ViTs: Teaching Machines to See Time Series Anomalies Like Human Experts

Zexin Wang*
Computer Network Information Center, Chinese Academy of Sciences
Beijing, China

Changhua Pei†
Yang Liu
Computer Network Information Center, Chinese Academy of Sciences
Beijing, China

Hengyue Jiang
Hangzhou Institute for Advanced Study, University of Chinese Academy of Sciences
Beijing, China

Quan Zhou
Haotian Si
Hang Cui
Computer Network Information Center, Chinese Academy of Sciences
Beijing, China

Jianhui Li‡
Gaogang Xie
Jingjing Li
Computer Network Information Center, Chinese Academy of Sciences
Beijing, China

Dan Pei
Tsinghua University
Beijing, China

Abstract

Web service administrators must ensure the stability of multiple systems by promptly detecting anomalies in Key Performance Indicators (KPIs). Achieving the goal of "train once, infer across scenarios" remains a fundamental challenge for time series anomaly detection models. Beyond improving zero-shot generalization, such models must also flexibly handle sequences of varying lengths during inference—ranging from one hour to one week—without retraining. Conventional approaches rely on sliding-window encoding and self-supervised learning, which restrict inference to fixed length inputs. Large Language Models (LLMs) have demonstrated remarkable zero-shot capabilities across general domains. However, when applied to time series data, they face inherent limitations due to context length. To address this issue, we propose **ViTs**, a Vision-Language Model (VLM)-based framework that converts time series curves into visual representations. By rescaling time series images, temporal dependencies are preserved while maintaining a consistent input size, thereby enabling efficient processing of arbitrarily long sequences without context constraints. Training VLMs for this purpose introduces unique challenges, primarily due to the scarcity of aligned time series image–text data. To overcome this, we employ an evolutionary algorithm to automatically generate thousands of high-quality image–text pairs and design a three-stage training pipeline consisting of: (1) time series knowledge injection, (2) anomaly detection enhancement, and (3) anomaly reasoning refinement. Extensive experiments demonstrate that ViTs substantially enhance the ability of VLMs to understand and detect anomalies in time series data. All datasets and code will be publicly released at: https://anonymous.4open.science/r/ViTs-C484/.

CCS Concepts

• **Mathematics of computing** → **Time series analysis**; • **Computing methodologies** → **Anomaly detection**.

Keywords

Time Series, Anomaly Detection, Vision Language Models

ACM Reference Format:
Zexin Wang, Changhua Pei, Yang Liu, Hengyue Jiang, Quan Zhou, Haotian Si, Hang Cui, Jianhui Li, Gaogang Xie, Jingjing Li, and Dan Pei. 2026. ViTs: Teaching Machines to See Time Series Anomalies Like Human Experts. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (WWW '26)*. ACM, New York, NY, USA, 13 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Rapid detection of anomalies in Key Performance Indicators (KPIs) and timely recovery from failures are essential for ensuring the reliability of web systems [17, 26, 30]. However, with the continuous expansion of web services, the number of time series requiring monitoring has grown substantially. Consequently, enabling a "train once, infer across scenarios" paradigm with zero-shot capability has become a critical challenge for Time Series Anomaly Detection (TSAD) models. Moreover, KPIs exhibit considerable heterogeneity (e.g., periodicity), which results in significant variations in the optimal detection window size. An effective model should therefore be able to adaptively handle time series of different lengths during inference. Unfortunately, existing deep learning-based TSAD methods [23, 26, 30, 31] are typically designed for fixed window sizes. Owing to their limited model capacity, these models must be re-trained or fine-tuned when applied to new scenarios, and thus lack the desired zero-shot generalization ability.

Having established the "train once, infer across scenarios" paradigm in the text domain, Large Language Models (LLMs) are expected to demonstrate comparable success in time series analysis, sparking growing research interest [10, 16, 21, 25]. Some studies have attempted to directly feed time series data into LLMs in textual form, leveraging either prompt engineering [15] or fine-tuning

*Also with University of Chinese Academy of Sciences.
†Corresponding author. Email: chpei@cnic.cn. Also with Hangzhou Institute for Advanced Study, University of Chinese Academy of Sciences.
‡Also with Nanjing University.

WWW '26, Dubai, United Arab Emirates
2026. ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

techniques [29] to construct Time Series LLMs (TS-LLM in Figure 1). However, representing time series as text introduces significant challenges: (i) extremely long context lengths, which hinder the inference of long time series, and (ii) unstable numerical token relationships, which often result in incoherent or inconsistent outputs. To address the challenges, several studies [29] have proposed dedicated time series encoders that are jointly trained with textual modalities to achieve cross-modal alignment (Patch-based Time Series Encoders LLMs, PTSE-LLMs, in Figure 1). These methods effectively mitigate the issue of context window overflow in LLMs when processing long time series. However, introducing a separate time series encoder inevitably imposes a fixed input length constraint, limiting the model's ability to generalize to sequences longer than those seen during training.

To address these problems, we draw inspiration from the analytical practices of human experts. We observe that, when determining whether a time series exhibits anomalies, experts typically do not inspect each data point in isolation as deep learning models often do. Instead, they visualize the time series curve on a dashboard and make holistic judgments by analyzing its overall patterns. Motivated by this insight, we propose a TSAD approach based on Vision Language Models (VLMs). By converting time series curves into images, our method effectively emulates the cognitive process of human experts. During downstream inference, time series of varying lengths are proportionally rescaled into fixed-size images, ensuring consistency with the training phase and maintaining stable inference performance. It is important to emphasize that time series image rescaling is fundamentally different from curve point sampling. Specifically, image rescaling only scales the indices of the time series points proportionally (e.g., $[0, 1, \ldots, 800] \rightarrow [0, 0.25, \ldots, 200]$) without altering the corresponding signal values. In contrast, direct downsampling of the raw time series modifies its intrinsic shape—sharp peaks, for instance, may become smoothed or flattened due to averaging over multiple points.

Although TSAD based on VLMs holds significant promise, directly applying existing general VLMs to time series analysis yields unsatisfactory results. The primary reason is that these models have not been pre-trained on large-scale datasets containing time-series–related images. In addition, they lack exposure to anomalous time series samples and corresponding analytical dual-modal text data, both of which are crucial for strengthening their capability in TSAD. To tackle these challenges, we propose **ViTs**, a TS-VLM specifically designed for TSAD tasks. To enhance data diversity, ViTs introduces a novel Seasonal-Trend decomposition using Loess (STL)-based approach [7] for periodic data generation and incorporates a wide range of anomaly injection strategies. In terms of training, we develop a three-stage chain-style fine-tuning framework, termed **Chain-of-TS**, which comprises: (1) *time series knowledge injection*, (2) *anomaly detection enhancement*, and (3) *anomaly reasoning refinement*.

The main contributions can be summarized as follows:

(1) We conduct a comprehensive exploration of different time series large models and demonstrate the effectiveness of VLMs. In addition, we investigate how various visualization approaches impact the performance of TS-VLMs.
(2) We propose **ViTs**, a novel TS-VLM specifically designed for TSAD. To this end, we introduce a three-stage fine-tuning strategy, termed **Chain-of-TS**, which leverages diverse data types and fine-tuning techniques to substantially enhance the capability and effectiveness of VLMs in TSAD.
(3) We design a new fixed length training and adaptive length inference paradigm. Time series image rescaling during inference enables ViTs to generalize effectively across time series of arbitrary lengths.
(4) Extensive experiments show that ViTs surpasses state-of-the-art (SOTA) VLMs by more than 20%, despite utilizing a relatively modest number of parameters.

2 Preliminaries

2.1 Problem Definition

Given a time series $X = \{x_1, x_2, \ldots, x_n\}$ collected at regular intervals, where x_i denotes the value of a specific metric at time step i, the task of TSAD is defined as follows:

At time t, a sliding window of length w is extracted, denoted as $X_t = \{x_{t-w}, \ldots, x_t\}$. The goal is to assign an anomaly score to each point within the window, resulting in a score sequence $S_t = \{s_{t-w}, \ldots, s_t\}$. These scores are then used to identify anomalous intervals $L = \{[start_1, end_1], [start_2, end_2], \ldots\}$ via thresholding methods. Alternatively, the TSAD task can also be formulated as directly detecting anomalous intervals from the raw time series X without explicitly computing point-wise anomaly scores.

