	0.196	0.281	0.652	0.400	0.328	0.350
	192	0.446	0.441	0.482	0.479	0.382	0.391	0.285	0.361	0.239	0.299	0.317	0.395	0.196	0.284	0.601	0.375	0.369	0.378
	336	0.487	0.465	0.602	0.546	0.415	0.415	0.385	0.429	0.282	0.332	0.351	0.413	0.209	0.300	0.608	0.377	0.417	0.410
	720	0.516	0.513	0.840	0.661	0.474	0.451	0.556	0.523	0.347	0.384	0.356	0.413	0.244	0.332	0.648	0.398	0.498	0.459
	Avg.	0.462	0.458	0.569	0.522	0.404	0.408	0.355	0.402	0.266	0.319	0.328	0.399	0.211	0.299	0.627	0.388	0.403	0.399
FEDformer	96	**0.377**	0.419	0.351	0.392	0.385	0.422	0.195	0.284	0.222	0.304	0.236	0.355	0.194	0.311	0.566	0.348	0.316	0.354
	192	**0.421**	0.445	0.442	0.450	0.441	0.448	0.266	0.326	0.299	0.360	0.255	0.359	0.212	0.328	0.607	0.381	0.368	0.387
	336	**0.469**	0.473	0.499	0.490	0.466	0.470	0.331	0.368	0.339	0.386	**0.247**	0.355	0.219	0.335	0.609	0.370	0.397	0.406
	720	0.504	0.503	0.493	0.494	0.529	0.498	0.438	0.433	0.421	0.435	0.278	0.383	0.247	0.356	0.643	0.397	0.444	0.437
	Avg.	0.443	0.460	0.446	0.457	0.455	0.460	0.308	0.353	0.320	0.371	0.254	0.363	0.218	0.333	0.606	0.374	0.381	0.396
PatchTST	96	0.383	0.405	0.314	0.358	0.336	0.374	0.183	0.266	0.174	0.216	0.218	0.275	0.176	0.270	0.454	0.293	0.280	0.307
	192	0.447	0.440	0.391	0.407	0.372	0.394	0.255	0.315	0.220	0.256	0.250	0.295	0.184	0.277	0.461	0.296	0.323	0.335
	336	0.490	0.472	0.431	0.441	0.410	0.417	0.321	0.356	0.276	0.296	0.274	0.308	0.201	0.294	0.477	0.303	0.360	0.361
	720	0.525	0.503	0.442	0.457	0.465	0.452	0.416	0.414	0.353	0.346	0.267	0.299	0.241	0.326	0.510	0.322	0.402	0.390
	Avg.	0.461	0.455	0.395	0.416	0.396	0.409	0.294	0.338	0.256	0.279	0.252	0.294	0.201	0.292	0.476	0.304	0.341	0.348
TimeMixer	96	0.394	0.402	0.295	0.344	0.327	0.364	0.179	0.261	**0.163**	**0.210**	0.206	0.257	**0.157**	0.248	0.472	0.301	0.274	0.298
	192	0.453	0.439	0.390	0.403	0.374	0.386	**0.242**	0.302	0.225	0.266	**0.233**	0.275	**0.171**	0.261	0.481	0.304	0.321	0.330
	336	0.498	0.455	0.437	0.443	**0.392**	0.406	**0.303**	0.342	**0.264**	**0.291**	0.258	0.297	**0.190**	0.282	0.504	0.309	0.356	0.353
	720	0.585	0.521	0.466	0.460	0.464	0.447	**0.402**	**0.397**	**0.346**	0.346	0.301	0.317	**0.227**	0.312	0.521	0.327	0.414	0.391
	Avg.	0.483	0.454	0.397	0.413	0.389	0.401	**0.282**	0.326	**0.250**	0.278	0.250	0.287	**0.186**	0.276	0.495	0.310	0.341	0.343
iTransformer	96	0.391	0.406	0.305	0.355	0.346	0.377	0.184	0.266	0.177	0.216	0.218	0.247	**0.154**	**0.244**	**0.407**	**0.273**	**0.273**	0.298
	192	0.448	0.441	0.381	0.404	0.398	0.405	0.252	0.310	0.224	0.257	0.239	**0.266**	**0.167**	**0.257**	**0.431**	**0.282**	0.318	0.328
	336	0.485	0.456	0.436	0.436	0.439	0.429	0.316	0.353	0.282	0.299	0.256	**0.279**	**0.184**	**0.275**	**0.449**	**0.291**	0.356	0.352
	720	0.504	0.491	0.462	0.461	0.501	0.466	0.419	0.411	0.359	0.351	**0.261**	0.281	**0.221**	**0.307**	**0.481**	0.311	0.401	0.385
	Avg.	0.457	0.449	0.396	0.414	0.421	0.419	0.293	0.335	0.261	0.281	0.244	**0.268**	**0.182**	**0.271**	**0.442**	**0.289**	0.337	0.341
TimesNet	96	0.415	0.431	0.338	0.379	0.336	0.377	0.188	0.268	0.178	0.226	0.297	0.328	0.208	0.310	0.628	0.350	0.324	0.334
	192	0.464	0.456	0.406	0.414	0.388	0.402	0.256	0.310	0.225	0.265	0.381	0.377	0.226	0.324	0.639	0.350	0.373	0.362
	336	0.498	0.472	0.471	0.468	0.417	0.425	0.320	0.349	0.287	0.308	0.341	0.373	0.268	0.350	0.655	0.359	0.407	0.388
	720	0.529	0.502	0.441	0.452	0.473	0.452	0.429	0.411	0.357	0.353	0.360	0.381	0.302	0.373	0.687	0.372	0.447	0.412
	Avg.	0.477	0.465	0.414	0.428	0.404	0.414	0.298	0.335	0.262	0.288	0.345	0.365	0.251	0.339	0.652	0.358	0.388	0.374
TimeFFM	96	0.400	0.403	**0.290**	**0.341**	0.338	0.362	0.191	0.276	0.195	0.237	0.328	0.370	0.200	0.283	0.691	0.440	0.329	0.339
	192	0.457	0.436	**0.371**	0.393	0.380	0.383	0.247	0.307	0.240	0.272	0.368	0.381	0.212	0.304	0.647	0.424	0.365	0.363
	336	0.497	0.453	**0.414**	0.428	0.413	0.404	0.313	0.351	0.292	0.307	0.401	0.396	0.225	0.314	0.651	0.425	0.401	0.385
	720	0.483	0.463	0.425	0.444	0.477	0.438	0.409	0.405	0.364	0.352	0.401	0.397	0.258	0.337	0.687	0.440	0.438	0.410
	Avg.	0.459	0.439	0.375	0.402	0.402	0.397	0.290	0.335	0.273	0.292	0.375	0.386	0.224	0.310	0.669	0.432	0.383	0.374
EntroPE	96	0.379	0.396	0.295	0.348	**0.315**	**0.355**	0.180	0.267	**0.168**	0.216	**0.206**	0.254	0.169	0.262	0.485	0.288	0.275	0.298
	192	**0.422**	0.425	0.378	0.399	**0.362**	**0.381**	0.246	0.306	**0.212**	**0.254**	0.240	0.276	0.178	0.269	0.488	0.295	**0.316**	0.326
	336	**0.459**	**0.446**	**0.387**	**0.420**	**0.388**	**0.402**	0.309	0.348	**0.267**	0.295	**0.255**	0.286	0.194	0.284	0.494	0.297	**0.344**	0.347
	720	**0.446**	**0.458**	**0.414**	**0.439**	**0.449**	**0.436**	0.406	0.406	**0.341**	**0.342**	**0.255**	0.283	0.232	0.320	0.525	0.311	**0.384**	0.374
	Avg.	**0.427**	**0.431**	**0.369**	**0.402**	**0.379**	**0.394**	0.285	0.332	**0.247**	**0.277**	**0.239**	0.275	0.193	0.284	0.498	0.298	**0.330**	0.336

