TRACE: Grounding Time Series in Context for Multimodal Embedding and Retrieval

Jialin Chen[1,*], Ziyu Zhao[2,*], Gaukhar Nurbek[3], Aosong Feng[1], Ali Maatouk[1], Leandros Tassiulas[1], Yifeng Gao[3], Rex Ying[1]

[1]Yale University, [2]McGill University, [3]University of Texas Rio Grande Valley

{jialin.chen, aosong.feng, ali.maatouk, leandros.tassiulas, rex.ying}@yale.edu, ziyu.zhao2@mail.mcgill.ca; {gaukhar.nurbek01, yifeng.gao}@utrgv.edu

Abstract

The ubiquity of dynamic data in domains such as weather, healthcare, and energy underscores a growing need for effective interpretation and retrieval of time-series data. These data are inherently tied to domain-specific contexts, such as clinical notes or weather narratives, making cross-modal retrieval essential not only for downstream tasks but also for developing robust time-series foundation models by retrieval-augmented generation (RAG). Despite the increasing demand, time-series retrieval remains largely underexplored. Existing methods often lack semantic grounding, struggle to align heterogeneous modalities, and have limited capacity for handling multi-channel signals. To address this gap, we propose TRACE, a generic multimodal retriever that grounds time-series embeddings in aligned textual context. TRACE enables fine-grained channel-level alignment and employs hard negative mining to facilitate semantically meaningful retrieval. It supports flexible cross-modal retrieval modes, including Text-to-Timeseries and Timeseries-to-Text, effectively linking linguistic descriptions with complex temporal patterns. By retrieving semantically relevant pairs, TRACE enriches downstream models with informative context, leading to improved predictive accuracy and interpretability. Beyond a static retrieval engine, TRACE also serves as a powerful standalone encoder, with lightweight task-specific tuning that refines context-aware representations while maintaining strong cross-modal alignment. These representations achieve state-of-the-art performance on downstream forecasting and classification tasks. Extensive experiments across multiple domains highlight its dual utility, as both an effective encoder for downstream applications and a general-purpose retriever to enhance time-series models.

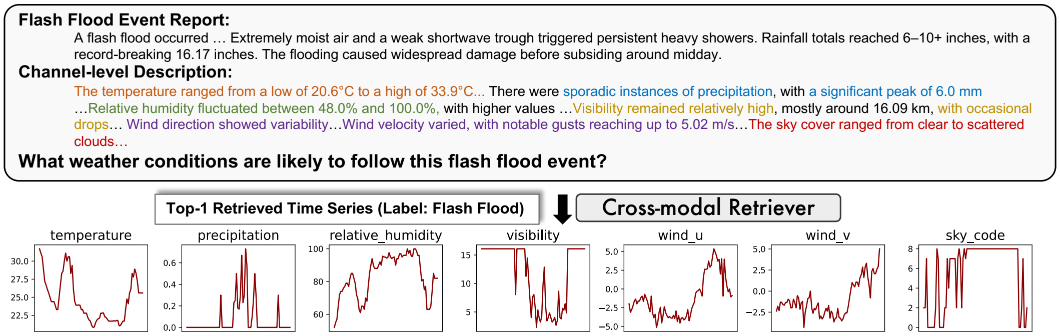
1 Introduction

Time-series data is prevalent across critical domains such as healthcare, weather, and energy [1–4]. Crucially, such data rarely exists in isolation in real-world applications. It is typically accompanied by rich, domain-specific textual context, *e.g.,* clinical notes and weather reports [5–7]. This inherent multimodality necessitates a shift beyond unimodal time-series analysis towards multi-modal frameworks that seamlessly integrate these heterogeneous data types.

Cross-modal retrieval between time series and text is not only natural but necessary. As shown in Figure 1, given a flash flood report describing extreme rainfall and high wind gusts, retrieving historical time series that exhibit similar patterns can support downstream tasks such as weather forecasting and disaster warning. Such retrieval also enables the integration of semantically aligned

[*] Both authors contributed equally to this paper.

Preprint. Under review.



Flash Flood Event Report:

A flash flood occurred … Extremely moist air and a weak shortwave trough triggered persistent heavy showers. Rainfall totals reached 6–10+ inches, with a record-breaking 16.17 inches. The flooding caused widespread damage before subsiding around midday.

Channel-level Description:

The temperature ranged from a low of 20.6°C to a high of 33.9°C... There were sporadic instances of precipitation, with a significant peak of 6.0 mm …Relative humidity fluctuated between 48.0% and 100.0%, with higher values …Visibility remained relatively high, mostly around 16.09 km, with occasional drops… Wind direction showed variability…Wind velocity varied, with notable gusts reaching up to 5.02 m/s…The sky cover ranged from clear to scattered clouds…

What weather conditions are likely to follow this flash flood event?

Figure 1: A Use Case of Text-to-Timeseries Retrieval

external knowledge into time series foundation models [8–10], guiding model attention to relevant segments, and facilitating more generalizable inference via retrieval-augmented generation (RAG).

Despite the clear demand, time-series retrieval, particularly in a cross-modal context, remains significantly underexplored. Existing approaches often fall short in several ways [11–15]. They overlook the rich textual context within time-series data and rely on shallow similarity measures rather than contextual understanding, leading to a lack of effective cross-modal alignment between time-series signals and their associated textual descriptions. Moreover, they struggle with the multi-channel nature of real-world time series, where each channel can encode distinct yet interrelated information [16–18]. Importantly, prior work rarely explores retrieval-augmented generation (RAG) for time series foundation models, restricting their utility in augmenting downstream models.

To address this gap, we introduce TRACE, a novel multimodal **T**ime-series **R**etriever with **A**ligned **C**ontext **E**mbedding. As illustrated in Figure 2, TRACE adopts a two-stage training: a pre-training stage for the time-series encoder, followed by a cross-modal alignment. To address the challenge of modeling multivariate time series, we introduce Channel Identity Tokens (CITs) into a masked autoencoder framework pre-trained at both the token level and channel level in Stage 1. CITs guide the model to attend to unique channel behaviors and enable the learning of channel disentangled representations, overcoming the limitation of conventional decoder-only foundation models which often yield embeddings lacking discriminative power for retrieval and classification. In Stage 2, we propose a novel component for effective cross-modal alignment between time-series embeddings and their textual counterparts through a hierarchical hard negative mining strategy. At the channel level, we identify distractor single-channel segments that exhibit misleadingly similar patterns. At the sample level, we dynamically mine hard negatives by selecting highly similar text descriptions but with divergent semantics. This dual-level contrastive learning encourages the model to learn both local precision and global consistency, leading to strong generalization in downstream tasks.

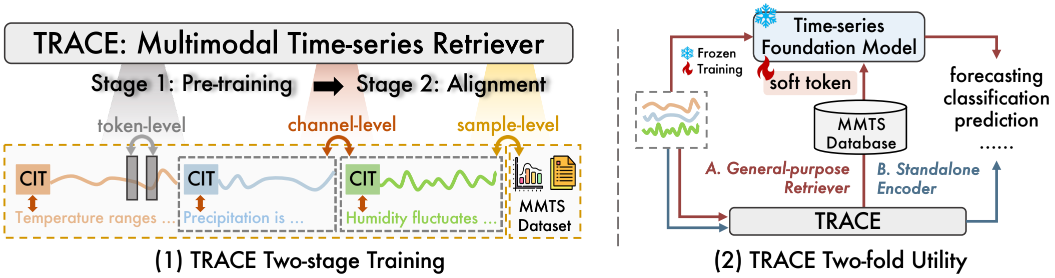


Figure 2: Overview of TRACE. CIT stands for Channel Identity Tokens, which serve as a key bridge to connect two stages. MMTS denotes multimodal time series.

TRACE is designed with a two-fold utility. It acts as a general-purpose retriever, which provides relevant information via a soft token interface. The soft token summarizes retrieved time-series snippets into a latent vector, which is then prepended as a conditioning token, guiding a frozen time-series foundation model towards more context-aware predictions. Moreover, TRACE serves as a powerful standalone encoder, producing rich embeddings that achieve state-of-the-art performance on downstream forecasting and classification tasks. Extensive experiments on both public benchmarks and our curated multimodal dataset validate the effectiveness of TRACE, demonstrating superior retrieval accuracy. The retrieved context substantially boosts downstream time-series models in retrieval-augmented settings, with up to 4.56% increase in classification accuracy and 4.55% reduction in forecasting error. In addition, TRACE produces high-quality time-series embeddings that achieve state-of-the-art results on a wide range of forecasting and classification benchmarks.

The contributions of this paper are: (1) we propose the first multimodal retriever, TRACE, that learns semantically grounded time-series embeddings through fine-grained dual-level alignment; (2) we establish new benchmarks on cross-modal retrieval between time series and text, and (3) extensive validation showcases that TRACE consistently delivers state-of-the-art performance both as a general-purpose retriever for time-series models and a powerful encoder for time series analysis.

2 Related Work

Time Series Forecasting. Recent work on time-series forecasting has led to a range of model architectures, each emphasizing different inductive biases. Transformer-based models leverage self-attention to capture long-range dependencies and flexible temporal dynamics [19–28]. Linear-based models assume time-series signals can be effectively decomposed and modeled with simple linear projections [29, 30]. Frequency-domain and mixing-based approaches aim to model periodicity and multi-scale temporal structures using Fourier transforms or token mixers [31]. Recently, a variety of time series foundation models have emerged. Timer-XL [9] leverages Kronecker attention and is pre-trained with multivariate next-token prediction to enable unified, long-context forecasting. Chronos [32] tokenizes time series via scaling and quantization, and trains a T5-style model for zero-shot probabilistic forecasting. Time-MoE [10] introduces a sparse mixture-of-experts architecture to support variable horizons and input lengths. TimesFM [33] uses input patching and is pre-trained on large-scale data for strong zero-shot performance. Moment [8] and Moirai [34] adopt masked prediction pretraining to enable generalization across diverse multivariate forecasting tasks. While these models perform well on forecasting tasks, they are generally unimodal and not designed for retrieval or integration of external context, highlighting a gap addressed by our cross-modal retrieval framework.

Time Series Language Models. Recently, several multimodal encoders have been proposed to integrate time series and text [35–40], which aim to leverage the generalization capabilities of large language models by reprogramming time series into token-like representations or textual prototypes. ChatTime[41] models time series as a foreign language by normalizing and discretizing continuous signals into token sequences, which are then processed by a large language model (LLM). ChatTS[42] supports both understanding and reasoning by fine-tuning on synthetic datasets generated via attribute-based sampling. TimeXL[43] combines a prototype-based time series encoder with a multimodal prediction framework to capture explainable temporal patterns guided by aligned textual cues. However, they primarily treat text as global context and lack fine-grained alignment between structured time series components and textual semantics, leading to suboptimal cross-modal embedding or retrieval.

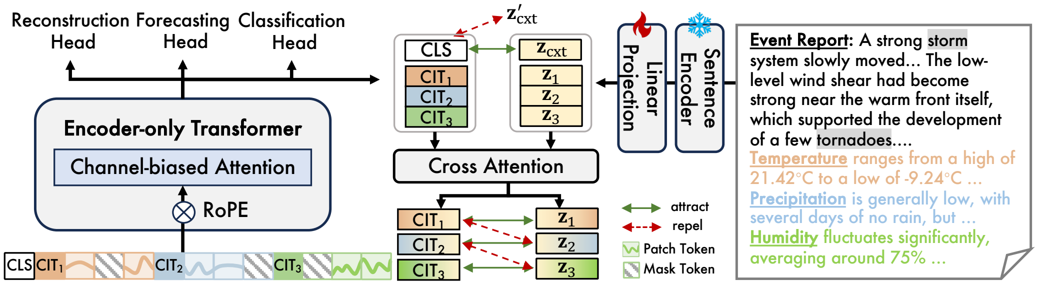
Time Series Retrieval System. Recent work has explored retrieval systems for time series data, primarily within a unimodal setting [44, 13, 45, 15]. CTSR [11] supports content-based time-series retrieval using contextual metadata. TimeRAF [12] integrates a trainable retriever with task-specific time-series knowledge bases for downstream augmentation. TS-RAG [14] retrieves relevant time series segments using pre-trained encoders and combines them via a mixture-of-experts module to improve forecasting. However, all of these methods rely solely on time series embeddings and do not incorporate textual signals, limiting their ability to support multimodal and context-aware retrieval.