2.2 Anomaly Types

Based on extensive observations of public time series datasets, this paper focuses primarily on periodic time series and categorizes time series anomalies into four types: spike anomaly, trend anomaly, level anomaly, and frequency anomaly. Details can be find in Section B and Figure 10.

3 Motivation

3.1 TS-LLM & TS-VLM & PTSE-LLM

In Section 1, we introduced three main approaches for constructing time series large models: TS-LLM, TS-VLM, and PTSE-LLM. To verify that TS-VLM is a simpler and more effective approach, we conducted experiments. Specifically, we compared the performance of TS-LLM and TS-VLM both before and after STF with respect to TSAD. The same fine-tuning data (10k synthetic QA datasets) was used for different methods, with evaluation performed using 2k synthetic QA datasets (consistent with the synthetic evaluation datasets described in Section 5.6). The models used were Qwen2.5-VL-7B-Instruct (with TS-LLM utilizing only its LLM backbone, and TS-VLM utilizing both the Vision Encoder and LLM backbone). For PTSE-LLM, we directly used the fine-tuned ChatTS. The results before and after fine-tuning are shown in Table 1.

It can be observed that before SFT, TS-LLM has almost no TSAD capability. This is because the time series in text format generates a large number of numeric tokens, and LLMs are not specifically trained on sequential numeric tokens. Consequently, LLMs are prone to issues such as repetitive output of numbers and mismatches between answers and questions, leading to poor TSAD

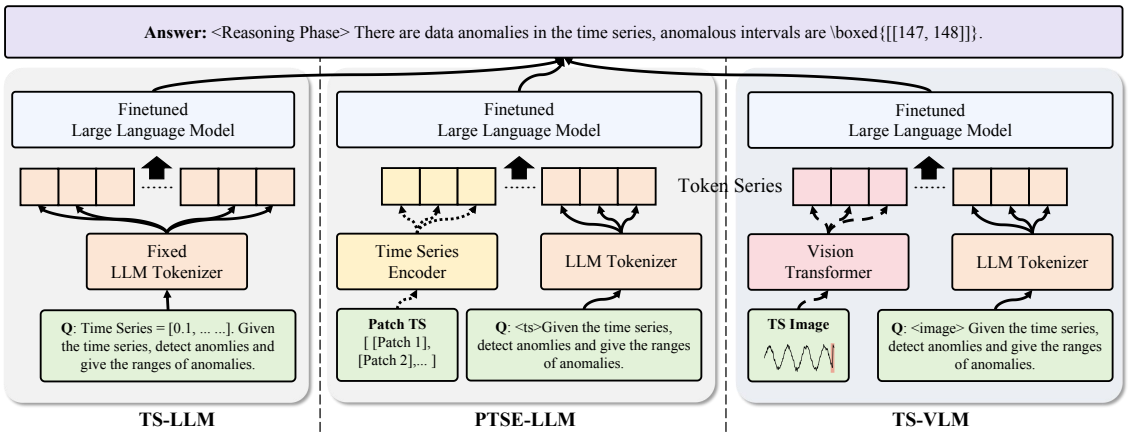


Figure 1: Comparison of TS-LLM, PTSE-LLM, and TS-VLM .

Table 1: The performance of different types of time series large models on TSAD. Both TS-LLM and TS-VLM utilize Qwen2.5-VL-7B-Instruct. P indicates precision, R indicates recall, and F1 represents the best F1 score after point adjustment. The best results are highlighted in bold, and the second best are underlined.

	Spike			Trend			Frequency			Level		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
TS-LLM	0.4496	0.0099	0.0194	0.3327	0.0316	0.0577	0.0000	0.0000	0.0000	0.1660	0.0086	0.0164
TS-VLM	0.5631	0.5490	0.5560	0.7019	0.0913	0.1616	0.8993	0.3035	0.4538	0.5733	0.5819	0.5776
PTSE-LLM(SFT)	0.1821	0.1432	0.1605	0.1954	0.1025	0.1348	0.1732	0.1520	0.1620	0.1610	0.1245	0.1406
TS-LLM(SFT)	0.9434	0.7385	0.8286	1.0000	0.0326	0.0631	0.9964	0.3840	0.5543	0.9465	0.8155	0.8761
TS-VLM(SFT)	0.9347	0.8203	**0.8737**	0.9365	0.1543	**0.2649**	0.9328	0.4813	**0.6350**	0.9460	0.8732	**0.9081**

performance. On the other hand, TS-VLM shows significant differences. Even a naive, non-fine-tuned Qwen2.5-VL-7B-Instruct can correctly detect a substantial portion of anomalies. This is because there are commonalities among different types of images, and training in general image understanding enhances the ability to understand time series images.

After SFT, it becomes apparent that PTSE-LLM performs the worst, which is understandable because training a time series encoder from scratch is quite challenging. TS-VLM outperforms TS-LLM across various anomaly types. This is easy to understand. The general image understanding capabilities of VLMs can significantly enhance their ability to interpret time series images, whereas time series in text form are difficult to understand and consume a large number of tokens. Therefore, using TS-VLM is a more suitable approach.

3.2 Comparison of Plot Type

For general tasks, providing LLMs with more detailed information typically enhances their performance [19, 27]. We hypothesize that this principle might also apply in VLMs. To test this assumption, we conducted a series of experiments. Considering that VLMs comprise both vision and textual modalities—and that the textual modality generally relies on pre-trained LLM backbones—our investigation centers on whether more detailed visual input can enhance TSAD performance.

The initial challenge is determining what kind of visual information should be fed into the VLM. The most common visualization method for time series data is the line chart, which intuitively illustrates temporal fluctuations. While widely adopted, recent research increasingly underscores the importance of frequency-domain information [26, 33, 38]. However, VLMs do not inherently extract such information from line charts. Therefore, we examine whether augmenting the input with frequency-domain visualizations—specifically Short-Time Fourier Transform (STFT) [6] and Wavelet Transform [37] plots—can improve TSAD performance (shown in Figure 10). The next question concerns how to incorporate this additional visual information. We explored two strategies: (1) Supplying multiple images to the VLM, including both line charts and frequency-domain plots; and (2) Merging the line chart with the frequency-domain plots into a single composite image with subplots. (Examples shown in Section C.)

The experimental results, depicted in Figure 2, demonstrate that supplying multiple separate images does not improve the F1 score without SFT. In contrast, combining the line and frequency-domain

Figure 2: Performance across different image types. LINE refers to a single line plot. LINE_STFT denotes a composite image containing two subplots—one showing the line plot and the other displaying the STFT. LINE_STFT_2 indicates two separate input images: one for the line plot and another for the STFT.

plots into a single image results in a noticeable performance improvement. We attribute this to the index alignment across subplots, which enables the model to correlate temporal and frequency-domain anomalies more effectively. Certain anomalies, which may be subtle in the time domain, become more apparent in the frequency domain.

To further enhance TSAD performance, we applied SFT using a large volume of synthetic data. However, contrary to expectations, the results deteriorated, with line charts alone delivering the best performance. To investigate this, we employed Qwen2.5-VL-7B-Instruct to generate image descriptions for various input formats (results shown in Section D). We observed that when exposed to a single line chart, the VLM produced detailed and relevant descriptions. However, for composite images with subplots, the VLM focused more on superficial elements, such as axis labels, which are not directly pertinent to TSAD.

Therefore, although using multiple image types may offer some marginal gains, the potential improvement appears limited. As such, this paper adopts single line charts as the visual input.

4 ViTs

The overall workflow of ViTs is illustrated in Figure 3.

4.1 STL-based Time Series Generator

To enhance the zero-shot TSAD capabilities of VLMs, it is crucial to train them with a substantial volume of time series. According to the STL theory, any time series can be decomposed into three components: seasonal, trend, and noise. Based on this principle, we propose an STL-based time series generator that synthesizes time series by generating these three components and then combining them.

4.1.1 Trend. When generating trends, we randomly choose from three types: increase, decrease, and keep steady. For increase and decrease, we design multiple modes, including exponential, linear, and others. The intensity of the trend is determined randomly.