C.3 Longer Contexts and Forecast Horizons

We compare dynamic and fixed patching at small, medium, and large model sizes. Within each comparison, the training budget is held fixed and the static patch length matches the average dynamic patch length. The static comparator follows the PatchTST architecture. TimeMosaic provides an additional adaptive-patching baseline. The ETT, Weather, and Solar comparisons use $L = 512$. M4-Monthly uses $L = 36$ and $T = 18$.

Table 13 gives absolute errors, and Table 14 expresses each paired difference relative to the static model. At $512 \to 96$, dynamic patching has lower MSE for all five datasets and all three sizes. Gains are mixed for M4-Monthly and at longer horizons. On Solar, static patching has lower MSE for the small model at $T = 480$ and $T = 720$ and for the large model at $T = 720$.

Table 13: Forecasting results for short-, medium-, and long-horizon settings. In each S/M/L column, entries are **Dynamic/Static** for the indicated metric. The final column instead reports **MSE/MAE** for TimeMosaic. Lower error is better. S/M/L denote model size, while L in $L \to T$ denotes input length.

Dataset	L→T	S-MSE	S-MAE	M-MSE	M-MAE	L-MSE	L-MAE	TimeMosaic (MSE/MAE)
M4-Monthly	36→18	.200/.199	.195/.197	.196/.199	.193/.196	.196/.195	.192/.192	.198/.194
ETTh1	512→96	.392/.420	.417/.437	.408/.422	.427/.438	.387/.443	.410/.452	.389/.418
ETTh2	512→96	.312/.318	.371/.380	.294/.324	.357/.384	.322/.334	.375/.380	.314/.318
ETTm1	512→96	.310/.326	.357/.370	.324/.337	.364/.381	.305/.313	.353/.361	.320/.362
ETTm2	512→96	.182/.188	.270/.276	.178/.194	.271/.277	.179/.194	.267/.279	.179/.279
Weather	512→96	.159/.172	.215/.230	.158/.166	.214/.221	.159/.163	.212/.217	.156/.220
ETTh1	512→480	.451/.485	.463/.485	.434/.474	.457/.484	.432/.463	.452/.478	.435/.455
ETTh2	512→480	.380/.380	.431/.432	.367/.386	.416/.432	.399/.393	.442/.437	.361/.415
ETTm1	512→480	.394/.412	.407/.417	.430/.424	.430/.423	.397/.423	.406/.425	.391/.419
ETTm2	512→480	.349/.349	.380/.383	.343/.360	.381/.387	.331/.346	.367/.384	.348/.380
Solar	512→480	.260/.245	.331/.315	.222/.250	.300/.321	.233/.233	.315/.318	.252/.300
Weather	512→480	.286/.290	.312/.313	.291/.290	.316/.316	.286/.288	.309/.311	.288/.316
Solar	512→720	.255/.250	.320/.319	.229/.249	.301/.323	.243/.237	.318/.318	.256/.319

Table 14: Relative error reduction (%) for dynamic versus static patching in Table 13, defined as $100(e_{\text{static}} - e_{\text{dynamic}})/e_{\text{static}}$. Positive values favor dynamic patching; negative values favor static patching.