3 Proposed Method

As shown in Figure 3, TRACE learns robust time series representations through a masked reconstruction objective with channel-biased attention in the pre-training stage (Sec. 3.2). Then, each time series channel is aligned with its corresponding textual description via fine-grained contrastive learning in the cross-modal alignment stage (Sec. 3.3). We further propose a novel retrieval-augmented generation strategy for time series foundation models, where TRACE retrieves relevant context for downstream tasks (Sec. 3.4). This modular design enables both strong standalone performance and effective integration with existing time series foundation models.

3.1 Problem Definition

Multimodal Time-series. Let $\mathbf{X} \in \mathbb{R}^{C \times T}$ denote a multivariate time series instance, where C is the number of channels (or variables) and T is the number of time steps. We assume the availability of



Event Report: A strong storm system slowly moved... The low-level wind shear had become strong near the warm front itself, which supported the development of a few tornadoes....
Temperature ranges from a high of 21.42°C to a low of -9.24°C ...
Precipitation is generally low, with several days of no rain, but ...
Humidity fluctuates significantly, averaging around 75% ...

Figure 3: Illustration of TRACE, which encodes multivariate time series using channel-biased attention and aligns token embeddings with its corresponding textual description (*e.g.,* $\mathbf{z}_i$ and $\mathbf{z}_{\text{cxt}}$) through cross-attention and dual-level contrastive learning. $\mathbf{z}'_{\text{cxt}}$ indicates an in-batch hard negative sample.

two types of textual information aligned with $\mathbf{X}$. First, for each channel c in an instance $\mathbf{X}$, there is a corresponding textual description τ_c that summarizes the behavior or trend of $\mathbf{X}_c$ over the time window $[0, T)$. These descriptions are denoted as $\mathcal{T}^{\text{ch}} = \{\tau_c | c = 1, \cdots, C\}$. Additionally, there is a sample-level context τ_{cxt} summarizing the overall condition occurring during the same time window, which could be weather reports or clinical narratives, depending on the application domain.

Task Objectives. The goal is to jointly embed the multivariate time series $\mathbf{X}$ and its corresponding textual context $\mathcal{T} = \mathcal{T}^{\text{ch}} \cup \{\tau_{\text{cxt}}\}$ into a shared space that supports multiple downstream tasks, including: (1) forecasting future values $\mathbf{X}_{T:T+H} \in \mathbb{R}^{C \times H}$ for the next H time steps; (2) classification, where the model predicts a categorical label for each time series instance; and (3) cross-modal retrieval, where the goal is to retrieve relevant time series $\mathbf{X}$ based on a text query τ_{cxt} or retrieve historical relevant reports from $\mathcal{T}$ given a time series query, etc.

3.2 Stage 1: Time Series Encoder Pre-training

Time Series Tokenization. Given an input multivariate time series $\mathbf{X} \in \mathbb{R}^{C \times T}$, we divide the temporal dimension into non-overlapping (or strided) patches of length P, resulting in $\hat{T} = \lfloor \frac{T}{P} \rfloor$ patches per channel. Each patch is flattened and linearly projected into a d-dimensional embedding space using a learnable linear projection. This converts each channel into a sequence of patch tokens $X_c^{\text{patch}} \in \mathbb{R}^{\hat{T} \times d}$, for $\forall c \in \{1, \ldots, C\}$. To capture localized semantics within each channel, we prepend a learnable channel identity token $\texttt{[CIT]} \in \mathbb{R}^{1 \times d}$ to the patch token sequence of each channel. These tokens serve as explicit representations of channel-level summaries. Each token is uniquely indexed and not shared across channels, initialized from a standard Gaussian distribution, and trained jointly with the model. This design allows the model to differentiate between channels and effectively aggregate channel-wise patterns. We then concatenate all tokenized channels into a single sequence and insert a global learnable $\texttt{[CLS]}$ token at the beginning of the full sequence. The final token sequence for a multivariate instance is structured as:

$$\mathbf{H} = \Big[\texttt{[CLS]}; \texttt{[CIT]}_1; X_1^{\text{patch}}; \texttt{[CIT]}_2; X_2^{\text{patch}}; \ldots; \texttt{[CIT]}_C; X_C^{\text{patch}} \Big] \in \mathbb{R}^{L \times d}, \tag{1}$$

where $L = C(\hat{T} + 1) + 1$ is the total sequence length after flattening all channel in 1. This tokenization strategy preserves both temporal and structural granularity: patchification encodes token-level patterns; $\texttt{[CIT]}$ summarizes intra-channel dynamics; and $\texttt{[CLS]}$ provides a global and sample-level embedding that can be used for downstream retrieval and classification tasks.

Channel-biased Attention and Rotary PE. To encode channel dependencies in multivariate time series, we introduce a novel Channel-biased Attention (CbA) mechanism that incorporates both inductive bias for channel disentanglement and temporal order encoding via rotary positional embeddings (RoPE) [46]. In our CbA, we design a biased attention mask $M \in \{0, 1\}^{L \times L}$ to prevent unintended semantic entanglement across heterogeneous variables. Specifically, for each channel identity token $\texttt{[CIT]}_c$ located at index i_c in the flattened sequence, we define $M_{i_c, j} = 0$ if token $j \notin$ channel c and 1 otherwise, and $M_{k,j} = 1$ if token k is not a $\texttt{[CIT]}$. Let $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{L \times d}$ be the learned linear projections of the input token embedding $\mathbf{H}$. We apply RoPE to the query ($\mathbf{Q}$) and key ($\mathbf{K}$) vectors before computing attention. RoPE is applied independently within each channel to the $\hat{T}$ temporal tokens, and is not applied to the channel identity tokens, which act as position-agnostic aggregators. The attention weight between tokens i and j in a RoPE-enhanced attention is given by $\alpha_{ij} = \mathrm{softmax}_j \left(Q_i^\top R_{\theta_{\Delta t_{ij}}} K_j / \sqrt{d} + \log M_{ij} \right)$, where $R_{\theta_{\Delta t_{ij}}}(\cdot)$ denotes a rotation by angle $\theta_{\Delta t_{ij}}$, and Δt_{ij} is the relative time difference between tokens i and j in their original unflattened sequence. This is crucial in the multichannel setting, as two tokens that are close in actual time may appear far apart in the flattened sequence. Using Δt_{ij} ensures that the position encoding remains consistent with the true temporal structure rather than the flattened channel order. M_{ij} mask enforces channel disentanglement, while still allowing rich token-level interactions across the full sequence.

Pre-training Setup. We adopt an encoder-only Transformer [47] with multi-head channel-based attention layers in TRACE. We apply reversible instance normalization [48] to multivariate time series before tokenizing and embedding. A fixed proportion of these tokens is randomly masked with a mask ratio of γ, and the model is pre-trained to reconstruct the missing values based on the unmasked context. We use mean squared error (MSE) loss to supervise pre-training, encouraging the model to capture cross-channel dependencies while learning transferable representations for downstream tasks.

3.3 Stage 2: Multimodal Alignment Learning

Motivation. Standard contrastive learning methods typically rely on sample-level random negatives. However, textual descriptions frequently reference specific variables (*e.g.,* temperature spikes, wind gusts), which cannot be precisely aligned using a single global embedding. To address this, we introduce channel-level alignment that explicitly models the interaction between individual time-series channels and their corresponding textual context. This not only enhances semantic precision but also promotes modularity in representation learning and enables variable-specific interactions.

Cross-attention Between Modalities. After pre-training the time-series encoder via masked reconstruction, we obtain hidden embedding $\mathbf{H}^{\text{out}} \in \mathbb{R}^{L \times d}$ from the final transformer layer, where L is the full sequence length after flattening all channels. From this, we extract the [CLS] token embedding $\mathbf{h}_{\texttt{[CLS]}} \in \mathbb{R}^d$, and the set of channel identity token embeddings $\mathbf{H}_{\texttt{[CIT]}} = [\mathbf{h}_1, \ldots, \mathbf{h}_C] \in \mathbb{R}^{C \times d}$, each corresponding to a [CIT] token and serving as fine-grained anchors that enable structured reasoning at the channel level. Let τ_{cxt} and τ_c denote the sample-level and the c-th channel textual context for a time series instance, respectively. The textual inputs are first encoded using a pre-trained language model (*e.g.,* a frozen Sentence-Transformer [49]), followed by a learnable linear layer that projects them into the same d-dimensional embedding space as the time series representations, collectively denoted as $f_{\text{t}}(\cdot)$. This yields semantic embeddings $\mathbf{z}_{\text{cxt}} = f_{\text{t}}(\tau_{\text{cxt}}) \in \mathbb{R}^d$ for the sample-level context and $\mathbf{z}_c = f_{\text{t}}(\tau_c) \in \mathbb{R}^d$ for each channel-level description. We further apply a cross-attention between $\mathbf{H}_{\texttt{[CIT]}} \in \mathbb{R}^{C \times d}$ and channel text embeddings $\mathbf{Z}_{\text{ch}} = [\mathbf{z}_1, \ldots, \mathbf{z}_C] \in \mathbb{R}^{C \times d}$, allowing information to be fused across aligned channels. This interaction allows the model to refine its channel-wise time-series representations using semantically aligned textual information.

Dual-level Hard Negative Mining. To enhance the discriminative capacity of the model, we develop a dual-level hard negative mining strategy that introduces fine-grained contrastive pressure at both the sample and channel levels. This approach enables the model to distinguish not only between unrelated time series and text, but also between subtly confusable pairs that share superficial temporal similarity but diverge semantically. For each time series instance i, we mine negative candidates from all other sample-level reports in the same batch based on embedding cosine similarity. For a certain channel, we mine channel-level negatives from a broader candidate pool that includes both intra-instance distractors (other channels within the same sample) and inter-instance distractors (same-indexed channels across different samples). Specifically, for the c-th channel of the i-th instance, we define the sample-level and channel-level negative candidate set as

$$\mathcal{N}_{\text{cxt}}^{(i)} = \mathrm{Top}_K \left\{ \mathrm{sim}(\mathbf{h}_{\texttt{[CLS]}}^{(i)}, \mathbf{z}_{\text{cxt}}^{(j)}) \mid j \neq i \right\}, \mathcal{N}_{\text{ch}}^{(i,c)} = \mathrm{Top}_K \left\{ \mathrm{sim}(\mathbf{h}_c^{(i)}, \mathbf{z}_{c'}^{(j)}) \mid c' \neq c \text{ or } j \neq i \right\},$$

where K is number of negative samples at each level. Symmetric negative sets are defined in the reverse direction for $\mathbf{z}_{\text{cxt}}^{(i)}$ and $\mathbf{z}_{\text{c}}^{(i)}$ by swapping the roles of time series and text. We then compute a bidirectional InfoNCE loss at sample levels: $\mathcal{L}_{\text{global}}^{\text{text}\to\text{ts}}$, $\mathcal{L}_{\text{global}}^{\text{ts}\to\text{text}}$, and similarly for channel-level losses. The total alignment objective is the average of both directions (Formulations detailed in Appendix C):

$$\mathcal{L}_{\text{align}} = \frac{1}{2} \left(\mathcal{L}_{\text{global}}^{\text{text} \to \text{ts}} + \mathcal{L}_{\text{global}}^{\text{ts} \to \text{text}} \right) + \lambda_{\text{ch}} \cdot \frac{1}{2} \left(\mathcal{L}_{\text{channel}}^{\text{text} \to \text{ts}} + \mathcal{L}_{\text{channel}}^{\text{ts} \to \text{text}} \right), \tag{2}$$

where λ_{ch} controls the contribution of channel-level alignment. The entire alignment objective is optimized jointly with the trainable parameters of the time series encoder in the pre-training stage and the linear projection head in f_{t}, while keeping the backbone language model frozen.