Algorithm 1 Seasonal Generation

Require: $period\_type$, $amplitude\_series$, ts_length
Ensure: Generated seasonal data $data$

Initialize $data \leftarrow 0$
if $period\_type$ is long **then**
  $period \leftarrow random(10, \lfloor ts\_length/2 \rfloor)$
else
  $period \leftarrow random(\lfloor ts\_length/2 \rfloor, 3 \times ts\_length)$
end if
$base\_frequency \leftarrow 1/period$
$num\_harmonics \leftarrow random\_integer(1, 10)$
for $n = 1$ to $num\_harmonics$ **do**
  $phase \leftarrow random\_uniform(0, 2\pi)$
  $harmonic\_amplitude \leftarrow \frac{amplitude\_series}{n} \times \Big(1 + random\_uniform(0, 0.05) \cdot \sin\big(random\_uniform(1, 3)\pi \cdot t/ts\_length + random\_uniform(0, 2\pi)\big)\Big)$
  $data \leftarrow data + harmonic\_amplitude \cdot \sin(2\pi \cdot base\_frequency \cdot n \cdot t + phase)$
end for

4.1.2 Seasonal. For the seasonal component, Fourier series theory [8] demonstrates that any periodic function $f(x)$ can be represented as an infinite sum of sine and cosine functions. Drawing on insights from KAN-AD [39], which shows that most normal time series patterns can be well-approximated using only a small number of sine and cosine terms $S_N(x)$ (proof detail in Section F), we generate the seasonal signal by randomly combining a limited set of such functions.

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} (a_n \cos nx + b_n \sin nx)$$

$$S_N(x) = \frac{a_0}{2} + \sum_{n=1}^{N} (a_n \cos nx + b_n \sin nx), |f(x) - S_N(x)| \leq \sigma$$

Specifically, as shown in Algorithm 1, we first randomly select a period length, distinguishing between large periods and small periods to facilitate the subsequent generation of periodic description information. This distinction is essential because if the period of the time series is less than half of the time series length, it is unlikely that significant periodicity will be observable in the image. In real-world data, it is common for a window not to contain a full period. Next, we determine the fundamental frequency based on the period and randomly select *num_harmonics* frequency components. We then generate time series data corresponding to these frequency components and sum them up. It is important to note that different frequency components have different intensities. Generally, low-frequency components have higher intensities, while high-frequency components (which are likely noise) have lower intensities. Therefore, we use 1/n to control the intensity of high-frequency components and introduce randomness to ensure random perturbations in amplitude. With this approach, we generate a large number of random periodic components.

4.1.3 Noise. During the noise insertion phase, we randomly select high or low noise based on the amplitude of the entire time series.

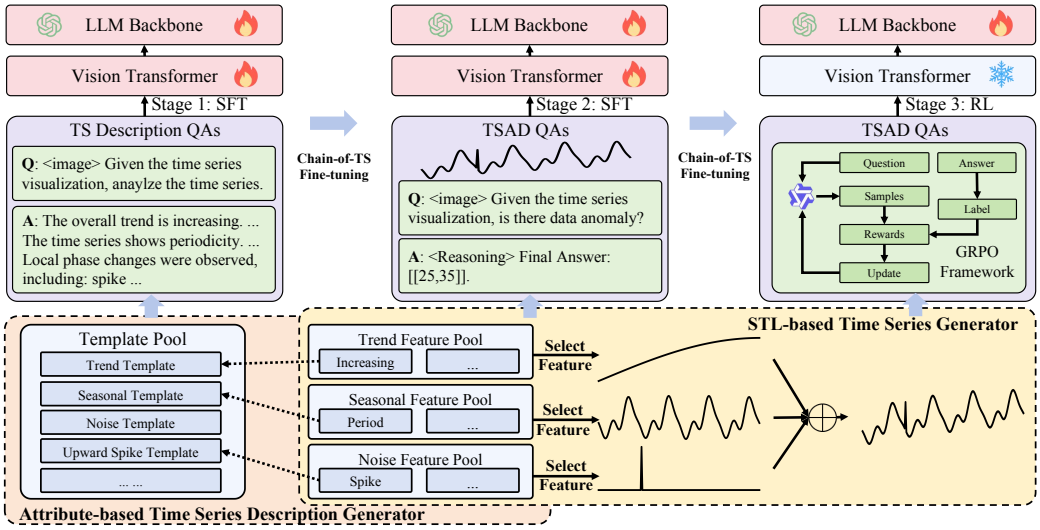


Figure 3: Overview of ViTs.

Table 2: Details of different types of anomalies.

Type	Category or Description
Spike	Upward spike, Downward spike, Continuous upward spike, Continuous downward spike, Upward convex, Downward convex, Rapid rise followed by slow decline, Slow rise followed by rapid decline ...
Level	Sudden increase, Sudden decrease, Increase after downward spike, Decrease after upward spike, Increase after upward spike ...
Trend	Random Generate Multiple Trend
Frequency	Modify Low Frequency Components

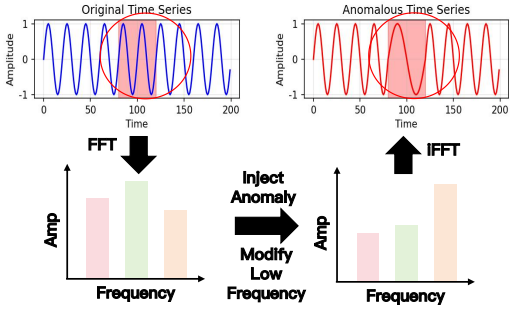


Figure 4: Illustration of frequency anomalies.

More importantly, we insert anomalies into the time series. To ensure a sufficient diversity of anomaly types, we analyzed a large amount of real data and summarized the four types of anomalies in Section 2.2. Each type of anomaly includes multiple implementation methods, as shown in Table 2.

Spike is the most common type of anomaly in real-world time series data, but spike patterns vary widely. Therefore, we categorize them into multiple forms such as upward spike. Level shift is another common anomaly in real-world systems and frequently occurs during system changes. Besides the common sudden increases and decreases, a level shift might be preceded by a spike, resulting in types like decrease after an upward spike. For trend anomalies, we randomly select a segment of the time series and generate a new trend to be inserted as the anomaly. For frequency anomalies, as depicted in Figure 4, we insert anomalies by modify the frequency domain. This ensures that the periodicity of this segment is significantly different from the normal time series around it. It is important to note that when selecting the frequency components to perturb, we randomly choose components with relatively high intensity, as smaller frequency components might be noise.

4.2 Attribute-based Time Series Description Generator

We demonstrate later that the primary reason for the suboptimal performance of current VLMs on TSAD lies in the vision encoder's inability to extract meaningful information relevant to TSAD—specifically, periodic and trend-related information. CLIP [18] has shown that the most effective way to train a vision encoder is through large-scale image–text pair datasets. Therefore, to enhance the vision encoder's ability to capture key features of time series, we propose an attribute-based time series description generator to generate time series-related QA. Specifically, we generate descriptive information based on the attributes selected during the previous STL stage when creating the time series. The descriptions mainly encompass four aspects: (1) The trend of the time series: whether it is increasing, decreasing, etc. (2) The periodicity of the time series: if the period is less than half the time series length, it is considered to have significant periodicity. (3) Local anomalies in the time series: descriptions of various inserted anomalies. We set a description template for each type of anomaly and generate

descriptive information. (4) Noise intensity: some time series have strong noise, while others have weak noise. Examples and details can be found in Section G.2.

4.3 Chain-of-TS Fine-tuning Strategy

As illustrated in Figure 3, Chain-of-TS involves a three-stage fine-tuning process.

4.3.1 Stage 1: Time Series Knowledge Injection with SFT. In Section 3.2, we highlight that a key limitation in TSAD performance lies in the vision encoder's inability to effectively capture detail-rich features relevant to TSAD. Therefore, enhancing the feature extraction capabilities of the vision encoder is crucial for improving TSAD performance. A common approach for fine-tuning vision encoders involves using image-text QA pairs with the CLIP [18]. In our work, we adopt a similar strategy to achieve CLIP-equivalent performance by directly fine-tuning the VLM in an end-to-end manner. During this fine-tuning phase, we open up all parameters, including the vision encoder. The training data used in this stage is QAs generated by the attribute-based time series description generator, which provides rich information on periodicity, trends, and other relevant features. Subsequent experiments validate the significance of this stage, demonstrating that it substantially enhances the upper bound of the VLM's ability to understand time series.

4.3.2 Stage 2: Anomaly Detection Enhancement with SFT. In the first stage, we only taught the VLM how to understand time series images and how to better extract features from them. However, it still does not know how to perform time series tasks such as TSAD. Therefore, in the second stage, we open up all parameters and use TSAD-related QA to teach the TS-VLM how to efficiently and accurately complete TSAD tasks. Since we know the ground truth of the inserted anomalies, it is easy to obtain TSAD QA.