Dataset	L→T	S-MSE	S-MAE	M-MSE	M-MAE	L-MSE	L-MAE
M4-Monthly	36→18	-0.313	+0.770	+1.336	+1.317	-0.465	-0.346
ETTh1	512→96	+6.845	+4.467	+3.201	+2.490	+12.654	+9.340
ETTh2	512→96	+1.945	+2.530	+9.462	+6.998	+3.591	+1.264
ETTm1	512→96	+5.074	+3.594	+3.783	+4.561	+2.432	+2.164
ETTm2	512→96	+3.276	+2.272	+8.108	+1.976	+7.700	+4.310
Weather	512→96	+7.360	+6.868	+4.876	+2.968	+2.086	+2.229
ETTh1	512→480	+6.958	+4.416	+8.394	+5.682	+6.789	+5.440
ETTh2	512→480	-0.027	+0.407	+4.889	+3.836	-1.514	-1.240
ETTm1	512→480	+4.418	+2.576	-1.303	-1.489	+6.254	+4.508
ETTm2	512→480	-0.233	+0.909	+4.892	+1.439	+4.353	+4.494
Solar	512→480	-6.053	-5.012	+11.155	+6.473	+0.105	+0.743
Weather	512→480	+1.423	+0.258	-0.183	+0.046	+0.733	+0.523
Solar	512→720	-2.038	-0.206	+8.130	+6.732	-2.336	+0.048

C.4 PatchTST Integration Experiment

We replace PatchTST's static patching with EntroPE's dynamic patcher and fusion decoder while keeping the remaining components fixed. Table 15 reports this six-dataset integration experiment separately from the eight-dataset main comparison. Both changed components contribute to the comparison. The combined variant has lower MSE on all six datasets. On ETTm2, MAE changes from 0.334 to 0.335.

Table 15: PatchTST integration experiment on six datasets. The second row uses EntroPE's dynamic patcher and fusion decoder. Relative reductions use PatchTST as the reference and are rounded to whole percentages.

Model	**ETTh1**		**ETTh2**		**ETTm1**		**ETTm2**		**Weather**		**ECL**	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
PatchTST	0.516	0.484	0.391	0.411	0.406	0.407	0.290	0.334	0.265	0.285	0.216	0.318
EntroPE	0.416	0.425	0.366	0.387	0.378	0.391	0.286	0.335	0.242	0.273	0.182	0.271
Δ% Improvement	19%	12%	6%	6%	7%	4%	1%	0%	9%	4%	16%	15%

D Forecasting Variability Across Seeds

Table 16 compares EntroPE with four baselines in the main forecasting evaluation. It reports horizon-averaged errors across three seeds on seven datasets and single-seed results on Traffic. We then examine EntroPE across five seeds in two settings. Table 17 reports per-horizon variability under a fixed configuration. Finally, Table 18 reproduces the earlier six-dataset comparison, in which we evaluated EntroPE using best-seed reporting.

D.1 Seed Variability in the Main Forecasting Comparison

Table 16 compares EntroPE with four baselines from the main forecasting benchmark. Across the seven datasets evaluated with three seeds, EntroPE's horizon-averaged MSE standard deviation ranges from 0.001 on ETTm2, Weather, and Electricity to 0.005 on ETTh2. Among the five methods shown, EntroPE has the lowest mean MSE on ETTh1, ETTh2, ETTm1, Weather, and Solar. TimeMosaic has the lowest mean MSE on ETTm2, and iTransformer on Electricity.

Table 16: Horizon-averaged test MSE and MAE at input length $L = 96$. For each seed, we average each metric over $T \in \{96, 192, 336, 720\}$. Entries give the mean $\pm$ sample standard deviation across seeds 2021, 2022, and 2023. Traffic entries give the single-seed score. A displayed standard deviation of 0.000 is below 0.0005.

Dataset	**Metric**	**TimeMosaic**	**CALF**	**iTransformer**	**HDMixer**	**EntroPE**
ETTh1	MSE	0.437 ± 0.004	0.443 ± 0.002	0.457 ± 0.003	0.452 ± 0.003	0.427 ± 0.004
	MAE	0.426 ± 0.003	0.435 ± 0.001	0.449 ± 0.002	0.438 ± 0.003	0.431 ± 0.002
ETTh2	MSE	0.375 ± 0.002	0.379 ± 0.003	0.396 ± 0.005	0.383 ± 0.005	0.369 ± 0.005
	MAE	0.393 ± 0.001	0.402 ± 0.001	0.414 ± 0.003	0.405 ± 0.001	0.402 ± 0.003
ETTm1	MSE	0.389 ± 0.003	0.403 ± 0.001	0.421 ± 0.004	0.398 ± 0.002	0.379 ± 0.003
	MAE	0.384 ± 0.001	0.397 ± 0.000	0.419 ± 0.003	0.402 ± 0.001	0.394 ± 0.001
ETTm2	MSE	0.283 ± 0.001	0.285 ± 0.001	0.293 ± 0.002	0.284 ± 0.002	0.285 ± 0.001
	MAE	0.319 ± 0.000	0.324 ± 0.000	0.335 ± 0.002	0.329 ± 0.002	0.332 ± 0.000
Weather	MSE	0.257 ± 0.000	0.257 ± 0.000	0.261 ± 0.001	0.256 ± 0.000	0.247 ± 0.001
	MAE	0.273 ± 0.001	0.277 ± 0.000	0.281 ± 0.001	0.285 ± 0.000	0.277 ± 0.000
Solar	MSE	0.268 ± 0.001	0.242 ± 0.001	0.244 ± 0.002	0.313 ± 0.011	0.239 ± 0.003
	MAE	0.269 ± 0.001	0.243 ± 0.001	0.268 ± 0.000	0.336 ± 0.001	0.275 ± 0.001
ECL	MSE	0.199 ± 0.000	0.192 ± 0.001	0.182 ± 0.000	0.199 ± 0.001	0.193 ± 0.001
	MAE	0.276 ± 0.001	0.271 ± 0.001	0.271 ± 0.000	0.301 ± 0.001	0.284 ± 0.001
Traffic	MSE	0.485	0.467	0.442	0.543	0.498
	MAE	0.290	0.285	0.289	0.379	0.298

D.2 Per-Horizon Seed Variability Under a Fixed Configuration

To measure how forecasting error varies with the training seed, we evaluate EntroPE on five datasets using seeds 2021–2025. Across datasets and horizons, we fix the model width ($d_{\text{model}} = 16$), depth (two layers), number of attention heads (four), entropy quantile ($\alpha = 0.75$), and learning rate (0.001). Table 17 reports the mean and standard deviation separately for each forecast horizon. The largest MSE standard deviations occur at $T = 720$ on ETTh1 (0.013), ETTh2 (0.012), and ETTm2 (0.010). On Weather, the MSE standard deviation is at most 0.001 at every horizon.