3.4 Retrieval-augmented Generation with Time Series Foundation Models

As shown in Figure 2, TRACE enables retrieval-augmented generation (RAG) for time series foundation models, inspired by the success of RAG in NLP [50, 13]. Given a query time series, TRACE retrieves semantically relevant time-series–text pairs from a large multimodal corpus based on the embedding similarity. The retrieved context is then encoded into a soft token, which is a trainable, dense vector that serves as a continuous prompt to condition the downstream forecasting model. This design allows the forecaster to incorporate external knowledge without architectural modification. Importantly, the base time-series foundation model remains frozen during training, as the soft tokens are differentiable and model-agnostic, improving efficiency and enabling plug-and-play integration across diverse backbone architectures. In effect, TRACE acts as a structured, external memory, enriching the model's input with historically grounded and semantically aligned context.

4 Experiments

We evaluate TRACE from three key perspectives: (1) its effectiveness in cross-modal retrieval compared to strong time series encoders (Sec. 4.2), (2) its utility as a retriever in retrieval-augmented forecasting pipelines (Sec 4.3), and (3) its generalization ability as a standalone encoder for forecasting and classification (Sec. 4.4). Experiments are conducted on public benchmarks and our curated multimodal dataset designed to assess cross-modal alignment and retrieval performance.

4.1 Experimental Setting

Dataset. To support real-world multimodal time series applications, we construct a new dataset in the weather domain with three aligned components: multivariate time series, sample-level event reports, and synthetic channel-level descriptions, specifically for downstream forecasting and event-type classification tasks. The event reports are sourced from the NOAA Events Database [51], while the associated time series data are retrieved from the NOAA Global Historical Climatology Network (GHCN) [52]. We focus on stations and time windows characterized by frequent severe weather events and extract historical multivariate time-series segments at multiple temporal resolutions, anchored at event onset. To enhance data diversity and model robustness, we also sample non-event (*i.e.,* typical) periods from the same stations, as well as from geographically distinct locations. Each time-series segment includes seven variables (*e.g.,* temperature, relative humidity, precipitation) and is annotated with either a specific event type or a non-event label. To evaluate performance in the univariate setting, we further incorporate the three largest subsets—Health, Energy, and Environment—from TimeMMD [5], a multimodal benchmark designed for time series forecasting, where each single-variate instance is aligned with a sample-level textual report (*e.g.,* clinical notes, incident logs). This setting allows us to assess the model's generalization across diverse domains and varying channel configurations. Full dataset details and illustrative examples are provided in Appendix B.

Baselines. We evaluate against the state-of-the-art traditional time series models and recent time series foundation models. Traditional baselines include DLinear [29], iTransformer [24], PatchTST [22], TimesNet [53], TimeMixer [54], and multimodal model FSCA [37]. These models are trained from scratch on each task. For foundation models, we include Chronos[32], TimesFM [55], Timer-XL [9], Time-MoE [10], Moirai [34] and Moment [8]. We refer to Appendix D.1 for baseline details.

Implementation Details. The default TRACE consists of a 6-layer Transformer encoder with a hidden dimension of 384 and 6 attention heads. We use the AdamW [56] optimizer with a linear warmup followed by a cosine decay schedule. Pre-training is conducted with a mask ratio of 0.3, and runs for up to 400 epochs. We take 32 in-batch negative samples at each level in the alignment stage and run for up to 300 epochs. All experiments are conducted over five runs with different random seeds on NVIDIA A100 40GB GPUs. We refer to Appendix D.2 for experiment configurations and details.

4.2 Cross-modal Retrieval

Alignment Setup. To evaluate the model's retrieval performance, we conduct a controlled comparison by replacing the encoder in TRACE with several strong time series foundation models that produce fixed-length embeddings. Each encoder is jointly fine-tuned end-to-end with a lightweight projection layer following the sentence encoder, using a contrastive learning objective. While TRACE leverages

`[CLS]` and `[CIT]` embeddings for dual-level alignment, other baselines use mean pooling over the sequence due to their architectural constraints.

Evaluation Metrics. `TRACE` supports flexible retrieval modes, including cross-modal (Text-to-TS and TS-to-Text) and unimodal TS-to-TS retrieval. We provide results of TS-to-TS retrieval in Appendix D.8. For cross-modal retrieval, a query in one modality is used to retrieve its corresponding counterpart in the other modality based on embedding cosine similarity. The evaluation includes several metrics:

- **Label Matching** uses P@k to measure the precision of correctly labeled items among the top-k retrieval, and Mean Reciprocal Rank (MRR) to assess the rank of the first correct item.
- **Modality Matching** evaluates whether a query retrieves its paired instance from the opposite modality, using P@k for top-k precision and MRR for the rank of the true counterpart.
- **Text Similarity** uses ROUGE between the query text and the text paired with the top-1 retrieved time series (for text-to-ts scenario), or between the top-1 retrieved text and the original text paired with the query time series (for ts-to-text scenario).
- **Time Series Similarity** computes MAE and MSE between the time series linked to the query and that of the top-1 retrieved pair, defined similarly to Text Similarity.

Results. As shown in Table 1, `TRACE` consistently achieves state-of-the-art performance in two retrieval settings with approximately 90% top-1 label matching and 44% top-1 modality matching. Notably, this retrieval precision surpasses the classification accuracy of all train-from-scratch models reported in Table 3, highlighting the strength of alignment supervision in learning discriminative representations. Among baselines, Moment outperforms other foundation models, suggesting that encoder-only architectures are better suited for dense retrieval tasks. In contrast, `TRACE` provides fine-grained embeddings for cross-modal alignment, enabling it to recover semantical counterparts with high precision. `TRACE` supports flexible retrieval modes, such as TS-to-TS.

Table 1: Retrieval results on 2,000 bidirectional Text–Timeseries query pairs. "Random" indicates a non-informative retriever that ranks candidates uniformly at random.

		Label Matching			**Modality Matching**			**Text**	**Time Series**	
	Retriever	P@1 (↑)	P@5 (↑)	MRR (↑)	P@1 (↑)	P@5 (↑)	MRR (↑)	ROUGE (↑)	MAE (↓)	MSE (↓)
	Random	42.61	47.50	0.583	0.00	0.00	0.00	0.416	0.874	1.653
TS-to-Text	*w/* Time-MoE	46.46	43.98	0.612	1.79	5.93	0.052	0.482	0.837	1.607
	w/ Timer-XL	36.34	38.16	0.543	4.29	12.61	0.090	0.482	0.793	1.493
	w/ TS2Vec	50.47	48.72	0.651	4.37	14.57	0.112	0.503	0.784	1.462
	w/ Moment	55.73	53.18	0.691	7.78	21.68	0.154	0.515	0.747	1.415
	`TRACE`	**90.08**	**77.60**	**0.940**	**44.10**	**70.24**	**0.560**	**0.717**	**0.403**	**0.771**
Text-to-TS	*w/* Time-MoE	57.08	52.22	0.656	0.75	2.89	0.031	0.460	0.857	1.578
	w/ Timer-XL	63.91	58.71	0.731	2.94	9.47	0.073	0.463	0.821	1.568
	w/ TS2Vec	60.28	56.41	0.706	7.42	23.70	0.184	0.471	0.806	1.490
	w/ Moment	64.67	59.53	0.740	5.83	18.15	0.133	0.488	0.778	1.467
	`TRACE`	**89.63**	**78.39**	**0.938**	**43.72**	**69.84**	**0.557**	**0.713**	**0.411**	**0.793**

4.3 Retrieval-augmented Time Series Forecasting

Setup. We use `TRACE` to retrieve the most relevant timeseries–text pairs from the curated corpus based on time-series embedding similarity, which is then passed through trainable linear layers to produce a soft prompt. For *TS-only* setting, the prompt is derived solely from the retrieved raw time series, denoted as h_{ts}; for *TS+Text*, we concatenate h_{ts} and semantic embedding $\mathbf{z}_{\text{cxt}}$ from the retrieved text to form the prompt. This soft prompt is then prepended to the query for the downstream forecasting layer, without fine-tuning the pre-trained model weights. We test two architecture families: (1) decoder-only models, including Timer-XL and Time-MoE, where the prompt is prepended at every autoregressive generation step, and (2) encoder-only models, Moment and `TRACE`, where the prompt is prepended to the encoder's hidden states and followed by a trainable forecasting head. In all settings, only the linear projection layers for prompt generation and the forecasting head (for encoder-only) are trained.

Table 2: Forecasting performance on Weather dataset for next 24 steps under different retrieval-augmented generation settings.

	Timer-XL		**Time-MoE**		**Moment**		`TRACE`	
Setting	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
w/o RAG	0.729	1.055	0.635	0.903	0.645	0.816	0.576	0.718
w/ TS-only	0.720	1.009	0.621	0.801	0.628	0.797	0.556	0.698
w/ TS+Text	0.712	0.984	0.611	0.787	0.631	0.801	0.555	0.696

Table 4: Forecasting results (MAE and MSE) of full-shot models and time series foundation models on multi-variate (**M**) and univariate (**U**) datasets. **Red**: the best, Blue: the 2nd best.

	Model	Weather (M)				Health (U)				Energy (U)				Environment (U)				# 1^{st}
		H = 7		H = 24		H = 12		H = 48		H = 12		H = 48		H = 48		H = 336		
		MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	
Zero-shot	Chronos	0.560	0.937	0.646	1.094	0.650	1.106	0.987	2.019	0.263	0.148	0.554	0.553	0.536	0.612	0.583	0.671	0
	Time-MoE	0.579	0.803	0.635	0.903	0.604	0.981	0.832	1.697	**0.205**	**0.089**	0.451	0.396	0.562	0.508	0.836	0.969	2
	TimesFM	0.550	0.859	0.640	1.034	0.610	0.913	0.865	1.685	0.248	0.137	0.499	0.482	0.503	0.532	0.531	0.569	0
	Timer-XL	0.645	0.912	0.729	1.055	0.741	1.235	0.988	1.892	0.236	0.118	0.460	0.424	0.549	0.564	0.565	0.574	0
	Moirai	0.593	1.001	0.675	1.135	0.976	3.029	1.569	8.125	0.318	0.273	0.692	1.415	0.935	12.428	2.237	25.011	0
	Moment	0.572	0.732	0.645	0.816	0.988	1.824	0.997	1.902	0.471	0.411	0.542	0.542	**0.449**	**0.375**	0.554	0.502	2
Full-shot	DLinear	0.593	0.778	0.691	0.884	1.178	2.421	1.132	2.256	0.410	0.273	0.546	0.512	0.561	0.515	0.581	0.534	0
	iTransformer	0.518	0.707	0.591	0.814	0.676	1.072	0.911	1.747	0.267	0.124	0.487	0.399	0.486	0.425	0.511	0.458	0
	PatchTST	0.529	0.723	0.599	0.826	0.656	1.034	0.902	1.708	0.263	0.121	0.489	0.407	0.493	0.462	0.525	0.511	0
	TimesNet	0.497	0.654	0.581	0.786	0.820	1.376	0.969	1.903	0.270	0.127	0.496	0.398	0.520	0.486	0.489	0.430	0
	TimeMixer	0.501	0.667	0.585	0.787	1.091	2.215	1.126	2.250	0.376	0.246	0.538	0.491	0.558	0.553	0.559	0.568	0
	FSCA	**0.496**	0.642	0.780	0.762	0.756	1.240	0.969	1.904	0.278	0.136	0.520	0.466	0.497	0.462	0.511	0.496	1
	TRACE	0.501	**0.623**	**0.576**	**0.718**	**0.547**	**0.768**	**0.827**	**1.435**	0.230	0.113	**0.448**	**0.389**	0.455	0.403	**0.475**	**0.413**	**11**

Results. Table 2 presents the forecasting results across decoder-only and encoder-only models under different RAG settings, augmented by top-R retrieved instances. We refer to Figure 4 (d) for ablation on R. The results reveal that retrieval augmentation consistently improves forecasting performance across all models, and the *TS+Text* setting leads to the most significant gains for decoder-only models like Timer-XL and Time-MoE. Notably, TRACE shows marginal improvement when moving from *TS-only* to *TS+Text* retrieval, which can be attributed to that its multimodal embedding space is already aligned with textual descriptions. This alignment reduces the dependency on additional textual signals and justifies TRACE 's design as a lightweight, general-purpose retriever for RAG pipelines. Moreover, these results indicate decoder-only models are more sensitive to the richness of retrieved modalities, whereas encoder-only models exhibit more stable and better capacity for internalizing and utilizing structured representations. While our RAG design adopts a simple embedding concatenation strategy, it primarily validates the general utility of retrieved content across different model families. We leave optimizing augmentation architectures for future work.