4.3.3 Stage 3: Anomaly Reasoning Refinement with RL. Through the first two stages, we have significantly enhanced the VLM's capabilities for TSAD. However, SFT primarily encourages pattern matching and may lead to partial overfitting, leaving the model's full potential underexplored. Recent research suggests that RL can further unlock model capacity by enabling exploration within the answer space through sampling and iterative reward-based optimization. This approach not only helps uncover deeper capabilities of the model but also tends to offer improved generalization. To this end, we adopt the Group Relative Policy Optimization [9](GPPO, detailed in Section H and Figure 3) to further explore and enhance the TSAD capabilities of the VLM.

$$f1\_reward = \begin{cases} -0.5, real = None\ and\ predict \neq None \\ 0.5, real = None\ and\ predict = None \\ f1(predict,\ real), otherwise \end{cases}$$

$$format\_reward = \begin{cases} 1, & \langle think \rangle \langle /think \rangle\ [[start1, start2]\ ...] \\ 0, & otherwise \end{cases}$$

$$reward = 0.9 \times f1\_reward + 0.1 \times format\_reward$$

We choose to incorporate an F1-based reward for its simplicity and effectiveness, while also including a format reward to guide the model in generating outputs that adhere to the required structure and enhance its reasoning ability. As described in the above formula, for the $f1\_reward$, we consider two aspects. For windows with anomalies, we directly calculate the F1 score as the reward. For windows without anomalies, the F1 score is 0 regardless of the model's response. Since we do not want the model to falsely identify anomalies in normal windows, we give a reward of 0.5 for correct predictions and -0.5 for incorrect predictions. We also demonstrate the effectiveness of this reward model through subsequent experiments. As shown in Figure 5, even a small number of training steps results in significant performance gains in TSAD, demonstrating the effectiveness of the final stage.

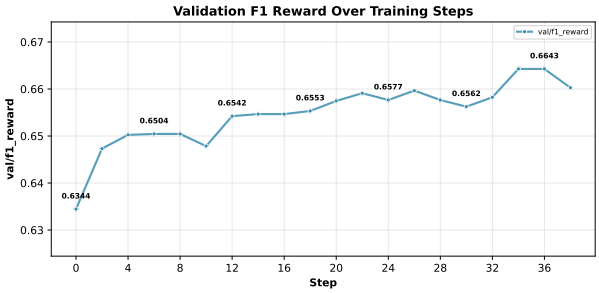


Figure 5: Reward during RL stage.

4.4 Fixed Length Training and Adaptive Length Inference

Traditional methods for constructing time series large models generally use time series of random lengths for training, allowing the model to handle as much diversity as possible. However, the downside of this approach is that the training data size becomes enormous. Unfortunately, even with a massive amount of data, it is impossible to cover all lengths of time series, and some scenarios require the analysis of very long time series. Therefore, an effective TS-VLM should be able to perform anomaly detection on time series of any length.

To address this, as illustrated in Figure 6, we adopt a fixed length training and adaptive length inference schema. Specifically, in the training phase, all time series lengths in images are kept the same at 200 (after rescale). During the inference phase, for time series with different characteristics, we dynamically select different lengths and rescale time series images to a length of 200 (e.g., $[0, 1, \ldots, 800] \rightarrow [0, 0.25, \ldots, 200]$). After inference, we rescale them back to their original lengths (e.g., $[0, 0.25, \ldots, 200] \rightarrow [0, 1, \ldots, 800]$). This method not only reduces the amount of training data but also enables the model to adapt to time series of any length and with any characteristics.

5 Evaluation

5.1 Datasets and Baselines

We evaluated the performance of ViTs using both synthetic data and public datasets. The synthetic dataset contains 2k samples, with anomalies randomly inserted into the time series. The public datasets used include the commonly used benchmarks KPI, Yahoo, and WSD. In terms of metrics, we employ commonly used TSAD

Table 3: Overall performance of different models on synthetic dataset. The VLM ViTs used is Qwen2.5-VL-7B. All Qwen series models we employ are instruct models. (* indicates textual time series input. All models of Qwen series are instruct models.)

	Spike			Trend			Level			Frequency			Mixed			Overall		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Qwen2.5-VL-7B*	0.4496	0.0099	0.0194	0.3327	0.0316	0.0577	0.1660	0.0086	0.0164	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1411	0.0117	0.0217
Qwen2.5-VL-7B	0.5631	0.5490	0.5560	0.7019	0.0913	0.1616	0.5733	0.5819	0.5776	0.8993	0.3035	0.4538	0.6910	0.3107	0.4287	0.3813	0.4216	0.4004
Qwen2.5-VL-32B	0.5436	0.6521	0.5929	0.6045	0.2652	0.3686	0.5398	0.6552	0.5919	0.6539	0.4432	0.5283	0.6825	0.2281	0.3419	0.3262	0.5216	0.4014
Qwen2.5-VL-72B	0.6385	0.8296	0.7216	0.6474	0.6431	0.6453	0.5841	0.8556	0.6943	0.7757	0.7513	0.7633	0.7144	0.4885	0.5802	0.4071	0.7671	0.5319
InternVL3-8B	0.5273	0.6147	0.5677	0.6634	0.5098	0.5766	0.4416	0.5713	0.4981	0.7318	0.5192	0.6075	0.8024	0.5109	0.6243	0.3795	0.5661	0.4544
InternVL3-14B	0.6312	0.6813	0.6553	0.6759	0.5650	0.6155	0.5680	0.6342	0.5993	0.7861	0.7081	0.7450	0.8059	0.5109	0.6254	0.3785	0.6406	0.4758
InternVL3-38B	0.5806	0.7289	0.6464	0.6053	0.6012	0.6032	0.4907	0.7799	0.6024	0.7543	0.7847	0.7692	0.7509	0.4666	0.5756	0.3035	0.7078	0.4248
GPT-4o-mini	0.4326	0.4628	0.4472	0.6194	0.4614	0.5289	0.3883	0.4420	0.4134	0.7001	0.5845	0.6371	0.6801	0.4885	0.5686	0.2173	0.4805	0.2993
GPT-4o	0.6938	0.7533	0.7223	0.8064	0.3674	0.5048	0.7024	0.7415	0.7214	0.8836	0.5438	0.6733	0.7634	0.4447	0.5620	0.5710	0.6284	0.5983
ViTs	0.9565	0.8835	**0.9185**	0.9010	0.6949	**0.7846**	0.9687	0.9404	**0.9543**	0.9526	0.8336	**0.8891**	0.9516	0.6668	**0.7842**	0.8791	0.8380	**0.8581**

Table 4: Zero-shot performance of different methods on public datasets. Zero-shot setting is explained in detail in Section E.

Methods	KPI			Yahoo			WSD			Average		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Spot [20]	0.9661	0.2213	0.3603	0.5723	0.3281	0.4174	0.9475	0.3156	0.4728	0.8280	0.2883	0.4168
SubLOF [4]	0.7191	0.7488	0.7024	0.5213	0.8361	0.5678	0.6523	0.8806	0.6993	0.6309	0.8218	0.6565
Anomaly-Transformer [31]	0.6917	0.7135	0.6701	0.5450	0.8022	0.5717	0.7157	0.8329	0.6473	0.6508	0.7828	0.6297
DCDetector [35]	0.5495	0.7114	0.5862	0.3709	0.8219	0.4597	0.3096	0.8349	0.3830	0.4100	0.7894	0.4763
TranAD [23]	0.8667	0.7768	**0.7855**	0.6770	0.8326	0.6698	0.7549	0.8583	0.6988	0.7662	0.8225	0.7180
TimesNet [28]	0.7245	0.7988	0.7183	0.6922	0.8491	0.7101	0.8345	0.9598	**0.8617**	0.7504	0.8692	0.7633
ViTs	0.7656	0.7590	0.7626	0.7444	0.9558	**0.8373**	0.7645	0.9565	0.8497	0.7581	0.8904	**0.8142**

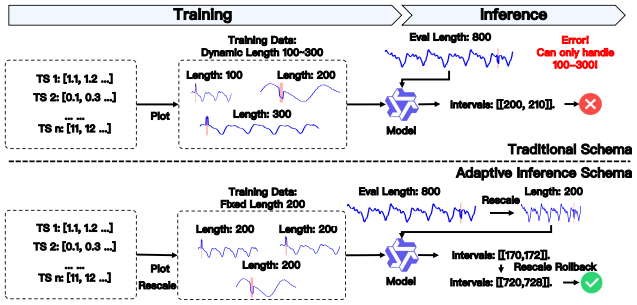


Figure 6: Fixed length training and adaptive length inference schema.

indicators: precision, recall, and point-adjusted F1 score. For baseline comparisons, among open-source models, we select the current SOTA VLMs, including the Qwen2.5-VL [22] series and InternVL [5] series, while for proprietary models, we choose the leading SOTA models GPT-4o-mini and GPT-4o [2]. Additionally, we compare ViTs with SOTA TSAD methods. Detailed experimental settings can be found in Section E.