Table 17: Five-seed robustness under a shared hyperparameter configuration. Each entry is the mean $\pm$ standard deviation across seeds 2021–2025. Errors and standard deviations are rounded to three decimals; a reported standard deviation of 0.000 reflects this rounding.

Dataset	Horizon T	MSE	MAE
ETTh1	96	0.385 ± 0.004	0.407 ± 0.002
	192	0.430 ± 0.001	0.433 ± 0.001
	336	0.460 ± 0.004	0.446 ± 0.003
	720	0.468 ± 0.013	0.469 ± 0.007
ETTh2	96	0.333 ± 0.005	0.376 ± 0.004
	192	0.409 ± 0.007	0.421 ± 0.005
	336	0.405 ± 0.004	0.432 ± 0.004
	720	0.442 ± 0.012	0.460 ± 0.013
ETTm1	96	0.344 ± 0.006	0.373 ± 0.004
	192	0.384 ± 0.003	0.392 ± 0.001
	336	0.413 ± 0.007	0.415 ± 0.003
	720	0.473 ± 0.006	0.450 ± 0.003
ETTm2	96	0.189 ± 0.003	0.275 ± 0.002
	192	0.265 ± 0.005	0.322 ± 0.002
	336	0.336 ± 0.007	0.364 ± 0.003
	720	0.432 ± 0.010	0.421 ± 0.005
Weather	96	0.172 ± 0.001	0.218 ± 0.001
	192	0.217 ± 0.000	0.258 ± 0.001
	336	0.273 ± 0.001	0.297 ± 0.001
	720	0.346 ± 0.001	0.346 ± 0.001

D.3 Six-Dataset Comparison Under Best-Seed Reporting

Table 18 reproduces the earlier six-dataset comparison, reporting EntroPE using its best-seed results. We used an NVIDIA RTX 6000 Ada GPU for the EntroPE runs and took baseline results reported through 2024 from Huang et al. [16]. The main eight-dataset evaluation follows the protocol in Section C.1, with per-horizon results in Table 12.

Table 18: Best-seed forecasting performance (MSE and MAE) of **EntroPE** and 15 baselines on six datasets. The look-back window is fixed at $L = 96$ with forecast horizons $T = \{96, 192, 336, 720\}$. **Red** marks the best result and **Blue** the second best. Avg. is the mean of the four horizons. Experiments are conducted on an NVIDIA RTX Ada 6000 GPU. We extract baseline results up to 2024 from Huang et al. [16].