4.4 Standalone Time Series Encoder

Setup. To evaluate TRACE as a standalone encoder, we conduct experiments on forecasting and classification tasks. We compare TRACE against full-shot models all trained from scratch, and time series foundation models. All foundation models are evaluated in a zero-shot setting, except for Moment and TRACE, which are fine-tuned on the forecasting head following the official protocol for forecasting. For classification, we evaluate on our curated weather dataset and fine-tune all foundation models in the same setting to ensure a fair comparison (detailed in Appendix D.6).

Results. As shown in Table 4, TRACE outperforms baselines across different datasets and showcases capability on longer forecasting horizons (H), whereas the performance of baselines exhibits considerable variation. This observation justifies the cross-modal design behind TRACE, which equips the model with stronger semantic grounding and context-aware forecasting. In the event-type classification task (as shown in Table 3), we observe that fine-tuned foundation models underperform traditional train-from-scratch baselines, suggesting that their embeddings may be overgeneralized and poorly adapted to domain-specific classification signals. In contrast, TRACE achieves significantly higher accuracy and F1 without RAG, and benefits further from the retrieval-augmented setting. This demonstrates TRACE's ability to retain discriminative structure while maintaining broad semantic alignment, which is essential for robust downstream deployment. Full results of other foundation model variants are in Appendix D.4.

Table 3: Weather Event Classification Results.

Model	**Accuracy**	**F1**
Train-from-scratch Model		
DLinear	82.37	65.78
iTransformer	84.99	68.29
PatchTST	84.78	69.13
TimesNet	86.09	68.97
TimeMixer	84.78	68.65
FSCA	85.62	69.41
Finetune a Pre-trained Model		
Time-MoE$_{\text{large}}$	59.09	19.74
Moment$_{\text{base}}$	65.43	28.29
Timer-XL	72.38	33.45
Chronos$_{\text{tiny}}$	74.79	40.21
TRACE *w/o* RAG	85.20	69.98
TRACE *w/* RAG	**89.76**	**72.36**

5 Ablation Studies

Hyper-parameter Sensitivity. Figure 4 presents a comprehensive ablation study investigating the effects of patch length P, positional embedding (PE) types, and the number of retrieved instances

R used in our RAG setup. Rotary PE consistently outperforms Relative PE by achieving lower reconstruction and forecasting MSEs as well as higher classification accuracy, particularly when using a smaller model size ($d = 384$). Notably, increasing the model size to $d = 768$ does not yield significant improvements, especially for downstream forecasting and classification tasks, suggesting that careful architectural design and PE choice may matter more than simply scaling parameters. Across tasks, mid-range patch lengths (*e.g.,* $P = 6$) offer the best trade-off between local and global temporal resolution. In Figure 4 (d), we observe that time series foundation models are relatively robust to the choice of R, and models augmented with aligned text generally outperform their TS-only counterparts, highlighting the benefit of cross-modal retrieval in improving forecasting performance.

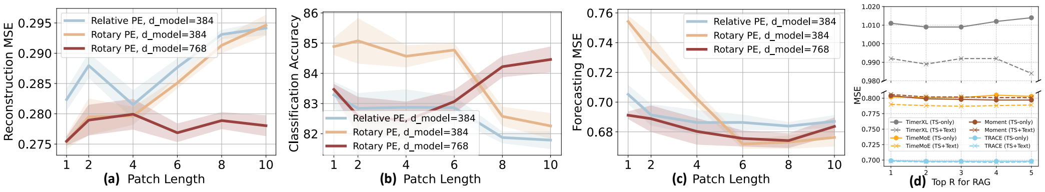


Figure 4: Ablation studies on patch length, positional embedding, and hidden dimension for (a) Reconstruction MSE, (b) Classification Accuracy (%), and (c) Average Forecasting MSE. (d) shows ablation studies on the number of retrieved instances (R) in the RAG pipeline.

Attention Variants. Table 5 assess the impact of key architectural choices in TRACE, including channel identity token (CIT) and different attention mechanisms. Removing CIT results in a notable increase in average MSE, indicating its importance for capturing fine-grained temporal dependencies. We also replace the channel-biased attention (CbA) with two alternatives: full attention, similar to a multivariate variant of Moment [8], and causal attention, analogous to decoder-only designs like Timer-XL [9], Both alternatives yield degraded performance. These results highlight the effectiveness of the architectural design in TRACE, particularly the synergy between CIT and CbA in achieving outstanding performance. We refer to Appendix D.9 for runtime and efficiency evaluation.

Cross-Attention and Hard Negative Sampling. Figure 5 presents an ablation study on key components in TRACE for retrieval precision under different numbers of negative samples (K). "all" indicates using the entire batch (excluding the paired counterpart) as negatives. The default model, using nomic text encoder [57], consistently achieves the highest performance, especially when K is small, highlighting its efficacy in low-computation settings. Removing the final cross-attention module between time series and text leads to notable performance degradation under small K, suggesting that cross-modal fusion becomes especially crucial when fewer negatives are available. Similarly, eliminating channel-level alignment yields a consistent drop, confirming the strength of the proposed dual-level contrastive mechanism. Substituting nomic with weaker text encoders like bge or MiniLM results in worse performance, implying that high-quality embeddings are necessary for discriminating harder negatives. Overall, these trends support the effectiveness of our hard negative mining strategy and emphasize the importance of dual-level alignment in retrieval performance. We provide empirical case studies in Appendix D.7.

Table 5: Ablation study on attention variants in pre-training architecture.

	Avg. MSE	Acc. (%)
TRACE	**0.670±0.013**	**85.20±0.13**
w/o CIT	0.713±0.016	85.04±0.26
CbA ⇒ Full Attn	0.705±0.013	84.18±0.11
CbA ⇒ Causal Attn	0.682±0.015	83.72±0.13

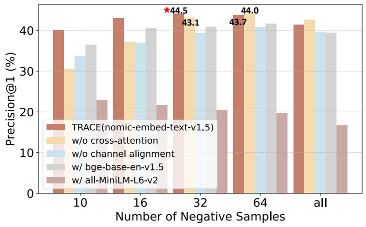


Figure 5: Retrieval performance under varying numbers of negative samples. The best is indicated by ⋆.

6 Conclusion and Future Work

We introduce TRACE, a multimodal framework that aligns time series with textual descriptions at both channel and sample levels. Extensive experiments demonstrate that TRACE outperforms strong baselines across retrieval, standalone encoding, retrieval-augmented settings, and generalizes well across model families. One limitation is that TRACE relies on supervised textual alignment, which may not be readily available in all domains. Future work includes extending to more real-world domains and exploring more expressive integration mechanisms for time-series RAG. We refer to Appendix E for detailed discussion on social impact and future works.

References

[1] Francisco Martinez Alvarez, Alicia Troncoso, Jose C Riquelme, and Jesus S Aguilar Ruiz. Energy time series forecasting based on pattern sequence similarity. *IEEE Transactions on Knowledge and Data Engineering*, 23(8):1230–1243, 2010.

[2] Irena Koprinska, Dengsong Wu, and Zheng Wang. Convolutional neural networks for energy time series forecasting. In *2018 international joint conference on neural networks (IJCNN)*, pages 1–8. IEEE, 2018.

[3] Rafal A Angryk, Petrus C Martens, Berkay Aydin, Dustin Kempton, Sushant S Mahajan, Sunitha Basodi, Azim Ahmadzadeh, Xumin Cai, Soukaina Filali Boubrahimi, Shah Muhammad Hamdi, et al. Multivariate time series dataset for space weather data analytics. *Scientific data*, 7(1):1–13, 2020.

[4] Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-Rosen, Weihua Hu, et al. Learning skillful medium-range global weather forecasting. *Science*, 382(6677):1416–1421, 2023.

[5] Haoxin Liu, Shangqing Xu, Zhiyuan Zhao, Lingkai Kong, Harshavardhan Kamarthi, Aditya B Sasanur, Megha Sharma, Jiaming Cui, Qingsong Wen, Chao Zhang, et al. Time-mmd: A new multi-domain multimodal dataset for time series analysis. *arXiv preprint arXiv:2406.08627*, 2024.

[6] Jialin Chen, Aosong Feng, Ziyu Zhao, Juan Garza, Gaukhar Nurbek, Cheng Qin, Ali Maatouk, Leandros Tassiulas, Yifeng Gao, and Rex Ying. Mtbench: A multimodal time series benchmark for temporal reasoning and question answering, 2025.

[7] Geon Lee, Wenchao Yu, Kijung Shin, Wei Cheng, and Haifeng Chen. Timecap: Learning to contextualize, augment, and predict time series events with large language model agents, 2025.

[8] Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. Moment: A family of open time-series foundation models. In *Proceedings of the 41st International Conference on Machine Learning*, pages 16115–16152. PMLR, 2024.

[9] Yong Liu, Guo Qin, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer-xl: Long-context transformers for unified time series forecasting. *arXiv preprint arXiv:2410.04803*, 2024.

[10] Xiaoming Shi, Shiyu Wang, Yuqi Nie, Dianqi Li, Zhou Ye, Qingsong Wen, and Ming Jin. Time-moe: Billion-scale time series foundation models with mixture of experts. *arXiv preprint arXiv:2409.16040*, 2024.

[11] Chin-Chia Michael Yeh, Huiyuan Chen, Xin Dai, Yan Zheng, Junpeng Wang, Vivian Lai, Yujie Fan, Audrey Der, Zhongfang Zhuang, Liang Wang, et al. An efficient content-based time series retrieval system. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 4909–4915, 2023.

[12] Huanyu Zhang, Chang Xu, Yi-Fan Zhang, Zhang Zhang, Liang Wang, Jiang Bian, and Tieniu Tan. Timeraf: Retrieval-augmented foundation model for zero-shot time series forecasting. *arXiv preprint arXiv:2412.20810*, 2024.

[13] Jingwei Liu, Ling Yang, Hongyan Li, and Shenda Hong. Retrieval-augmented diffusion models for time series forecasting. *Advances in Neural Information Processing Systems*, 37:2766–2786, 2024.

[14] Kanghui Ning, Zijie Pan, Yu Liu, Yushan Jiang, James Y Zhang, Kashif Rasul, Anderson Schneider, Lintao Ma, Yuriy Nevmyvaka, and Dongjin Song. Ts-rag: Retrieval-augmented generation based time series foundation models are stronger zero-shot forecaster. *arXiv preprint arXiv:2503.07649*, 2025.

[15] Silin Yang, Dong Wang, Haoqi Zheng, and Ruochun Jin. Timerag: Boosting llm time series forecasting via retrieval-augmented generation. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2025.

[16] Lu Han, Han-Jia Ye, and De-Chuan Zhan. The capacity and robustness trade-off: Revisiting the channel independent strategy for multivariate time series forecasting. *arXiv preprint arXiv:2304.05206*, 2023.

[17] Jongseon Kim, Hyungjoon Kim, HyunGi Kim, Dongjun Lee, and Sungroh Yoon. A comprehensive survey of deep learning for time series forecasting: architectural diversity and open challenges. *Artificial Intelligence Review*, 58(7):1–95, 2025.

[18] Jialin Chen, Jan Eric Lenssen, Aosong Feng, Weihua Hu, Matthias Fey, Leandros Tassiulas, Jure Leskovec, and Rex Ying. From similarity to superiority: Channel clustering for time series forecasting. *Advances in Neural Information Processing Systems*, 37:130635–130663, 2024.

[19] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

[20] Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary transformers: Rethinking the stationarity in time series forecasting. *arXiv preprint arXiv:2205.14415*, 2022.

[21] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34:22419–22430, 2021.

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

[23] Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The Eleventh International Conference on Learning Representations*, 2022.

[24] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting, 2024.

[25] Binh Tang and David S Matteson. Probabilistic transformer for time series analysis. *Advances in Neural Information Processing Systems*, 34:23592–23608, 2021.