In real-world evaluations, LLM-based methods often struggle to output anomaly probabilities or scores, while users are more concerned with the confidence level of the anomalies. To address this issue, we propose a sliding window-based anomaly score. Specifically, by using a sliding window, each point in the time series is evaluated multiple times by the LLM. The number of times the LLM identifies a point as an anomaly is used as the anomaly score, which is then incorporated into the final F1 score calculation.

5.2 Overall Performance

The evaluation results on the synthetic dataset are shown in Table 3. It can be observed that for both open-source and proprietary models, the TSAD performance for various types of anomalies improves significantly with an increase in model size. Notably, the capabilities of Qwen2.5-VL-72B are comparable to those of GPT-4o, with superior performance for certain anomalies. However, even the current SOTA models have an F1 score that falls short of 0.6. In contrast, ViTs, with only 7B parameters and trained with 15k data samples, achieves an average F1 score improvement of over 25% and exhibits balanced performance across various types of anomalies.

The evaluation results on public datasets are shown in Table 4. Despite being trained solely on synthetic data, ViTs is able to zero-shot detect anomalies in public datasets, outperforming the current SOTA and achieving an average F1 score of approximately 0.8. This demonstrates the diversity of ViTs' data construction and the effectiveness of its training strategy, while also proving ViTs' excellent generalization capabilities.

5.3 Ablation Study

To demonstrate the effectiveness of Chain-of-TS fine-tuning strategy, we compared different strategies. The results are shown in Table 5. It can be observed that both the initial SFT steps and the third RL step contribute to improvements in the final TSAD performance, with the third stage RL providing nearly a 10% boost. Removing any of the stages significantly degrades performance. Overall, SFT injects new knowledge into the model, while RL helps the model explore and utilize this knowledge for better results. Additionally, we evaluated the impact of freezing LLM parameters during the SFT stage. It can be seen that freezing LLM parameters in

Table 5: Overall performance of different settings on synthetic data. Full results are shown in Table 7.

Setting	Variant	P	R	F1
Naive		0.3813	0.4216	0.4004
#1 SFT-#2 SFT	#1 Frozen LLM	0.9388	0.6280	0.7526
#1 SFT-#2 SFT	#1 Full	0.9483	0.6602	0.7785
#2 SFT		0.9345	0.6400	0.7597
#3 RL		0.5892	0.7101	0.6440
#1 SFT-#2 SFT-#3 RL	#3 Full	0.8791	0.8380	**0.8581**
#1 SFT-#2 SFT-#3 RL	#3 Frozen Vision	0.8829	0.8224	0.8516

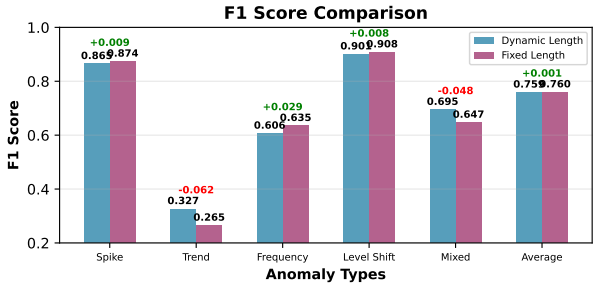


Figure 7: Comparison of fixed length training and dynamic length training.

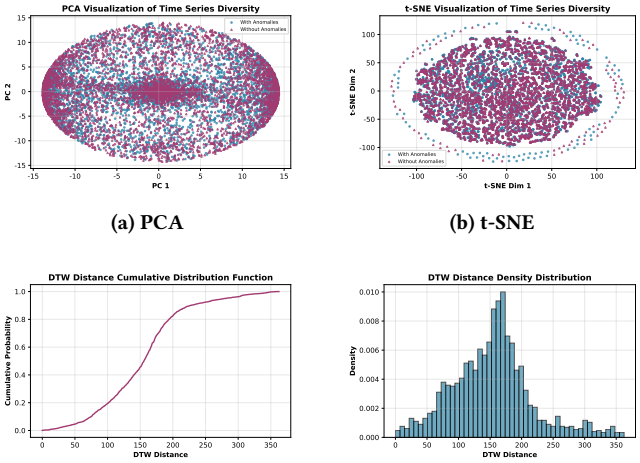
Table 6: Performance on KPI and Yahoo datasets with different window sizes.

Dataset	Window	F1	P	R	VUS_ROC
KPI	100	0.5630	0.4090	0.9018	0.6166
KPI	200	0.7626	0.7656	0.7590	0.6106
Yahoo	50	0.8373	0.7444	0.9558	0.9389
Yahoo	100	0.8324	0.7378	0.9552	0.9304

the first stage negatively affects the results, as it creates a modality gap. Freezing the vision parameters during RL has little impact since the reasoning and exploration capabilities are primarily provided by the LLM backbone. Therefore, in the third stage, we chose to freeze the vision parameters to accelerate training efficiency.

5.4 Fixed Length Training and Adaptive Length Inference

In Section 4.4, we proposed a new fiexed length training and adaptive inference schema with time series image rescaling. To verify the effectiveness of fixed length training, we used the same amount of training data (10k TSAD QA of the same type as in stage 2) and trained models using both fixed length and dynamic length training approaches. The evaluation results on the same synthetic dataset (length = 200) are shown in Figure 7. It can be seen that for most types of anomalies, fixed length training outperforms dynamic length training. This is understandable, as fixed length training allows the VLM to learn more samples of that length, thereby enhancing its TSAD capabilities.



(a) PCA (b) t-SNE

(c) CDF (d) Distribution

Figure 8: Visualization of data diversity.

To verify the effectiveness of adaptive length inference enabled by time series image rescaling, we evaluated models trained with a fixed length on several public datasets. The corresponding results are presented in Table 6. As shown, the optimal sequence length varies across different datasets. Therefore, with adaptive length inference, models can flexibly select suitable lengths for evaluation in different scenarios without requiring retraining.

5.5 Data Diversity

In the field of LLMs, training data serves as the cornerstone, with its richness and diversity determining the types of tasks a model can perform and the accuracy with which it can complete them. The same holds true for TSAD; only if a model has encountered sufficiently rich time series data can it effectively handle new and different time series. Although we have previously demonstrated that data synthesized through STL, where the seasonal component can approximate any continuous function with bounded error, further evidence of the diversity of synthetic data is provided here.

For this purpose, we perform t-SNE and PCA processing on 10k samples and visualize the results. As shown in Figure 8, normal and anomalous time series are evenly distributed, forming a circular outline. This is expected because all time series were standardized using z-score normalization, ensuring they are bounded.

Additionally, we randomly sampled 2000 pairs and calculated their DTW distances, plotted as a CDF and distribution graph. The results indicate that most time series are relatively distant from each other, further proving the diversity of the time series.

5.6 Reward Model

RL plays an important role as a key part of the Chain-of-TS strategy. However, a critical element in RL is the design of the reward model. DeepSeek R1 [9] has demonstrated that a simple rule-based reward model can achieve relatively good results. Therefore, we designed a reward model based on the F1 score. However, we did not solely use

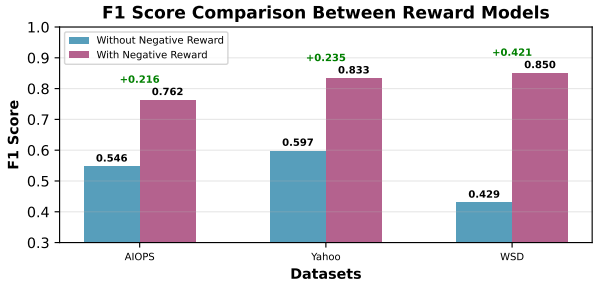


Figure 9: Comparison of different reward model.

the F1 score; we also designed a negative reward for segments without anomalies. This is because, without restrictions, any prediction result for an anomaly-free window would yield an F1 score of 0. This would cause the model to bias towards sampling segments that satisfy the anomaly window's F1 value, leading to a high likelihood of predicting anomalies in windows without anomalies.