Dataset	T	**EntroPE**		TimeMosaic		HDMixer		TimeBase		CALF		FilterTS		LangTime [28]		TimeMixer		Time-FFM		TSLANet		iTrans.		PatchTST		TimesNet		DLinear		FEDformer		Autoformer	
		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	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	96	**0.317**	0.354	**0.312**	**0.351**	0.340	0.369	0.373	0.388	0.319	**0.348**	0.323	0.362	0.319	**0.348**	**0.317**	0.356	0.336	0.369	0.321	0.362	0.334	0.368	0.352	0.374	0.338	0.375	0.346	0.374	0.379	0.419	0.505	0.475
	192	**0.360**	0.379	0.370	0.379	0.379	0.385	0.411	0.409	0.375	**0.376**	0.364	0.382	0.368	**0.375**	0.367	0.384	0.378	0.389	**0.361**	0.383	0.377	0.391	0.390	0.393	0.374	0.387	0.382	0.391	0.426	0.441	0.553	0.496
	336	0.388	**0.399**	**0.386**	**0.399**	0.398	0.409	0.436	0.421	0.410	**0.399**	0.396	0.404	0.413	**0.402**	0.391	0.406	0.411	0.410	**0.382**	0.404	0.426	0.420	0.421	0.414	0.410	0.411	0.415	0.415	0.445	0.459	0.621	0.537
	720	**0.446**	**0.433**	0.469	0.440	0.467	0.443	0.503	0.461	0.478	0.439	0.462	0.441	0.487	0.439	**0.454**	0.441	0.469	0.441	**0.446**	**0.438**	0.491	0.459	0.462	0.449	0.478	0.450	0.473	0.451	0.543	0.490	0.671	0.561
	Avg.	**0.378**	**0.391**	0.384	**0.392**	0.396	0.402	0.431	0.420	0.396	**0.391**	0.386	0.397	0.397	**0.391**	**0.382**	0.397	0.399	0.402	**0.378**	0.397	0.407	0.410	0.406	0.408	0.400	0.406	0.404	0.408	0.448	0.452	0.588	0.517
ETTm2	96	0.180	0.265	0.177	**0.254**	0.183	0.266	0.188	0.271	**0.174**	**0.252**	**0.174**	0.256	0.188	0.258	**0.175**	0.257	0.181	0.267	0.179	0.261	0.180	0.264	0.183	0.270	0.187	0.267	0.193	0.293	0.203	0.287	0.255	0.339
	192	**0.241**	0.311	**0.240**	**0.295**	0.246	0.306	0.251	0.309	**0.241**	**0.297**	**0.240**	0.299	0.245	**0.297**	**0.240**	0.302	0.247	0.308	0.243	0.303	0.250	0.309	0.255	0.314	0.249	0.309	0.284	0.361	0.269	0.328	0.281	0.340
	336	0.313	0.352	**0.298**	**0.336**	0.307	0.347	0.311	0.346	0.304	**0.338**	**0.301**	0.339	**0.301**	**0.336**	0.303	0.343	0.309	0.347	0.308	0.345	0.311	0.348	0.309	0.347	0.321	0.351	0.382	0.429	0.325	0.366	0.339	0.372
	720	0.411	0.413	**0.397**	**0.391**	0.408	0.404	0.411	0.401	0.402	0.395	0.399	0.399	0.402	**0.393**	**0.392**	0.396	0.406	0.404	0.403	0.400	0.412	0.407	0.412	0.404	0.408	0.403	0.558	0.525	0.421	0.415	0.433	0.432
	Avg.	0.286	0.335	**0.278**	**0.319**	0.286	0.331	0.290	0.332	0.280	**0.321**	**0.279**	0.323	0.284	**0.321**	**0.278**	0.325	0.286	0.332	0.283	0.327	0.288	0.332	0.290	0.334	0.291	0.333	0.354	0.402	0.305	0.349	0.327	0.371
ETTh1	96	**0.375**	0.399	**0.369**	0.397	0.387	0.401	0.399	0.392	0.377	0.395	0.377	**0.391**	0.391	**0.388**	0.385	0.402	0.385	0.400	0.387	0.405	0.386	0.405	0.460	0.447	0.384	0.402	0.397	0.412	0.395	0.424	0.449	0.459
	192	**0.423**	0.425	**0.419**	0.439	0.441	0.428	0.455	**0.423**	0.429	0.427	0.431	**0.423**	0.429	**0.419**	0.443	0.430	0.439	0.430	0.448	0.436	0.441	0.436	0.512	0.477	0.436	0.429	0.446	0.441	0.469	0.470	0.500	0.482
	336	**0.429**	**0.432**	0.454	0.451	0.452	**0.433**	0.501	0.443	0.475	0.449	0.479	0.448	0.462	0.440	0.512	0.470	0.480	0.449	**0.451**	0.437	0.487	0.458	0.546	0.496	0.638	0.469	0.489	0.467	0.490	0.477	0.521	0.496
	720	**0.439**	**0.454**	0.466	0.479	0.513	0.485	0.498	0.458	0.482	0.470	0.471	0.466	**0.458**	**0.445**	0.497	0.476	0.462	0.456	0.505	0.485	0.503	0.491	0.544	0.517	0.521	0.500	0.513	0.510	0.598	0.544	0.514	0.512
	Avg.	**0.417**	**0.428**	**0.427**	0.442	0.448	0.437	0.463	0.429	0.441	0.435	0.440	0.432	0.435	**0.423**	0.459	0.445	0.442	0.434	0.448	0.441	0.454	0.448	0.516	0.484	0.495	0.450	0.461	0.458	0.488	0.479	0.496	0.487
ETTh2	96	**0.281**	**0.336**	0.292	**0.341**	0.289	**0.336**	0.338	0.376	**0.288**	**0.336**	0.293	0.344	0.299	**0.336**	0.289	0.342	0.301	0.351	0.299	0.354	0.297	0.349	0.308	0.355	0.340	0.374	0.340	0.394	0.358	0.397	0.346	0.388
	192	0.371	0.393	**0.367**	0.391	0.386	0.397	0.402	0.405	**0.368**	**0.386**	0.374	0.389	0.374	**0.382**	0.378	0.397	0.378	0.397	0.382	0.399	0.380	0.400	0.393	0.405	0.402	0.414	0.482	0.479	0.429	0.439	0.456	0.452
	336	**0.392**	**0.394**	0.417	0.427	0.438	0.442	0.437	0.440	0.415	0.423	0.411	0.423	0.410	0.418	0.432	0.434	0.422	0.431	**0.379**	**0.396**	0.428	0.432	0.427	0.436	0.452	0.452	0.591	0.541	0.496	0.487	0.482	0.486
	720	0.421	**0.427**	**0.398**	0.436	0.421	0.454	0.460	0.477	**0.417**	0.435	0.423	0.441	0.418	**0.426**	0.464	0.464	0.427	0.444	0.429	0.445	0.427	0.445	0.436	0.450	0.462	0.468	0.839	0.661	0.463	0.474	0.515	0.511
	Avg.	**0.366**	**0.388**	**0.369**	0.399	0.384	0.407	0.409	0.425	0.372	0.395	0.375	0.399	0.375	**0.391**	0.391	0.409	0.382	0.406	0.372	0.399	0.383	0.407	0.391	0.412	0.414	0.427	0.563	0.519	0.437	0.449	0.450	0.459
Weather	96	0.164	0.211	0.166	**0.201**	0.175	0.223	0.170	0.215	0.164	0.205	**0.161**	0.208	0.178	**0.204**	**0.163**	0.209	0.191	0.230	0.177	0.217	0.174	0.214	0.186	0.227	0.172	0.220	0.195	0.252	0.217	0.296	0.266	0.336
	192	**0.210**	**0.252**	0.219	0.258	0.226	0.265	0.216	0.256	0.213	**0.252**	0.226	0.264	**0.211**	**0.249**	**0.211**	0.254	0.236	0.267	0.224	0.257	0.221	0.254	0.234	0.265	0.219	0.261	0.237	0.295	0.276	0.336	0.307	0.367
	336	**0.256**	**0.290**	0.273	0.296	0.264	0.301	0.272	0.297	0.270	**0.292**	0.279	0.304	0.269	**0.292**	0.263	0.293	0.289	0.303	0.278	0.297	0.278	0.296	0.284	0.301	**0.246**	0.337	0.282	0.331	0.339	0.380	0.359	0.395
	720	**0.339**	**0.342**	0.354	0.351	0.348	0.349	0.351	0.348	0.352	0.346	0.345	**0.344**	0.351	0.347	**0.344**	0.348	0.362	0.350	0.355	0.347	0.358	0.347	0.356	0.349	0.365	0.359	0.345	0.382	0.403	0.428	0.419	0.428
	Avg.	**0.242**	**0.274**	0.253	0.277	0.253	0.285	0.252	0.279	0.250	**0.274**	0.253	0.280	0.252	**0.273**	**0.245**	0.276	0.270	0.288	0.259	0.280	0.258	0.278	0.265	0.286	0.251	0.294	0.265	0.315	0.309	0.360	0.338	0.382
Electricity	96	0.163	0.252	0.161	0.261	0.180	0.271	0.212	0.279	**0.145**	**0.239**	0.153	0.247	0.181	0.266	0.153	0.245	0.198	0.282	0.175	0.260	**0.148**	**0.240**	0.190	0.296	0.168	0.272	0.210	0.302	0.193	0.308	0.201	0.317
	192	0.177	0.268	0.176	0.274	0.184	0.275	0.209	0.281	**0.162**	**0.253**	0.168	0.260	0.185	0.273	**0.166**	**0.257**	0.199	0.285	0.182	0.268	**0.162**	**0.253**	0.199	0.304	0.184	0.322	0.210	0.305	0.201	0.315	0.222	0.334
	336	0.194	0.284	0.189	0.283	0.207	0.299	0.222	0.295	**0.177**	**0.268**	0.187	0.278	0.198	0.281	0.185	0.275	0.212	0.298	0.199	0.285	**0.178**	**0.269**	0.217	0.319	0.198	0.300	0.223	0.319	0.214	0.329	0.231	0.338
	720	0.235	0.321	0.225	**0.304**	0.249	0.335	0.264	0.327	**0.222**	**0.304**	0.227	0.313	0.241	0.320	0.224	**0.312**	0.253	0.330	0.240	0.317	0.225	0.317	0.258	0.352	**0.220**	0.320	0.258	0.350	0.246	0.355	0.254	0.361
	Avg.	0.192	0.281	0.188	0.281	0.205	0.295	0.227	0.296	**0.177**	**0.266**	0.184	0.275	0.201	0.285	0.182	0.272	0.216	0.299	0.199	0.283	**0.178**	**0.270**	0.216	0.318	0.193	0.304	0.225	0.319	0.214	0.327	0.227	0.338