[26] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. *arXiv preprint arXiv:2201.12740*, 2022.

[27] Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X Liu, and Schahram Dustdar. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *International Conference on Learning Representations*, 2021.

[28] Aosong Feng, Jialin Chen, Juan Garza, Brooklyn Berry, Francisco Salazar, Yifeng Gao, Rex Ying, and Leandros Tassiulas. Efficient high-resolution time series classification via attention kronecker decomposition. *arXiv preprint arXiv:2403.04882*, 2024.

[29] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? *arXiv preprint arXiv:2205.13504*, 2022.

[30] Si-An Chen, Chun-Liang Li, Nate Yoder, Sercan O Arik, and Tomas Pfister. Tsmixer: An all-mlp architecture for time series forecasting. *arXiv preprint arXiv:2303.06053*, 2023.

[31] 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. *arXiv preprint arXiv:2405.14616*, 2024.

[32] Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.

[33] Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Forty-first International Conference on Machine Learning*, 2024.

[34] Xu Liu, Juncheng Liu, Gerald Woo, Taha Aksu, Yuxuan Liang, Roger Zimmermann, Chenghao Liu, Silvio Savarese, Caiming Xiong, and Doyen Sahoo. Moirai-moe: Empowering time series foundation models with sparse mixture of experts. *arXiv preprint arXiv:2410.10469*, 2024.

[35] Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, et al. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*, 2023.

[36] Tian Zhou, Peisong Niu, Liang Sun, Rong Jin, et al. One fits all: Power general time series analysis by pretrained lm. *Advances in neural information processing systems*, 36:43322–43355, 2023.

[37] Yuxiao Hu, Qian Li, Dongxiao Zhang, Jinyue Yan, and Yuntian Chen. Context-alignment: Activating and enhancing llm capabilities in time series. *arXiv preprint arXiv:2501.03747*, 2025.

[38] Chenxi Liu, Qianxiong Xu, Hao Miao, Sun Yang, Lingzheng Zhang, Cheng Long, Ziyue Li, and Rui Zhao. Timecma: Towards llm-empowered multivariate time series forecasting via cross-modality alignment. *arXiv preprint arXiv:2406.01638*, 2024.

[39] Zijie Pan, Yushan Jiang, Sahil Garg, Anderson Schneider, Yuriy Nevmyvaka, and Dongjin Song. s^2 ip-llm: Semantic space informed prompt learning with llm for time series forecasting. In *Forty-first International Conference on Machine Learning*, 2024.

[40] Taibiao Zhao, Xiaobing Chen, and Mingxuan Sun. Enhancing time series forecasting via multi-level text alignment with llms. *arXiv preprint arXiv:2504.07360*, 2025.

[41] Chengsen Wang, Qi Qi, Jingyu Wang, Haifeng Sun, Zirui Zhuang, Jinming Wu, Lei Zhang, and Jianxin Liao. Chattime: A unified multimodal time series foundation model bridging numerical and textual data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 12694–12702, 2025.

[42] Zhe Xie, Zeyan Li, Xiao He, Longlong Xu, Xidao Wen, Tieying Zhang, Jianjun Chen, Rui Shi, and Dan Pei. Chatts: Aligning time series with llms via synthetic data for enhanced understanding and reasoning. *arXiv preprint arXiv:2412.03104*, 2024.

[43] Yushan Jiang, Wenchao Yu, Geon Lee, Dongjin Song, Kijung Shin, Wei Cheng, Yanchi Liu, and Haifeng Chen. Explainable multi-modal time series prediction with llm-in-the-loop. *arXiv preprint arXiv:2503.01013*, 2025.

[44] Yanyan Yue, Xingbo Zhang, Jiancheng Lv, and Bo Du. Ts2vec: Towards universal representation of time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8268–8276, 2022.

[45] Baoyu Jing, Si Zhang, Yada Zhu, Bin Peng, Kaiyu Guan, Andrew Margenot, and Hanghang Tong. Retrieval based time series forecasting. *arXiv preprint arXiv:2209.13525*, 2022.

[46] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

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

[48] Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International conference on learning representations*, 2021.

[49] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019.

[50] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yixin Dai, Jiawei Sun, Haofen Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2:1, 2023.

[51] DOC/NOAA/NESDIS/NCDC and National Climatic Data Center, NESDIS, NOAA, U.S. Department of Commerce. Storm Events Database, 2023. Accessed: February 21, 2025.

[52] Matthew J. Menne, Simon Noone, Nancy W. Casey, Robert H. Dunn, Shelley McNeill, Diana Kantor, Peter W. Thorne, Karen Orcutt, Sam Cunningham, and Nicholas Risavi. Global Historical Climatology Network-Hourly (GHCNh), 2023.

[53] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *International Conference on Learning Representations*, 2023.

[54] 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, 2024.

[55] Yuyang Wu, Haoran Zhang, Yong Liu, and Mingsheng Long. A decoder-only foundation model for time-series forecasting. *arXiv preprint arXiv:2405.12345*, 2024.

[56] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[57] Zach Nussbaum, John X Morris, Brandon Duderstadt, and Andriy Mulyar. Nomic embed: Training a reproducible long context text embedder. *arXiv preprint arXiv:2402.01613*, 2024.

[58] Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Yong Liu, Mingsheng Long, and Jianmin Wang. Deep time series models: A comprehensive survey and benchmark. 2024.

[59] Pavel Senin and Sergey Malinchik. Sax-vsm: Interpretable time series classification using sax and vector space model. In *2013 IEEE 13th international conference on data mining*, pages 1175–1180. IEEE, 2013.

Appendix

A Notations

The main notations used throughout this paper are summarized in Table 6.

Table 6: Summary of the notations used in this paper.

Notation	Description
$\mathbf{X} \in \mathbb{R}^{C \times T}$	Input multivariate time series with C channels and T time steps
P	Patch length for time series tokenization
$\hat{T}$	Number of temporal patches per channel ($\lfloor T/P \rfloor$)
C	Number of channels (variables)
T	Number of time steps in the original sequence
H	Forecasting horizon
d	Embedding dimension
L	Length of the flattened token sequence
X_i^{patch}	Embedding of the i-th patch token
$M \in \{0,1\}^{L \times L}$	Biased attention mask for the flatten token sequence
$\mathbf{H}^{\text{out}} \in \mathbb{R}^{L \times d}$	Output token embeddings from the Transformer encoder
$\mathbf{h}_{\texttt{[CLS]}} \in \mathbb{R}^d$	Embedding of the global [CLS] token
$\mathbf{H}_{\texttt{[CIT]}} \in \mathbb{R}^{C \times d}$	Embeddings of Channel Identity Tokens (CITs)
τ_{cxt}	Sample-level textual context associated with a time series
τ_c	Channel-level textual description for the c-th variable
$\mathbf{z}_{\text{cxt}} \in \mathbb{R}^d$	Semantic embedding of the sample-level text
$\mathbf{z}_c \in \mathbb{R}^d$	Semantic embedding of the channel-level text
$\mathcal{N}_{\text{cxt}}, \mathcal{N}_{\text{ch}}$	Sample-level and channel-level hard negative candidate sets

B Dataset Curation

We curate a new multimodal time series dataset in the weather domain by extending MTBench [6]. It is built from two primary sources:

- **Event reports** from the *NOAA Storm Events Database*[51], which contains detailed narratives of severe weather occurrences across the U.S.
- **Weather Time Series (TS) data** from the *NOAA Global Historical Climatology Network - Hourly (GHCN-h)*[52], covering multiple meteorological variables.

When applying TRACE to our curated dataset, the sample-level context is event report, while the channel-level description is synthetically generated by LLMs.

B.1 Station and Event Selection

We begin by selecting over 100 U.S. locations frequently affected by severe weather events and associated with long narrative reports. This yields approximately 5,000 event entries. For each event location, we identify nearby GHCN-h weather stations and extract multivariate TS data anchored at the start time of each event.

B.2 Time Series Sampling

Each event is treated as an anchor point to extract TS data at three resolutions:

- Hourly for 7 days
- Every 4 hours for 28 days

- Daily for 180 days

This results in approximately 15,000 TS samples from event-associated windows. To balance the dataset, we sample an additional 30,000 TS sequences from the same stations at random non-event times, ensuring no overlapping event narratives. To enhance weather diversity, we also sample 30,000 TS sequences from geographically distant stations without any event association, using randomly selected anchor times. See Figure 6. All time series instances contain seven channels: temperature, humidity, wind_u, wind_v, visibility, precipitation, and sky code. The curated weather dataset contains a total of 74,337 time series instances, and the lengths have a mean of 169.25 and a median of 168.0.

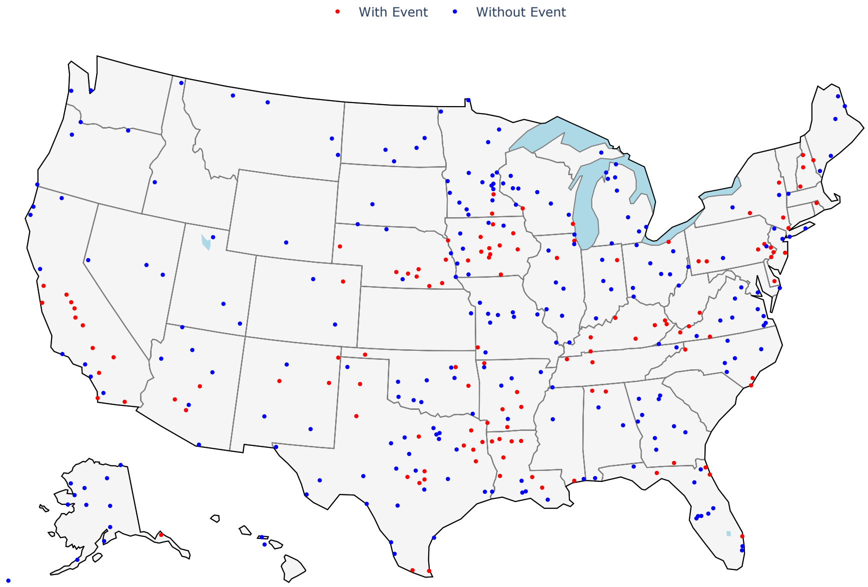


Figure 6: The red points are locations with event reports and the blue points are locations without event reports

Example: An Event Report of Debris Flow

"event type": "Debris Flow",
"state": "CALIFORNIA",
"cz name": "TULARE",
"begin datetime": "2021-12-14 13:14:00",
"end datetime": "2021-12-14 16:14:00",
"narrative": "A strong low pressure system dropped southeast out of the Gulf of Alaska on December 12 and intensified off the Pacific Northwest coast on December 13 pulling up some deep moisture which was pushed into central California during the afternoon. The precipitation intensified during the evening of December 13 through the morning of December 14 as the low carved out a deep upper trough which pushed across California during the afternoon of December 14. This system produced 2 to 4 inches of liquid precipitation over the Sierra Nevada from Sequoia National Park northward and 1 to 3 inches of liquid precipitation south of Sequoia Park. The precipitation fell mainly in the form of snow above 5500 feet and several high elevation SNOTELs estimated 2 to 4 feet of new snowfall. The snow level lowered to as low as 1500 feet during the evening of December 14 as the cooler airmass behind the system pushed into central California. Much of the San Joaquin Valley picked up between 1 to 2 inches of rainfall while the Kern County Mountains picked up between 0.75 and 1.5 inches of liquid precipitation. The Kern County Desert areas only picked up between a quarter and a half inch of rain at most locations due to rain shadowing. The storm produced

widespread minor nuisance flooding in the San Joaquin Valley and Sierra foothills with a few rock slides noticed. Several roads were closed as a precaution and chain restrictions were implemented on some roads in the Sierra Nevada. The storm also produced strong winds over the West Side Hills as well as in the Grapevine and Tehachapi areas in Kern County. Several stations in these areas measured wind gusts exceeding 50 mph with a few locations near the Grapevine measuring brief gusts exceeding 70 mph. California Highway Patrol reported mud, rock and dirt covering most of North Plano St. near Lynch Dr.",

B.3 Synthetic Description Generation

We use ChatGPT to generate channel-level textual descriptions for selected TS samples, where all TS samples linked to event reports are included. We also randomly select 50% of TS samples from both the non-event windows at event-associated stations and the non-event-associated stations to generate channel-level descriptions for event-label balance. The generated descriptions follow the style of TimeCap[7], but each is additionally annotated with one or more keywords selected from the set as auxiliary information: {Clear, Cloudy, Rainy, Snowy, Windy, Foggy, Hot, Cold, Humid, Stormy}. We use a consistent meta-prompt to elicit both descriptive and label-aligned outputs. A full example of the meta-prompt and a generated description is provided in B.4.