Here, we compared the impact of adding a negative reward on the results, as shown in Figure 9. It can be seen that not adding a negative reward increases the number of false positives, thereby significantly reducing precision and the F1 score.

6 Conclusion

This work presents **ViTs**, a novel VLM that significantly advances the SOTA in TSAD. ViTs introduces an STL-based periodic data generation method and the Chain-of-TS fine-tuning strategy. Additionally, we propose a new fixed length training and adaptive length inference schema. Extensive experiments across multiple datasets demonstrate that ViTs outperforms SOTA by substantial margins.

References

[1] Wsd dataset. Available: https://github.com/anotransfer/AnoTransfer-data/.

[2] Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).

[3] Alnegheimish, S., Nguyen, L., Berti-Equille, L., and Veeramachaneni, K. Large language models can be zero-shot anomaly detectors for time series? *arXiv preprint arXiv:2405.14755* (2024).

[4] Breunig, M. M., Kriegel, H.-P., Ng, R. T., and Sander, J. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data* (2000), pp. 93–104.

[5] Chen, Z., Wu, J., Wang, W., Su, W., Chen, G., Xing, S., Zhong, M., Zhang, Q., Zhu, X., Lu, L., et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2024), pp. 24185–24198.

[6] Chikkerur, S., Cartwright, A. N., and Govindaraju, V. Fingerprint enhancement using stft analysis. *Pattern recognition 40*, 1 (2007), 198–211.

[7] Cleveland, R. B. Stl : A seasonal-trend decomposition procedure based on loess.

[8] Fourier, J. B. J. *Théorie analytique de la chaleur*, vol. 1. Gauthier-Villars, 1888.

[9] Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948* (2025).

[10] Jian, C., Yang, K., and Jiao, Y. Tri-level navigator: Llm-empowered tri-level learning for time series ood generalization. *Advances in Neural Information Processing Systems 37* (2024), 110613–110642.

[11] Jin, M., Wang, S., Ma, L., Chu, Z., Zhang, J. Y., Shi, X., Chen, P.-Y., Liang, Y., Li, Y.-F., Pan, S., et al. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728* (2023).

[12] Li, Z., Zhao, N., Zhang, S., Sun, Y., Chen, P., Wen, X., Ma, M., and Pei, D. Constructing large-scale real-world benchmark datasets for aiops. *arXiv preprint arXiv:2208.03938* (2022).

[13] Liu, C., He, S., Zhou, Q., Li, S., and Meng, W. Large language model guided knowledge distillation for time series anomaly detection. *arXiv preprint arXiv:2401.15123* (2024).

[14] Liu, C., Xu, Q., Miao, H., Yang, S., Zhang, L., Long, C., Li, Z., and Zhao, R. Timecma: Towards llm-empowered multivariate time series forecasting via cross-modality alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2025), vol. 39, pp. 18780–18788.

[15] Liu, J., Zhang, C., Qian, J., Ma, M., Qin, S., Bansal, C., Lin, Q., Rajmohan, S., and Zhang, D. Large language models can deliver accurate and interpretable time series anomaly detection. *arXiv preprint arXiv:2405.15370* (2024).

[16] Liu, Y., Qin, G., Huang, X., Wang, J., and Long, M. Autotimes: Autoregressive time series forecasters via large language models. *Advances in Neural Information Processing Systems 37* (2024), 122154–122184.

[17] Pei, C., Wang, Z., Liu, F., Li, Z., Liu, Y., He, X., Kang, R., Zhang, T., Chen, J., Li, J., et al. Flow-of-action: Sop enhanced llm-based multi-agent system for root cause analysis. In *Companion Proceedings of the ACM on Web Conference 2025* (2025), pp. 422–431.

[18] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning* (2021), PmLR, pp. 8748–8763.

[19] Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K., and Yao, S. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems 36* (2023), 8634–8652.

[20] Siffer, A., Fouque, P.-A., Termier, A., and Largouet, C. Anomaly detection in streams with extreme value theory. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2017), pp. 1067–1075.

[21] Tan, M., Merrill, M., Gupta, V., Althoff, T., and Hartvigsen, T. Are language models actually useful for time series forecasting? *Advances in Neural Information Processing Systems 37* (2024), 60162–60191.

[22] Team, Q. Qwen2.5-vl, January 2025.

[23] Tuli, S., Casale, G., and Jennings, N. R. Tranad: Deep transformer networks for anomaly detection in multivariate time series data. *arXiv preprint arXiv:2201.07284* (2022).

[24] Wang, C., Qi, Q., Wang, J., Sun, H., Zhuang, Z., Wu, J., Zhang, L., and Liao, J. Chattime: A unified multimodal time series foundation model bridging numerical and textual data. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2025), vol. 39, pp. 12694–12702.

[25] Wang, X., Feng, M., Qiu, J., Gu, J., and Zhao, J. From news to forecast: Integrating event analysis in llm-based time series forecasting with reflection. *Advances in Neural Information Processing Systems 37* (2024), 58118–58153.

[26] Wang, Z., Pei, C., Ma, M., Wang, X., Li, Z., Pei, D., Rajmohan, S., Zhang, D., Lin, Q., Zhang, H., et al. Revisiting vae for unsupervised time series anomaly detection: A frequency perspective. In *Proceedings of the ACM Web Conference 2024* (2024), pp. 3096–3105.

[27] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems 35* (2022), 24824–24837.

[28] Wu, H., Hu, T., Liu, Y., Zhou, H., Wang, J., and Long, M. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186* (2022).

[29] Xie, Z., Li, Z., He, X., Xu, L., Wen, X., Zhang, T., Chen, J., Shi, R., and Pei, D. Chatts: Aligning time series with llms via synthetic data for enhanced understanding and reasoning. *arXiv preprint arXiv:2412.03104* (2024).

[30] Xu, H., Chen, W., Zhao, N., Li, Z., Bu, J., Li, Z., Liu, Y., Zhao, Y., Pei, D., Feng, Y., et al. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 world wide web conference* (2018), pp. 187–196.

[31] Xu, J., Wu, H., Wang, J., and Long, M. Anomaly transformer: Time series anomaly detection with association discrepancy. *arXiv preprint arXiv:2110.02642* (2021).

[32] Xu, X., Wang, H., Liang, Y., Yu, P. S., Zhao, Y., and Shu, K. Can multimodal llms perform time series anomaly detection? *arXiv preprint arXiv:2502.17812* (2025).

[33] Xu, Z., Zeng, A., and Xu, Q. Fits: Modeling time series with $10k$ parameters. *arXiv preprint arXiv:2307.03756* (2023).

[34] Yahoo. Yahoo dataset, 2015.

[35] Yang, Y., Zhang, C., Zhou, T., Wen, Q., and Sun, L. Dcdetector: Dual attention contrastive representation learning for time series anomaly detection. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (2023), pp. 3033–3045.

[36] Yu, X., Chen, Z., Ling, Y., Dong, S., Liu, Z., and Lu, Y. Temporal data meets llm–explainable financial time series forecasting. *arXiv preprint arXiv:2306.11025* (2023).

[37] Zhang, D. Wavelet transform. In *Fundamentals of image data mining: Analysis, Features, Classification and Retrieval*. Springer, 2019, pp. 35–44.

[38] Zhang, X., Zhao, Z., Tsiligkaridis, T., and Zitnik, M. Self-supervised contrastive pre-training for time series via time-frequency consistency. *Advances in*

neural information processing systems 35 (2022), 3988–4003.
[39] Zhou, Q., Pei, C., Sun, F., HanJing, Gao, Z., Zhang, H., Xie, G., Pei, D., and li, J. KAN-AD: Time series anomaly detection with kolmogorov–arnold networks. In *Forty-second International Conference on Machine Learning* (2025).
[40] Zhou, Z., and Yu, R. Can llms understand time series anomalies? *arXiv preprint arXiv:2410.05440* (2024).

A Related Work

LLMs for Time Series LLMs have gained significant traction in time series analysis due to their exceptional textual reasoning capabilities, with applications spanning three main directions: (1) Time Series Forecasting: Some methods [36] leverage LLMs' semantic knowledge through prompt engineering techniques (e.g., few-shot learning), while approaches like Time-LLM [11] and ChatTime [24] enhance forecasting performance by fine-tuning the LLM backbone to better utilize its parametric knowledge [14]; (2) Time Series Anomaly Detection: Methods such as LLMAD [15], AnomalyLLM [13], and SigLLM [3] primarily employ prompt engineering to describe anomaly patterns, though notably no current approach has investigated dedicated TSAD-oriented fine-tuning of LLMs; (3) Time Series Reasoning: Methods like ChatTS [29] demonstrate improved performance on time series classification and related tasks by incorporating specialized time series encoders and large-scale temporal data training, thereby extending LLMs' capabilities beyond their original text-based design.