E Computational Cost

E.1 Parameter Counts and Counted Operations

Table 19 reports model size and counted multiply–accumulate operations (MACs) per input sample at $L = 96$. For P patches of width d, global self-attention forms P^2 patch-pair scores with cost $\mathcal{O}(P^2 d)$. The full pipeline also includes entropy estimation, local encoding, and decoder attention. The table specifies which operations are counted.

Table 19: Computational cost of all compared models on each dataset, and the average over the eight datasets (Avg.). Parameters are total inference parameters, including frozen weights. MACs are per sample at $L = 96$, counted over linear and convolutional layers; attention and FFT products are excluded identically for every model. Each model is instantiated with its own official per-dataset and per-horizon configuration. Lower is better.

Models	Parameters (M)									MACs (G)								
	ETTh1	ETTh2	ETTm1	ETTm2	Weather	Solar	ECL	Traffic	Avg.	ETTh1	ETTh2	ETTm1	ETTm2	Weather	Solar	ECL	Traffic	Avg.
FilterTS	0.546	0.565	0.703	0.369	0.246	0.438	1.032	4.909	1.101	0.005	0.006	0.007	0.003	0.006	0.038	0.273	1.178	0.190
HDMixer	0.072	0.071	0.072	0.072	0.074	0.185	0.692	4.536	0.722	0.001	0.001	0.001	0.001	0.003	0.036	0.153	0.948	0.143
TimeBase	<0.001	<0.001	0.002	0.002	0.015	0.101	<0.001	<0.001	0.015	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	0.001	0.004	0.001
TimeMosaic	10.033	7.669	6.091	7.669	10.033	11.609	8.457	5.310	8.359	2.642	1.892	1.330	1.798	7.925	60.872	95.359	129.305	37.640
TSLANet	0.172	0.172	0.620	0.342	0.235	0.235	0.715	0.715	0.401	0.003	0.003	0.023	0.004	0.009	0.058	0.868	2.330	0.412
CALF	180.439	180.439	180.439	180.439	180.439	180.439	180.439	180.439	180.439	0.680	0.680	0.680	0.680	2.039	13.299	31.161	83.679	16.612
DLinear	0.065	0.065	0.065	0.065	0.065	0.065	0.065	0.065	0.065	<0.001	<0.001	<0.001	<0.001	0.001	0.009	0.021	0.056	0.011
FEDformer	16.827	16.827	16.827	16.827	16.899	17.493	18.435	21.206	17.668	2.077	2.077	2.077	2.077	2.098	2.275	2.555	3.380	2.327
PatchTST	5.226	11.531	8.379	9.955	8.379	8.379	8.379	5.230	8.182	0.280	0.809	0.544	0.677	1.633	10.656	24.968	34.510	9.260
TimeMixer	0.196	0.196	0.196	0.201	0.227	0.236	0.249	0.363	0.233	0.003	0.003	0.003	0.006	0.006	0.018	0.036	0.124	0.025
iTransformer	0.255	0.255	0.255	0.255	4.957	4.957	4.957	6.535	2.803	0.003	0.003	0.003	0.003	0.124	0.699	1.610	5.656	1.012
TimesNet	0.629	1.215	2.680	1.069	1.217	1.232	150.328	301.716	57.511	1.271	2.579	3.739	1.946	2.539	2.537	324.723	648.334	123.459
TimeFFM	100.004	100.004	100.004	100.004	100.004	100.004	100.004	100.004	100.004	5.627	5.627	5.627	5.627	16.636	107.979	252.919	678.842	134.861
EntroPE	0.385	0.878	0.344	0.952	0.344	0.695	0.695	1.653	0.743	0.030	0.061	0.024	0.055	0.073	0.869	1.944	12.849	1.988

E.2 Model Size in the Longer-Context Comparison

Table 20 compares the medium EntroPE configuration with the comparator sizes reported for each input length and forecast horizon in Section C.3. These ratios depend on the configuration.