B.4 Prompt for Weather Description Generation and an example synthetic description

Weather Summary Prompt

You are a daily weather reporter, asked to summarize the past seven days of hourly weather (or the past 28 days of 4-hourly weather, or the past 6 months of daily weather, depending on the selected mode).
It will be multichannel with `temperature`, `precipitation`, `relative_humidity`, `visibility`, `wind_u`, and `wind_v` aspects. Summarize these channels and label the overall weather with one or more keywords from the set:
{Clear, Cloudy, Rainy, Snowy, Windy, Foggy, Hot, Cold, Humid, Stormy}.
You are **not** expected to report each time point individually. Instead, analyze the entire period as a whole. Additionally, for `temperature`, `precipitation`, and `relative_humidity`, identify any noticeable trends, potential periodicities (*e.g.,* daily or weekly patterns), overall volatility, and any clear outliers that stand out. You do not need to analyze other channels for these advanced statistics.
The input includes:

- Location
- Date
- Temperature time series
- Precipitation time series
- Relative humidity time series
- Visibility time series
- Wind_u time series
- Wind_v time series
- Sky cover codes

Sky cover codes are interpreted as follows:

Code	Meaning	Sky Fraction Covered
00	CLR (Clear)	0/8 or 0%
01	FEW	1/8 (12%)
02	FEW	2/8 - 3/8 (25%-37%)
03	SCT (Scattered)	4/8 (50%)
04	SCT	5/8 (62%)
05	BKN (Broken)	6/8 (75%)
06	BKN	7/8 - 8/8 (87%-100%)
07	BKN	∼9/10
08	OVC (Overcast)	10/10 (100%)
09	VV (Vertical Visibility)	Sky obscured
10	X (Unknown)	Partially obscured

Please summarize the data using the following format:

- **Date:** {sentence of date}
- **Location:** {sentence of location}
- **Temperature:** {sentence of temperature}
- **Precipitation:** {sentence of precipitation}
- **Relative Humidity:** {sentence of relative humidity}
- **Visibility:** {sentence of visibility}
- **Wind_V:** {sentence of wind_v}
- **Wind_U:** {sentence of wind_u}
- **Sky Cover:** {sentence of sky cover}
- **Keywords:** {list of keywords from label set}

No additional explanation or commentary should be included in the output.

Example: Generated Weather Summary

Date: The past 28 days from January 30, 2021, to February 26, 2021.
Location: The weather data is from Pike, Kentucky.
Temperature: The temperature ranged from a low of -10.07 °C to a high of 20.25 °C, with noticeable fluctuations and a general upward trend towards the end of the period, indicating warming conditions.
Precipitation: There was no recorded precipitation throughout the 28 days, indicating dry weather.
Relative Humidity: Relative humidity varied significantly, peaking at 100% on multiple occasions, with a general trend of higher humidity levels during the earlier part of the period and lower levels towards the end.
Visibility: Visibility remained consistently high at 14.58 km throughout the reporting period.
Wind_V: Wind velocity showed variability, with occasional gusts and a general trend of calm conditions.
Wind_U: Wind direction fluctuated, with both positive and negative values indicating changes in wind patterns.
Sky Cover: The sky was consistently clear with no significant cloud cover reported.
Keywords: [Clear, Cold, Humid]

B.5 Dataset Details

Our curated weather dataset contains a total of 74,337 time series instances. We allocate 9,561 of these exclusively for the forecasting task, ensuring this subset is disjoint from the pretraining and classification data to avoid any potential label leakage or information overlap. The classification task is formulated as multi-class event prediction, where each time series instance is annotated by the NOAA System with a corresponding weather event type from nine common severe weather events, and one special category for non-events. The event labels are as follows: *Lightning (0), Debris*

Table 7: Dataset size for each task.

Dataset Type	**Train**	**Test**	**Val**	**Total**
Newly Curated Weather Dataset				
Forecasting (H=7)	6,690	957	1,914	9,561
Pretraining & Classification	45,339	6,484	12,953	64,776
Public Dataset from TimeMMD [5]				
Health (H=12)	929	266	129	1,324
Energy (H=12)	992	284	138	1,414
Environment (H=48)	7,628	2,173	1,064	10,865

Flow (1), Flash Flood (2), Heavy Rain (3), Tornado (4), Funnel Cloud (5), Hail (6), Flood (7), Thunderstorm Wind (8). Instances that do not correspond to any specific event are labeled as `None`. This setup ensures the model learns to distinguish between distinct event types while being robust to trivial (non-event) data. We follow the original split to create train/test/val set for TimeMMD forecasting tasks [5].

C Alignment Objective: Full Formulation

To fully capture the structured alignment between multivariate time series and text, we employ a dual-level contrastive learning strategy, consisting of sample-level and channel-level hard negative mining.

C.1 Hard Negative Candidate Sets

Given a time series instance i with C channels, we define the following negative sets:

Sample-level negative sets. For aligning the global [CLS] embedding $\mathbf{h}^{(i)}_{\texttt{[CLS]}}$ of instance i with its corresponding sample-level textual embedding $\mathbf{z}^{(i)}_{\text{cxt}}$, we mine hard negatives from other samples in the batch. Specifically:

$$\mathcal{N}^{(i)}_{\text{cxt}} = \text{Top}_K \left\{ \text{sim}(\mathbf{h}^{(i)}_{\texttt{[CLS]}}, \mathbf{z}^{(j)}_{\text{cxt}}) \mid j \neq i \right\}, \tag{3}$$

and symmetrically,

$$\mathcal{N}^{(i,\text{text})}_{\text{cxt}} = \text{Top}_K \left\{ \text{sim}(\mathbf{z}^{(i)}_{\text{cxt}}, \mathbf{h}^{(j)}_{\texttt{[CLS]}}) \mid j \neq i \right\}. \tag{4}$$

Channel-level negative sets. To align each channel-specific CIT embedding $\mathbf{h}^{(i)}_c$ with its corresponding channel-level text embedding $\mathbf{z}^{(i)}_c$, we mine two types of distractors:

- *Intra-instance negatives:* embeddings from other channels within the same instance, i.e., $\mathbf{z}^{(i)}_{c'}$ where $c' \neq c$;
- *Inter-instance negatives:* same-indexed channel embeddings across different instances, i.e., $\mathbf{z}^{(j)}_c$ where $j \neq i$.

Formally, the channel-level negative set is defined as:

$$\mathcal{N}^{(i,c)}_{\text{ch}} = \text{Top}_K \left\{ \text{sim}(\mathbf{h}^{(i)}_c, \mathbf{z}^{(j)}_{c'}) \mid c' \neq c \text{ or } j \neq i \right\}, \tag{5}$$

and similarly in the reverse direction:

$$\mathcal{N}^{(i,c,\text{text})}_{\text{ch}} = \text{Top}_K \left\{ \text{sim}(\mathbf{z}^{(i)}_c, \mathbf{h}^{(j)}_{c'}) \mid c' \neq c \text{ or } j \neq i \right\}. \tag{6}$$

C.2 Contrastive Alignment Loss

We adopt a bidirectional InfoNCE loss at both the sample and channel levels. For each alignment direction, the objective maximizes the similarity between the positive pair and minimizes similarity with hard negatives.

Sample-level loss.

$$\mathcal{L}_{\text{global}}^{\text{text}\to\text{ts}} = -\log \frac{\exp(\text{sim}(\mathbf{z}_{\text{cxt}}^{(i)}, \mathbf{h}_{\mathtt{[CLS]}}^{(i)})/\tau)}{\sum_{j\in\{i\}\cup\mathcal{N}_{\text{cxt}}^{(i,\text{text})}} \exp(\text{sim}(\mathbf{z}_{\text{cxt}}^{(i)}, \mathbf{h}_{\mathtt{[CLS]}}^{(j)})/\tau)} \tag{7}$$

$$\mathcal{L}_{\text{global}}^{\text{ts}\to\text{text}} = -\log \frac{\exp(\text{sim}(\mathbf{h}_{\mathtt{[CLS]}}^{(i)}, \mathbf{z}_{\text{cxt}}^{(i)})/\tau)}{\sum_{j\in\{i\}\cup\mathcal{N}_{\text{cxt}}^{(i)}} \exp(\text{sim}(\mathbf{h}_{\mathtt{[CLS]}}^{(i)}, \mathbf{z}_{\text{cxt}}^{(j)})/\tau)} \tag{8}$$

Channel-level loss.

$$\mathcal{L}_{\text{channel}}^{\text{text}\to\text{ts}} = \frac{1}{C}\sum_{c=1}^{C} -\log \frac{\exp(\text{sim}(\mathbf{z}_c^{(i)}, \mathbf{h}_c^{(i)})/\tau)}{\sum_{(j,c')\in\{(i,c)\}\cup\mathcal{N}_{\text{ch}}^{(i,c,\text{text})}} \exp(\text{sim}(\mathbf{z}_c^{(i)}, \mathbf{h}_{c'}^{(j)})/\tau)} \tag{9}$$

$$\mathcal{L}_{\text{channel}}^{\text{ts}\to\text{text}} = \frac{1}{C}\sum_{c=1}^{C} -\log \frac{\exp(\text{sim}(\mathbf{h}_c^{(i)}, \mathbf{z}_c^{(i)})/\tau)}{\sum_{(j,c')\in\{(i,c)\}\cup\mathcal{N}_{\text{ch}}^{(i,c)}} \exp(\text{sim}(\mathbf{h}_c^{(i)}, \mathbf{z}_{c'}^{(j)})/\tau)} \tag{10}$$

C.3 Total Loss Objective

The total alignment loss is the average of both sample-level and channel-level contrastive losses:

$$\mathcal{L}_{\text{align}} = \frac{1}{2}\left(\mathcal{L}_{\text{global}}^{\text{text}\to\text{ts}} + \mathcal{L}_{\text{global}}^{\text{ts}\to\text{text}}\right) + \lambda_{\text{ch}} \cdot \frac{1}{2}\left(\mathcal{L}_{\text{channel}}^{\text{text}\to\text{ts}} + \mathcal{L}_{\text{channel}}^{\text{ts}\to\text{text}}\right), \tag{11}$$

where τ is the temperature hyperparameter, and λ_{ch} is a hyperparameter, controlling the contribution of channel-level alignment. We set $\lambda_{\text{ch}} = 1.0$ as default in experiments.

D Experiments

D.1 Baselines

D.1.1 Full-shot Time Series Models

DLinear (Decomposition-Linear) [29]is a lightweight time-series forecasting model that decomposes the input into trend and seasonal components, and applies simple linear layers to each component separately. Despite its simplicity, DLinear has demonstrated strong performance on both long- and short-term forecasting tasks by effectively capturing linear temporal patterns without relying on complex neural architectures.

PatchTST[22] reformulates time-series forecasting as a patch-based sequence modeling problem. It splits the input time series into non-overlapping patches and applies a Transformer encoder to model inter-patch dependencies. The design removes positional encoding and avoids decoder layers, making the model more suitable for forecasting tasks while benefiting from the global receptive field of Transformers.

iTransformer[24] (Instance-aware Transformer) extends Transformer-based forecasting by modeling instance-wise variations. It introduces a shared backbone Transformer and an instance-specific modulation mechanism, enabling the model to better adapt to diverse temporal dynamics across different time-series samples. This design improves generalization and robustness, particularly for multivariate forecasting.

TimesNet[53] proposes a novel temporal block that captures multi-frequency patterns in time-series data using learnable convolutions in the frequency domain. By combining time and frequency-domain features, TimesNet achieves strong performance across a variety of datasets. It is particularly effective at modeling both short-term and long-term temporal dependencies.