VLMs for Time Series Recent research has witnessed a growing trend of employing VLMs for time series analysis. Notable works include AnoLLM [40], which systematically investigates how prompt engineering strategies can optimize TSAD performance through extensive experimentation, and MLLM [32], which comprehensively evaluates the current capabilities of multimodal LLMs in TSAD tasks. Compared to conventional approaches, VLMs demonstrate unique advantages for TSAD applications - their multimodal reasoning paradigm better aligns with human cognitive processes, offering superior interpretability and greater potential for future development in this domain.

B Anomaly Types

(1) **Spike Anomaly:** The most common type of anomaly, characterized by a sudden sharp increase or decrease in the time series value, followed by a return to the normal range after a short duration. (2) **Trend Anomaly:** Indicates a change in the overall trend of the time series, which may manifest as an alteration in the rate of increase/decrease or a reversal in trend direction (e.g., shifting from an upward to a downward trend). (3) **Level Anomaly:** Refers to an abrupt shift in the baseline level of the time series, after which the values fluctuate around this new baseline. (4) **Frequency Anomaly:** Represents a local disruption in the periodic pattern of the time series, typically involving significant changes in either the frequency or amplitude of fluctuations within a specific time window.

C Visualization of Different Plot Types

Shown in Figure 10.

D Output of Different Image Types

LINE: *Key observations from the chart: 1. The series starts at a low value around 0.00 and gradually increases. 2. There are fluctuations in the series, indicating variability or noise in the data. 3. A significant peak occurs around the 125th index, reaching a value close to 1.00. 4. After the peak, the series fluctuates but generally decreases and stabilizes around a value of 0.50.*
LINE_STFT: *The image consists of two plots. The top plot is labeled "Time Series (Index-based)" and shows a time series of data indexed from 0 to 160. The y-axis represents the "Value," which ranges from 0 to 1. The data appears to be a noisy signal with some peaks and troughs, particularly noticeable around index 125 where there is a sharp peak. The bottom plot is labeled "STFT Magnitude Spectrum."*

E Experiment Setting

The following briefly introduces some experimental settings. For more details, please refer to the code.

E.1 Zero-shot Setting

The zero-shot setting refers to the scenario where the test and training data are different. Therefore, when evaluating on public datasets, ViTs is trained using synthetic data, without using any real-world datasets. For other SOTA methods, training is conducted using datasets other than the evaluation datasets. For example, when evaluating the KPI dataset, training is done using the Yahoo, WSD, NAB, UCR, and TODS datasets. This allows for a relatively fair comparison between ViTs and other methods.

E.2 Training Data

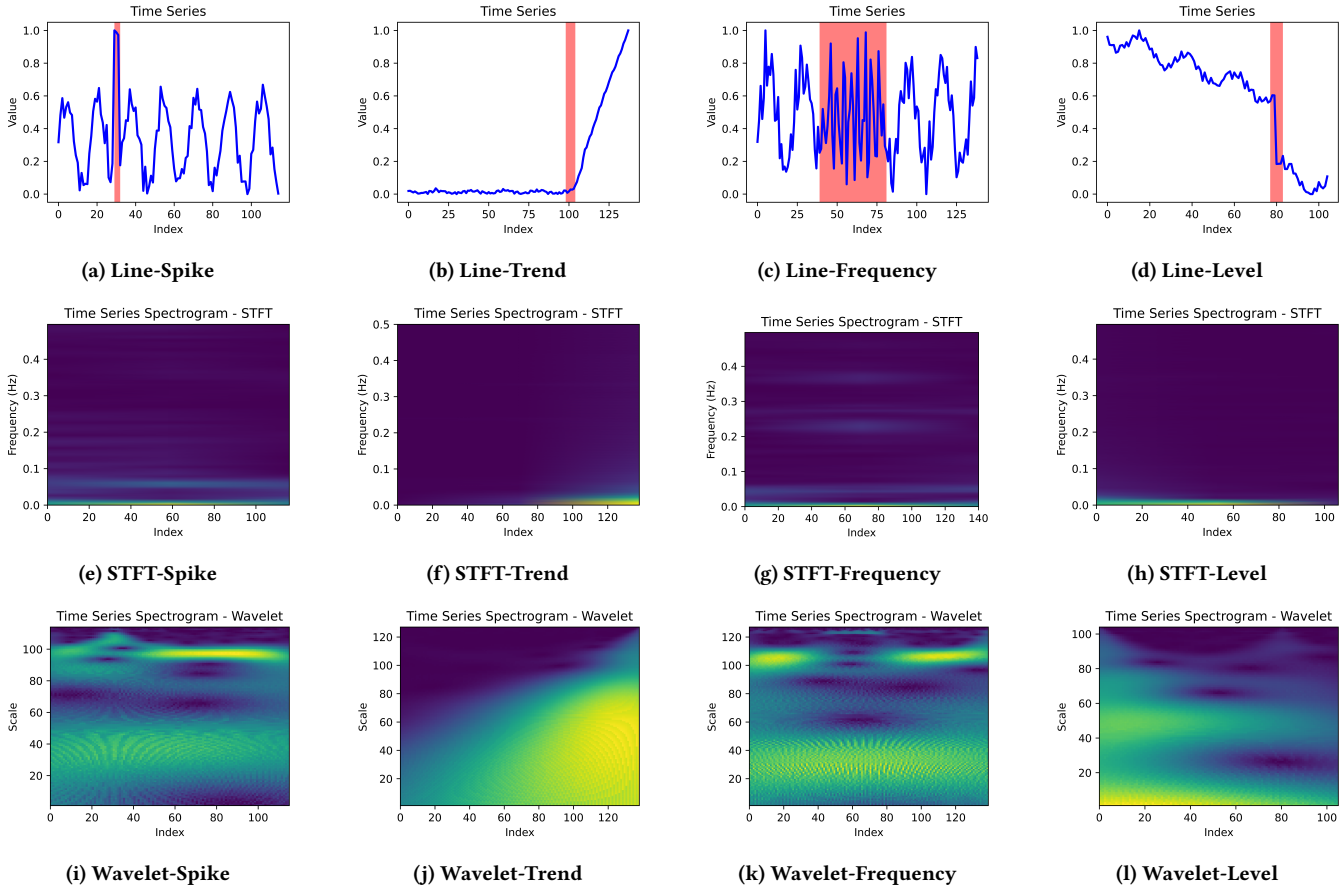
The training data consists of a total of 15k QA pairs. The first stage includes 5k time series description QA pairs, while the second and third stages each include 10k TSAD QA pairs. The time series length in the QA pairs is consistently 200, generated using the STL-based method proposed in the paper. Four types of anomalies are randomly inserted. It is important to note that frequency and trend anomalies are only inserted into time series with shorter periods because only short periods with sufficient periodic instances can effectively display the anomalies.

E.3 Testing Data

The testing data includes two parts. The synthetic test data, generated using the method described in the paper, consists of 2k TSAD QA pairs. For public datasets, to ensure testing efficiency, we select the last 50% of each time series from the Yahoo dataset and the last 20% from the KPI and WSD datasets as the test sets. A sliding window approach is then employed to generate different detection windows. To optimize testing speed, we set the window size to 200, step size to 200, and resize factor to 1 for the KPI and WSD datasets. For the Yahoo dataset, due to the shorter duration represented by each data point (one hour), we set the window size to 100, step size to 100, and resize factor to 0.5. The step size is set to 100 because our method is accurate enough to require only one test per point.

E.4 Pubulic Dataset Details

Yahoo [34] Yahoo is an open data set for anomaly detection released by Yahoo lab. The dataset consists of real and synthetic time series with tagged anomaly points. The dataset tests the detection



(a) Line-Spike (b) Line-Trend (c) Line-Frequency (d) Line-Level

(e) STFT-Spike (f) STFT-Trend (g) STFT-Frequency (h) STFT-Level

(i) Wavelet-Spike (j) Wavelet-Trend (k) Wavelet-Frequency (l) Wavelet-Level

Figure 10: LINE, STFT and Wavelet visualizations of four types of anomalies.

accuracy of various anomaly-types including outliers and change-points. The synthetic dataset consists of time series with varying trend, noise and seasonality. The real dataset consists of time series representing the metrics of various Yahoo services.