Table 20: Reported parameter counts for EntroPE-M and comparator models. Parentheses give each comparator's size relative to EntroPE-M. K and M denote thousands and millions of parameters, respectively. Ratios are approximate and configuration-specific.

$L \to T$	EntroPE-M	TimeMosaic	PatchTST	LangTime [28]
$36 \to 18$	176.68K	6.40M ($36.2\times$)	408.98K ($2.31\times$)	$\approx$494M+ ($> 2796\times$)
$512 \to 96$	952.83K	9.75M ($10.2\times$)	1.19M ($1.25\times$)	$\approx$494M+ ($> 518\times$)
$512 \to 480$	4.10M	22.49M ($5.5\times$)	4.33M ($1.06\times$)	$\approx$494M+ ($> 120\times$)

E.3 Downstream Training Time Per Epoch

With boundaries computed during downstream training, the dynamic configuration takes an additional 0.1–1.0 seconds per epoch across the six datasets in Table 21. These timings cover the downstream epoch. The entropy estimator is trained separately before this stage.

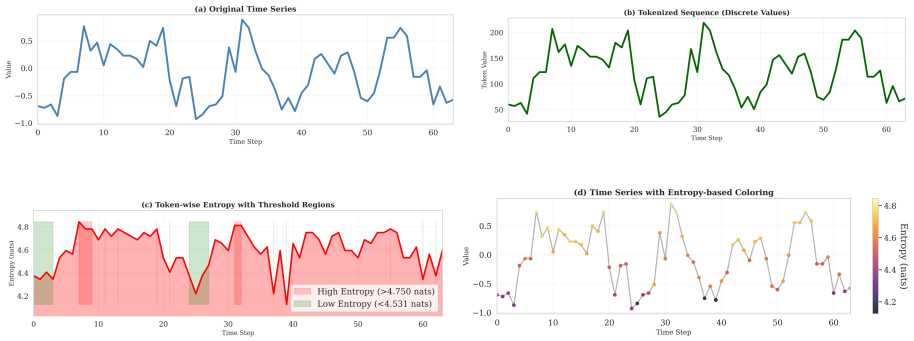
Table 21: Duration in seconds per downstream training epoch for dynamic and static patching. Overhead percentages are rounded.

Metric/Dataset	**ETTh1**	**ETTh2**	**ETTm1**	**ETTm2**	**Weather**	**ECL**
Epoch (dynamic)	3.2	3.5	17.2	20.8	54.5	181
Epoch (static)	3.1	3.4	17.0	20.5	54.1	180
Patching time	0.1	0.1	0.2	0.3	0.4	1
Overhead %	3%	3%	1%	1%	1%	1%

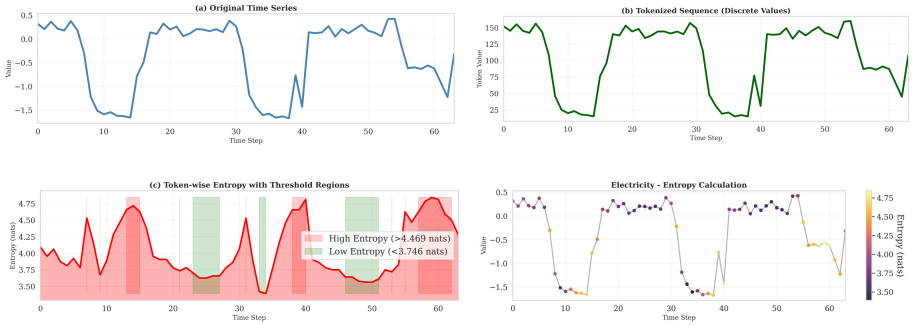
F Patch Behavior and Robustness

F.1 Qualitative Examples of Predictive Entropy

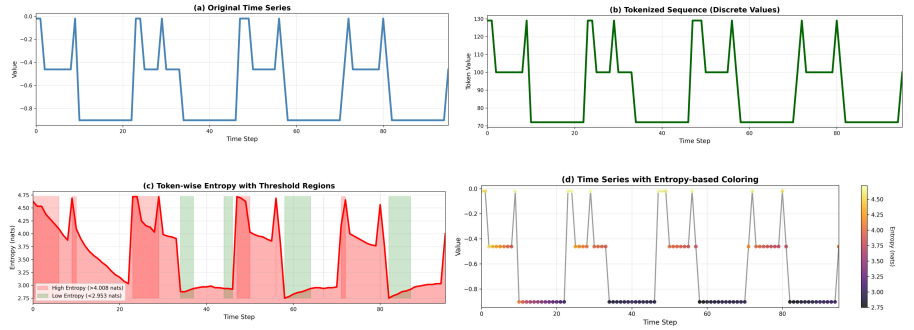
Figure 9 shows three examples from the observed signal to the entropy score used for segmentation. Within each example, the panels show the normalized series, its discrete tokens, the next-token entropy with threshold regions, and the observed series colored by entropy. In the illustrated ETTh1 window, entropy varies from approximately 3.2 to 4.4 nats. The plots show how the estimator responds within individual windows.



(a) Sample 1



(b) Sample 2



(c) Sample 3

Figure 9: Examples from ETT and Electricity. Each example shows the signal, its tokenization, the entropy scores and thresholds, and the signal colored by entropy.