TimeMixer[54] employs a structured state-space-inspired architecture where time mixing and channel mixing operations alternate. It replaces self-attention with parameter-efficient mixing blocks that blend information across the temporal and feature dimensions. TimeMixer is designed for scalable forecasting and excels in low-resource regimes due to its compact architecture and efficient training.

FSCA[37] introduces a new paradigm that aligns time series (TS) with a linguistic component in the language environments familiar to LLMs to enable LLMs to contextualize and comprehend TS data, thereby activating their capabilities. FSCA uses a Dual-Scale Context-Alignment Graph Neural Networks (DSCA-GNNs) framework to achieve both structural and logical alignment, demonstrate good performance in few-shot and zero-shot settings.

D.1.2 Time Series Foundation Model

Table 8: Comparison of time-series foundation models.

Method	**Chronos**	**Time-MoE**	**TimesFM**	**Moirai**	**Moment**	**Timer-XL**
Architecture	Encoder-Decoder	Decoder-Only	Decoder-Only	Encoder-Only	Encoder-Only	Decoder-only
(Max) Model Size	710M	2.4B	200M	311M	385M	84M
Input Token	Point	Point	Patch	Patch	Patch	Patch
Max Length	512	4096	512	5000	512	1024
FFN	Dense	Sparse	Dense	Dense	Dense	Dense
Cross-channel	✗	✗	✗	✓	✗	✓

We test several recent time-series foundation models that have been pretrained on large-scale datasets from relevant domains, including weather, healthcare, energy, and environment. These include Chronos [32], Time-MoE [10], TimesFM [55], Moirai [34], Moment [8], and Timer-XL [9], which offer strong generalization through large-scale pretraining. A comparison is given in Table 8. To evaluate retrieval-augmented performance on diverse real-world domains, we integrate our retriever with three publicly available time-series foundation models: Time-MoE, Timer-XL, and Moment, which are selected based on the availability of stable, open-source implementations that support customization and downstream fine-tuning. We leave the adaptation of our retriever to additional proprietary or closed-source foundation models, as well as its integration into unified pretraining pipelines, for future work.

Comparison of `TRACE` with Time-series Foundation Models. It is important to note that our model is not itself a cross-domain foundation model, but rather a modular encoder-based retriever capable of enhancing such models. Architecturally, our model adopts an encoder-only design with flexible point- and patch-based tokenization, supports input sequences exceeding 2,048 tokens, and enables effective cross-channel interactions through channel-biased attention mechanisms.

D.2 Experiment Configurations

All models are implemented in PyTorch and trained on NVIDIA A100 40GB GPUs. For most time series models, we adopt the implementation from TSLib[58][2]. The sequence length is fixed at 96 for both prediction horizons of 7 and 24. We use mean squared error (MSE) as the loss function for forecasting tasks, and accuracy for classification. Forecasting models are trained for 10 epochs, while classification models are trained for up to 150 epochs with early stopping. We follow the official code to implement other baselines [37][3]. All other hyperparameters follow the default settings in TSLib, except for those explicitly tuned to achieve the best performance, as reported in Tables 9. For our model, the initial learning rate is tuned from $\{10^{-4}, 10^{-3}\}$. The number of attention layers is tuned from $\{6, 12\}$, and the hidden dimension is from $\{384, 768\}$ with the number of heads in $\{6, 12\}$.

[2] `https://github.com/thuml/Time-Series-Library`

[3] `https://github.com/tokaka22/ICLR25-FSCA`

Table 9: Best hyperparameters per model

Model Name	Learning Rate	Encoder Layers	Hidden Dimension
DLinear	0.0010	2	32
PatchTST	0.0050	4	64
TimeMixer	0.0100	4	64
TimesNet	0.0010	4	64
iTransformer	0.0100	4	64
FSCA	0.0001	4	256

D.3 Embedding Visualization

Figure 7 presents the cosine similarity matrix between text and time series embeddings across the test set. The diagonal dominance indicates that TRACE successfully aligns each time series with its corresponding textual description, suggesting strong one-to-one semantic matching in the shared embedding space. Off-diagonal similarities remain low, demonstrating the model's ability to distinguish unrelated instances. Figure 8 visualizes the joint embedding space using UMAP. Each color represents a distinct event category, where circles (○) denote time series instances and crosses (×) denote their corresponding textual descriptions. A line connects each text–time series pair. We observe clear clustering by event type, with paired modalities positioned closely in the embedding space. Notably, for some events (*e.g.,* "Flood" and "Debris Flow"), clusters partially overlap, reflecting shared underlying dynamics. The tight alignment between paired points validates the effectiveness of our dual-level alignment strategy, and the modality-mixing within clusters suggests successful fusion of structured and unstructured signals.

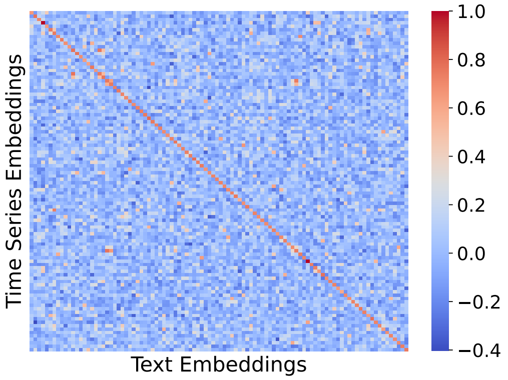


Figure 7: Cosine Similarity Matrix Between Text and Time Series Embeddings.

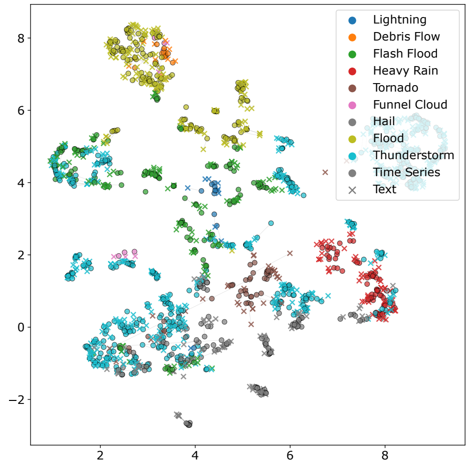


Figure 8: Umap Visualization of Aligned Text and Time Series Embeddings.

D.4 Classification Task

Table 10 reports the classification accuracy and F1 scores of different size variants of time-series foundation models on the weather event classification task. We observe that a larger model size does not necessarily lead to better performance. For example, Moment's base model achieves a higher F1 score than the large model despite a lower accuracy. In contrast, Chronos exhibits more stable performance across scales, with the tiny and mini variants achieving the best F1 scores, outperforming even the larger variants. These results suggest that, in domain-specific classification tasks with relatively limited supervision, scaling up foundation models may not always be beneficial, and smaller models can offer a better balance between accuracy and efficiency.

Table 10: Weather Event Classification Accuracy and F1 Score (%).

Model	**Size**	**Accuracy**	**F1**
Time-MoE	small	56.27	16.56
	large	59.09	19.74
Moment	base	65.43	28.29
	large	64.94	26.35
Chronos	tiny	74.79	40.21
	mini	73.89	37.98
	small	71.07	35.39
	base	71.42	36.40
	large	71.97	36.30

D.5 RAG Setting

In our retrieval-augmented generation (RAG) framework, given a query time series $\mathbf{X}_q$, we compute its `[CLS]` token embedding as $\mathbf{h}_q \in \mathbb{R}^d$ using the frozen encoder from `TRACE`. Based on cosine similarity, we retrieve the top-R most relevant multimodal pairs $(\mathbf{X}^i, \tau^i_{\text{cxt}})^R_{i=1}$ from the corpus, where $\mathbf{X}^i$ is a historical multivariate time series and τ^i_{cxt} is the associated sample-level context. Each retrieved pair is transformed into a soft prompt vector using a trainable linear projection layer. Specifically, the time series component is encoded to $\mathbf{h}^{(i)}_{\text{ts}} \in \mathbb{R}^d$, and the textual context τ^i_{cxt} is encoded to $\mathbf{z}\text{cxt}^{(i)} \in \mathbb{R}^d$ using a frozen SentenceTransformer, followed by a shared projection. For the *TS+Text* setting, we concatenate each pair as $\mathbf{p}^{(i)} = [h^{(i)}_{\text{ts}}; \mathbf{z}^{(i)}_{\text{cxt}}] \in \mathbb{R}^{2d}$, and stack all R vectors to form the final prompt:

$$\mathbf{P} = \texttt{Proj}\left([\mathbf{p}^{(1)}; \cdots ; \mathbf{p}^{(R)}]\right) \in \mathbb{R}^{d_f},$$

where `Proj` is a feedforward layer mapping from $\mathbb{R}^{2Rd} \to \mathbb{R}^{d_f}$, and d_f is the hidden dimension of the downstream time series foundation model. For the *TS-only* setting, we omit the text component and instead concatenate $[h^{(1)}_{\text{ts}}; \cdots ; h^{(R)}_{\text{ts}}] \in \mathbb{R}^{Rd}$ and project into $\mathbb{R}^{d_f}$ accordingly.

This dense prompt $\mathbf{P}$ is prepended to the query sequence during inference. For decoder-only models (*e.g.,* Timer-XL, Time-MoE), $\mathbf{P}$ is appended to the autoregressive context at each decoding step. For encoder-only models (*e.g.,* Moment, `TRACE`), $\mathbf{P}$ is inserted as a prefix to the encoder input, *i.e.,*

$$\hat{y} = \texttt{Head}([\mathbf{P}|\mathbf{H}_q]),$$

where $\mathbf{H}_q \in \mathbb{R}^{L\times d_f}$ is the encoded query and `Head` is a forecasting head trained from scratch. In all configurations, only `Proj` and `Head` are updated during training in RAG framework, while the backbone foundation model remains frozen.

D.6 Standalone Time Series Encoder

To evaluate the classification capabilities of time series foundation models, we finetune a multi-layer perceptron (MLP) classifier on top of each model's final output representation, as most existing time series foundation models do not support classification task by design, except Moment [8], The MLP consists of four hidden layers with sizes $[256, 128, 64, 32]$, followed by a softmax output layer corresponding to 9 weather event categories. This architecture was selected based on empirical tuning for optimal performance on our classification task. We include all available variants from four foundation model families: Time-MoE, Timer-XL, Moment, and Chronos. All backbone parameters of the time series foundation models are fully activated and updated during training to ensure consistency and fair evaluation. Each model is finetuned for 100 epochs using the Adam optimizer. The training batch size is set to 256 for small and mid-sized variants, and reduced to 128 for larger models to accommodate memory constraints.

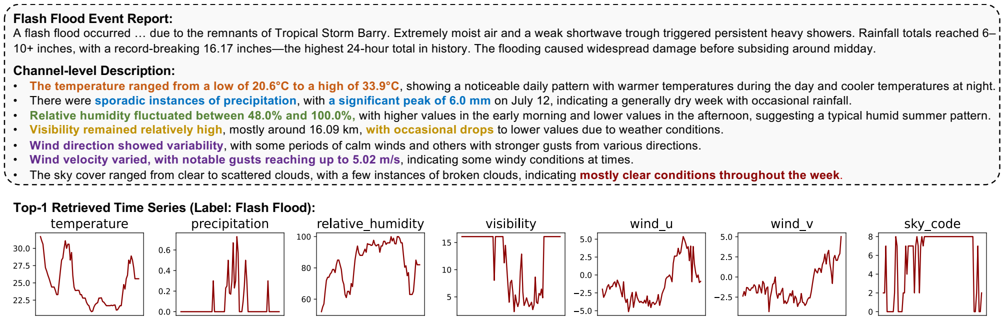
For full-shot time series models, we train them from scratch using a unified training and evaluation protocol with time series foundation models. Results are shown in Table 3 and Table 10.

D.7 Empirical Case Study

Figure 9 illustrates the capability to align detailed textual context with corresponding multivariate time series. The retrieval pool is constructed by excluding the query's paired time series instance. This setup ensures that retrieved results are non-trivial and reflect the model's ability to identify semantically similar yet distinct examples. `TRACE` leverages both high-level and fine-grained semantic cues to retrieve the most relevant time series from the curated candidate pool. The top-1 retrieved sequence closely reflects key patterns in the query text, which can serve as a valuable reference for downstream forecasting, scenario simulation, or contextual explanation.