KPI [12] KPI is collected from five large Internet companies (Sougo, eBay, Baidu, Tencent, and Ali). Specifically, this dataset includes 29 Curves in total. For each curve, experienced engineers in these companies manually labeled anomalies carefully. Most KPI curves have an interval of 1 minute between two adjacent data points, while some of them have an interval of 5 minutes.

WSD [1] Web service dataset (WSD) contains real-world KPIs collected from three top-tier Internet companies, Baidu, Sogou, and eBay, providing large-scale Web services. Experienced operation engineers have carefully labeled all KPIs in WSD.

E.5 Training Hyperparameters

For the first two SFT training stages, we use the LLaMA-Factory framework for full-parameter training, with one epoch for each stage and a learning rate of 1.0e-5. During the RL stage, we use the EasyR1 training framework. Due to slower convergence, we train for two epochs with a learning rate of 1.0e-6.

F Proof

We prove that if f is a 2π–periodic real function of bounded variation on $[0, 2\pi]$, then its Fourier partial sums

$$S_N(x) \;=\; \frac{a_0}{2} + \sum_{n=1}^{N} \big(a_n \cos nx + b_n \sin nx\big)$$

satisfy the uniform error estimate

$$\big|f(x) - S_N(x)\big| \;\le\; \frac{V_0^{2\pi}(f)}{\pi\, N},$$

where $V_0^{2\pi}(f)$ is the total variation of f on one period.

Let f be a real-valued, 2π–periodic function of bounded variation on $[0, 2\pi]$, with total variation

$$V_0^{2\pi}(f) := \sup_{0=x_0<x_1<\cdots<x_m=2\pi} \sum_{j=1}^{m} \big|f(x_j) - f(x_{j-1})\big|.$$

Its Fourier coefficients are

$$a_n = \frac{1}{\pi} \int_0^{2\pi} f(t) \cos(nt)\, dt,$$

$$b_n = \frac{1}{\pi} \int_0^{2\pi} f(t) \sin(nt)\, dt, \quad n = 0, 1, 2, \ldots.$$

Table 7: Performance of different settings on synthetic data.

Setting		Spike			Trend			Level Shift			Frequency			Mixed			Overall		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Naive		0.5631	0.5490	0.5560	0.7019	0.0913	0.1616	0.5733	0.5819	0.5776	0.8993	0.3035	0.4538	0.6910	0.3107	0.4287	0.3813	0.4216	0.4004
#1 SFT-#2 SFT	#1 Frozen LLM	0.9395	0.8222	0.8769	0.8420	0.0630	0.1172	0.9523	0.8714	0.9100	0.9468	0.4909	0.6465	0.9225	0.5405	0.6816	0.9388	0.6280	0.7526
#1 SFT-#2 SFT	#1 Full	0.9522	0.8104	0.8756	0.9019	0.2247	0.3597	0.9377	0.8574	0.8958	0.9767	0.5454	0.6999	0.9115	0.5804	0.7092	0.9483	0.6602	0.7785
#2 SFT		0.9347	0.8203	0.8737	0.9365	0.1543	0.2649	0.9459	0.8732	0.9081	0.9328	0.4813	0.6350	0.9038	0.5038	0.6470	0.9345	0.6400	0.7597
#3 RL		0.7850	0.7330	0.7582	0.9187	0.6646	0.7713	0.6922	0.6481	0.6695	0.9184	0.8154	0.8638	0.9025	0.5424	0.6776	0.5892	0.7101	0.6440
#1 SFT-#2 SFT-#3 RL	#3 Full	0.9565	0.8835	0.9185	0.9009	0.6949	0.7846	0.9687	0.9404	0.9543	0.9526	0.8336	0.8891	0.9516	0.6668	0.7842	0.8791	0.8380	0.8581
#1 SFT-#2 SFT-#3 RL	#3 Frozen Vision	0.9591	0.8895	0.9230	0.8822	0.6302	0.7352	0.9681	0.9399	0.9538	0.9608	0.8105	0.8793	0.9552	0.6176	0.7502	0.8829	0.8224	0.8516

Integrating by parts for $n \geq 1$ gives

$$a_n = \frac{1}{\pi}\int_0^{2\pi} f(t)\cos(nt)\,dt = \frac{f(t)\sin(nt)}{\pi n}\Big|_0^{2\pi} - \frac{1}{\pi n}\int_0^{2\pi} f'(t)\sin(nt)\,dt.$$

By periodicity $f(0) = f(2\pi)$, so the boundary term vanishes. Interpreting $f'(t)\,dt$ as the signed measure $df(t)$ and taking absolute values yields

$$|a_n| \leq \frac{1}{\pi n}\int_0^{2\pi} \big|df(t)\big| = \frac{V_0^{2\pi}(f)}{\pi\, n},$$

and similarly $|b_n| \leq V_0^{2\pi}(f)/(\pi n)$.

Define the remainder after N terms by

$$R_N(x) = f(x) - S_N(x) = \sum_{|k|>N} c_k\, e^{ikx},$$

where c_k are the complex Fourier coefficients. Then

$$\begin{aligned}\big|R_N(x)\big| &\leq \sum_{|k|>N} \big|c_k\big| \\ &= \sum_{n=N+1}^{\infty} \big(|c_n| + |c_{-n}|\big) \\ &\leq \sum_{n=N+1}^{\infty} \frac{|a_n| + |b_n|}{1} \\ &\leq \sum_{n=N+1}^{\infty} \frac{2\,V_0^{2\pi}(f)}{\pi\, n} \\ &= \frac{2\,V_0^{2\pi}(f)}{\pi} \sum_{n=N+1}^{\infty} \frac{1}{n}.\end{aligned}$$

Using the standard tail estimate $\sum_{n=N+1}^{\infty} 1/n \leq 1/N$, we conclude

$$\big|f(x) - S_N(x)\big| \leq \frac{V_0^{2\pi}(f)}{\pi\, N}.$$

Thus the N-term Fourier partial sum $S_N(x)$ approximates $f(x)$ uniformly with error $O(1/N)$, as claimed.

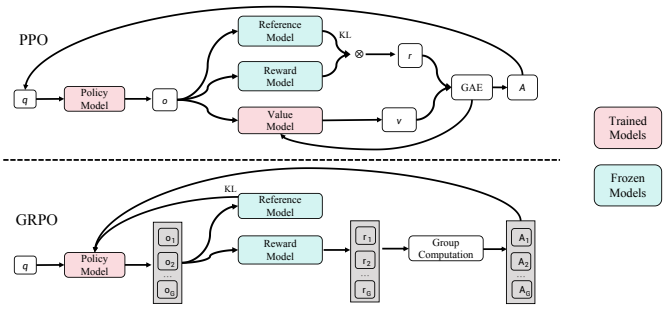


Figure 11: Camprision of GRPO and PPO.

G QA Examples

G.1 Time Series Description QA

Q: *<image>Given the time series visualization, analyze the time series.*
A: *The time series shows periodicity: The amplitude of the periodic fluctuation between point 0 and point 200 is 77.3, the period of the fluctuation is 96. From the perspective of the slope, the overall trend is decreasing. The overall noise standard deviation is around 0.02, very small compared the overall change of the curve. The curve is overall smooth with almost no noise.*

G.2 TSAD QA

Q: *<image>Given the time series visualization, is there any anomaly in the time series? Output the anomalous intervals.*
Output Foarmat: boxed[[start1, end1], [start2, end2], ...]
A: *Final Answer: boxed[]*

H GRPO

GRPO is currently a relatively effective method for RL training. Unlike the traditional Proximal Policy Optimization (PPO), which requires the training of a value function to estimate the advantage function, the core idea of GRPO is to estimate the baseline using intra-group relative rewards. This approach eliminates the need for an additional value function model, or critic model. Specifically, as illustrated in Figure 11, GRPO performs multiple samplings for the same input, generating multiple samples, which are then evaluated using a pre-defined reward model to compute the reward for each sample. Typically, this reward model employs rule-based computation methods, as they are relatively simple and effective. Once all the rewards within a group are obtained, any sample with a reward greater than the group's average reward is considered an advantageous sample. Ultimately, GRPO iteratively optimizes the policy to achieve superior performance.

I Full Results

Shown in Table 7.