F.2 Patch Counts and Lengths Across Thresholds

We compute patch statistics on 500 deterministic windows per dataset using $L = 512$, seed 2026, and maximum executed patch length 128. Tables 22 and 23 give the mean patch count, mean patch length, and 5th–95th percentile range of patch lengths. Higher thresholds yield fewer, longer patches.

We compute segmentation before the downstream encoder, so these statistics are shared across model sizes when the estimator and boundary rule are fixed.

For this diagnostic, 1,235 M4-Monthly series are long enough to provide a 512-observation window. The M4 forecasting results in Table 13 use $L = 36$ and $T = 18$.

Table 22: Threshold-dependent patch statistics on the ETT datasets. Each cell gives mean patch count / mean patch length, followed by [P05-P95] patch length. All diagnostics use 500 windows, $L = 512$, seed 2026, and maximum executed patch length 128.

Threshold α	ETTh1	ETTh2	ETTm1	ETTm2
0.55	215.68 / 2.37 [1–9]	220.61 / 2.32 [1–7]	224.01 / 2.29 [1–7]	221.53 / 2.31 [1–7]
0.60	191.67 / 2.67 [1–10]	195.40 / 2.62 [1–9]	198.32 / 2.58 [1–9]	195.95 / 2.61 [1–9]
0.65	164.67 / 3.11 [1–12]	169.12 / 3.03 [1–11]	172.46 / 2.97 [1–12]	170.65 / 3.00 [1–11]
0.70	140.28 / 3.65 [1–14]	144.52 / 3.54 [1–14]	147.55 / 3.47 [1–16]	145.77 / 3.51 [1–14]
0.75	115.67 / 4.43 [1–17]	119.25 / 4.29 [1–18]	121.64 / 4.21 [1–22]	120.42 / 4.25 [1–19]
0.80	92.52 / 5.53 [1–23]	94.64 / 5.41 [1–23]	97.57 / 5.25 [1–30]	96.68 / 5.30 [1–27]
0.85	67.63 / 7.57 [1–37]	70.49 / 7.26 [1–37]	72.20 / 7.09 [1–42]	70.58 / 7.25 [1–42]
0.90	44.21 / 11.58 [1–72]	46.87 / 10.92 [1–73]	48.89 / 10.47 [1–67]	47.93 / 10.68 [1–70]
0.95	22.07 / 23.20 [1–128]	23.41 / 21.87 [1–124]	24.62 / 20.80 [1–118]	24.18 / 21.17 [1–104]

Table 23: Threshold-dependent patch statistics on Solar, M4-Monthly, and Weather. M4-Monthly uses $L = 512$ only for this diagnostic, not for forecasting. Each cell gives mean patch count / mean patch length, followed by [P05–P95] patch length. All diagnostics use 500 windows, $L = 512$, seed 2026, and maximum executed patch length 128.

Threshold α	Solar	M4-Monthly	Weather
0.55	224.56 / 2.28 [1–3]	220.44 / 2.32 [1–6]	226.58 / 2.26 [1–5]
0.60	203.32 / 2.52 [1–2]	197.83 / 2.59 [1–10]	201.61 / 2.54 [1–6]
0.65	178.42 / 2.87 [1–4]	171.53 / 2.98 [1–16]	175.55 / 2.92 [1–8]
0.70	153.54 / 3.33 [1–6]	147.35 / 3.47 [1–22]	151.11 / 3.39 [1–11]
0.75	126.00 / 4.06 [1–9]	122.10 / 4.19 [1–25]	125.30 / 4.09 [1–17]
0.80	98.97 / 5.17 [1–18]	97.84 / 5.23 [1–27]	100.18 / 5.11 [1–26]
0.85	74.14 / 6.91 [1–43]	73.62 / 6.95 [1–31]	75.05 / 6.82 [1–44]
0.90	51.46 / 9.95 [1–101]	49.82 / 10.28 [1–36]	50.47 / 10.14 [1–82]
0.95	26.95 / 19.00 [1–128]	25.16 / 20.35 [1–123]	26.13 / 19.60 [1–128]

F.3 Sensitivity to Noise and Dataset Transfer

Additive noise. Table 24 compares ETTh1 boundaries obtained from noisy inputs with those obtained from the clean signal. As the signal-to-noise ratio decreases from 40 to 2 dB, Jaccard similarity falls from 0.626 to 0.281. Mean boundary counts remain close to 22 across these conditions, even as the boundary positions change.

Dataset transfer. When we apply the ETTh1-trained entropy estimator to ETTh2 and Weather, the reported boundary Jaccard similarities are 0.34 and 0.25, respectively. Boundary counts remain close

Table 24: Boundary stability under additive noise on ETTh1. Jaccard similarity is measured against clean-signal boundaries.

SNR	Boundary Jaccard	# Boundaries
Clean	1.000	22.35±0.00
40 dB	0.626	22.61±1.52
20 dB	0.528	22.47±1.67
10 dB	0.419	22.27±1.71
5 dB	0.314	21.82±2.10
2 dB	0.281	21.94±1.89

to 22. The similar counts and lower overlap show that boundary locations can change under transfer even when segmentation density is preserved.

Use of Large Language Models

We refined parts of this paper using a large language model (ChatGPT, GPT-5), exclusively for language polishing and improving clarity of exposition. The authors conceived and executed all ideas, methodology, experiments, and analyses.

Broader Impacts

More accurate forecasts of electricity demand and solar generation could help grid operators plan for changes in supply and demand. In industrial monitoring, `EntroPE` could also help operators identify sensor patterns that warrant inspection. These applications require care when interpreting patch boundaries. A boundary indicates uncertainty about the next observation, and sensor noise can shift its location even when the underlying equipment is operating normally. Treating each boundary as a fault alarm could therefore trigger unnecessary inspections, while processing channels independently could miss a fault visible only across several sensors. Before deployment, operators should evaluate false alarms and missed events on local data under representative noise and operating conditions, and corroborate alerts before taking consequential action.