D.8 Timeseries-to-Timeseries Retrieval

To assess the effectiveness of our model in time series retrieval, we conduct a TS-to-TS retrieval task where each query is matched against all other time series to identify the most semantically similar ones. The evaluation is performed using the label matching metrics (Sec. 4.2), including Precision@1, Precision@5, and Mean Reciprocal Rank (MRR), alongside query time as a proxy for computational efficiency.



Flash Flood Event Report:
A flash flood occurred … due to the remnants of Tropical Storm Barry. Extremely moist air and a weak shortwave trough triggered persistent heavy showers. Rainfall totals reached 6–10+ inches, with a record-breaking 16.17 inches—the highest 24-hour total in history. The flooding caused widespread damage before subsiding around midday.

Channel-level Description:

- **The temperature ranged from a low of 20.6°C to a high of 33.9°C**, showing a noticeable daily pattern with warmer temperatures during the day and cooler temperatures at night.
- There were **sporadic instances of precipitation**, with **a significant peak of 6.0 mm** on July 12, indicating a generally dry week with occasional rainfall.
- **Relative humidity fluctuated between 48.0% and 100.0%,** with higher values in the early morning and lower values in the afternoon, suggesting a typical humid summer pattern.
- **Visibility remained relatively high**, mostly around 16.09 km, **with occasional drops** to lower values due to weather conditions.
- **Wind direction showed variability**, with some periods of calm winds and others with stronger gusts from various directions.
- **Wind velocity varied, with notable gusts reaching up to 5.02 m/s**, indicating some windy conditions at times.
- The sky cover ranged from clear to scattered clouds, with a few instances of broken clouds, indicating **mostly clear conditions throughout the week**.

Top-1 Retrieved Time Series (Label: Flash Flood):

Figure 9: A case study of text-to-timeseries retrieval of flash flood-related time series. The key textual cues are highlighted in color for clarity.

Table 11: TS-to-TS Retrieval performance comparison. Evaluation is conducted over 1000 randomly sampled time series queries.

Method	P@1	P@5	MRR	Time (s)
ED	0.548	0.762	0.644	0.083
DTW	0.380	0.770	0.543	2273.93
SAX-VSM	0.551	0.769	0.649	0.343
CTSR	0.682	0.893	0.802	0.057
TRACE	**0.900**	**0.986**	**0.938**	**0.045**

Baseline Setup. We compare TRACE, against several representative time series retrieval methods. Euclidean Distance (ED) serves as a simple statistical baseline based on mean-pooled raw time series. Dynamic Time Warping (DTW), a classic elastic matching method, evaluates similarity by aligning sequences with potential shifts, but at significant computational cost. SAX-VSM [59] leverages symbolic aggregation and vector space modeling to convert time series into symbolic representations for efficient textual retrieval. CTSR [11] refers to a learned baseline that uses contextual metadata to enhance retrieval.

Analysis. The results shown in Table 11 demonstrate that TRACE substantially outperforms all baselines across accuracy metrics while maintaining the lowest retrieval latency. Notably, despite the design simplicity of SAX-VSM and its moderate performance gains over raw ED, it fails to capture deep temporal or semantic patterns. CTSR, while benefiting from structured cues, struggles to generalize as effectively in purely time-series scenarios. The results suggest that learned time-series representations, when equipped with task-driven objectives and textual alignment-aware training, provide not only superior retrieval quality but also enable scalable and efficient retrieval pipelines. The combination of semantic precision and runtime efficiency highlights the efficacy of TRACE for real-world applications where fast and accurate time series matching is critical.

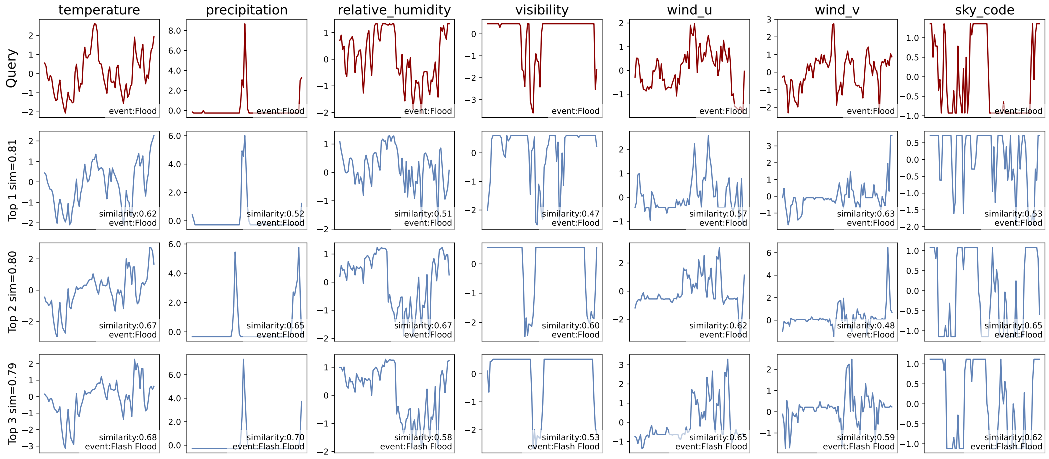


Figure 10: Visualization of Timeseries-to-Timeseries Retrieval by TRACE

TS-to-TS Case Study. Figure 10 illustrates a case study of TS-to-TS retrieval using `TRACE`. Given a query time series (top row) labeled as `Flood`, the system retrieves the top-3 most similar samples from the corpus based on embedding similarity in the shared representation space. The similarity score for each retrieved sample is shown on the left, with per-channel similarity values annotated below each plot. We observe that all retrieved samples have high overall similarity scores (approximately 0.79–0.81), reflecting strong semantic alignment. The top-2 retrievals are also labeled as `Flood`, while the third belongs to a semantically related event, `Flash Flood`, suggesting that `TRACE` is capable of retrieving contextually relevant samples even across closely related labels. Notably, `TRACE` enables fine-grained channel-level similarity assessment by leveraging its Channel Identity Tokens (CIT), which allow independent embedding of channel-specific signals.

However, we also find that high similarity in individual channels (*e.g.,* temperature or precipitation) does not always guarantee high overall semantic alignment. For instance, the first retrieval shows moderate similarity across channels but still achieves a high overall semantic score. This highlights the benefit of `TRACE`'s structured aggregation over all channels to capture global semantics and reveal the most semantically dominant channels that contribute most to the retrieval relevance. This capability enables `TRACE` to go beyond surface-level similarity, retrieving samples that share latent event signatures rather than merely matching patterns across all channels uniformly.

D.9 Complexity and Efficiency

D.9.1 Computational Complexity

We analyze the computational complexity of the main components in `TRACE`, including the encoder stage, the dual-level contrastive alignment, and the retrieval-augmented generation (RAG) setup.

1. Encoder Pre-training Complexity. Let $X \in \mathbb{R}^{C\times T}$ be the input multivariate time series with C channels and T time steps. The sequence is tokenized into $\hat{T} = \lfloor T/P \rfloor$ patches per channel, each projected to a d-dimensional embedding. The total token length after flattening is $1 + C(\hat{T} + 1)$. This includes one global `[CLS]` token, one `[CIT]` token per channel, and $\hat{T}$ patch tokens per channel. The encoder is a N-layer Transformer with multi-head channel-biased attention. The complexity per attention layer is $\mathcal{O}(L^2 d) = \mathcal{O}(C^2\hat{T}^2 d)$. Note that channel-biased attention applies a sparse mask $M \in \{0, 1\}^{L\times L}$ to restrict certain attention to within-channel interactions, which effectively reduces the constant factors in practice but not the asymptotic complexity.

2. Dual-level Contrastive Alignment. Let B be the batch size. For each time series, the alignment stage computes:

- Sample-level similarity: $\mathcal{O}(B^2 d)$ for all $\mathbf{h}_{\texttt{[CLS]}}$–$\mathbf{z}_{\text{cxt}}$ pairs.
- Channel-level similarity: For C channels and B instances, total cost is $\mathcal{O}(B^2C^2d)$ for $\mathbf{h}_c$–$\mathbf{z}_c$ pairs.
- Negative mining selects top-R hardest negatives per instance and per channel, which costs $\mathcal{O}(B \log R + BC \log R)$, and is negligible compared to similarity computation.

3. Retrieval-Augmented Generation. During inference, retrieval selects top-R neighbors for a query based on cosine similarity:

- Retrieval cost: $\mathcal{O}(Rd)$ using approximate methods (*e.g.,* FAISS) from a database.
- Prompt generation: if soft prompt dimension is d_f, and each retrieved pair contributes d-dim vector, this yields a projection cost of $\mathcal{O}(Rdd_f)$.
- The forecasting model remains frozen; only the soft prompt (a single vector of shape $[1, d_f]$) is appended, incurring no extra Transformer-layer cost.

Summary. Pre-training (Transformer encoder) yields $\mathcal{O}(L^2d)$ per layer. Alignment yields $\mathcal{O}(B^2d + B^2C^2d)$ and RAG inference yields $\mathcal{O}(Rdd_f)$ for retrieval and projection.

D.9.2 Empirical Runtime

We report the model size and empirical runtime of `TRACE` and other baselines in Table 12, including FSCA [37], which is the second-best train-from-scratch time series model, and time series foundation models with the availability of open-source implementations. `TRACE` activates only 0.12M parameters during finetuning with a lightweight linear head, which is nearly 200× fewer than FSCA and over 700× fewer than Time-MoE$_{\text{small}}$. This lightweight design results in substantially faster training and inference speed. Compared to Moment, TRACE achieves faster training time with significantly fewer trainable parameters and better performance, which can be attributed to its multichannel modeling with channel-biased attention. While slightly slower than Timer-XL, which is a decoder-only model with causal attention, TRACE offers an acceptable overhead given its significantly stronger retrieval performance and the high quality of embeddings it produces for cross-modal and TS-to-TS retrieval. It is worth noting that for Timer-XL and Time-MoE, despite their strong generalizability, parameter-efficient finetuning strategies are relatively underexplored, as all model parameters must be activated and updated during finetuning for reasonable performance in domain-specific tasks.

Table 12: Comparisons of model efficiency. Activated Params indicates the number of parameters activated during finetuning for 7-step forecasting on the weather dataset. Training and inference time are seconds per epoch on the forecasting dataset. Device is a single A100 40GB GPU.

	Total Params	**Activated Params**	**Training Time**	**Inference Time**
FSCA	82.35M	22.68M	1249.701	1.589
TRACE	10.78M	0.12M	6.054	0.955
Moment$_{\text{base}}$	109.87M	0.24M	11.706	1.691
Timer-XL$_{\text{base}}$	84.44M	84.44M	3.392	0.685
Time-MoE$_{\text{small}}$	113.49M	113.49M	106.308	15.545

E Discussion

Limitation. While TRACE demonstrates strong performance in multimodal retrieval and retrieval-augmented forecasting, it currently assumes the availability of aligned time series–text pairs during training. In some domains, such alignment may be noisy or incomplete. Additionally, although channel-level alignment improves interpretability and fine-grained matching, it introduces a modest increase in computational overhead during training. We believe these trade-offs are justified by the performance gains but acknowledge that further optimization may enhance scalability.

Future Work. In future work, we plan to extend TRACE to support weakly supervised and semi-supervised settings, where textual context is partially missing or noisy. Another promising direction is integrating domain adaptation techniques to improve generalization across unseen domains and sensor modalities (*e.g.,* image, video). Moreover, exploring autoregressive generation conditioned on retrieved time series–text pairs may further enhance understanding tasks in temporal modeling.

Broader Impact. TRACE offers a general framework for cross-modal reasoning in time series applications, with potential benefits in domains such as healthcare monitoring, disaster forecasting, and industrial diagnostics. By improving retrieval and interpretation of structured temporal data, our approach may enhance decision support and model transparency. However, we encourage responsible deployment and emphasize the importance of auditing training data and retrieval outputs to avoid amplifying biases present in either modality.