Multi-Scale Transformers with Contrastive Learning for UAV Anomaly Detection

Gang Hu, Zhongliang Zhou, Zhengxin Li, Zheng Dong, Jiayong Fang, Yu Zhao, *Graduate student member, IEEE*, and Chuhan Zhou

***Abstract*—With the widespread application of unmanned aerial vehicles (UAVs) in various fields, anomaly detection in flight data has become increasingly important. However, temporal variations and temporal correlations in flight data challenging to achieve accurate anomaly detection. To address these issues, the multi-scale Transformers with contrastive learning for UAV anomaly detection (MTCL-UAV) is proposed to achieve adaptive multi-scale modeling. The model is based on the mixture-of-experts (MoE) architecture, and each MoE block includes a router, expert networks (ENs) and an aggregator. The router performs a temporal decomposition to select optimal scales for each sample, and the aggregator fuses the outputs of the ENs into an integrated representation. To learn the temporal correlations of flight data and improve the model's representation, a dual attention mechanism enhanced by contrastive learning (CL-DAM) is introduced, which captures not only intra- and inter-patch correlation relationships but also neighborhood relationships among patches. Experiments on a real flight dataset demonstrate that MTCL-UAV not only achieves superior performance compared to other methods, but also exhibits strong robustness. The code is available at https://github.com/SteelHu/MTCL-UAV**

***Index Terms*—Unmanned aerial vehicles (UAVs), anomaly detection, flight data, multi-scale modeling, contrastive learning.**

I. Introduction

With advancements of sensor and communication technologies, unmanned aerial vehicles (UAVs) are widely used in communications [1], emergency response [2], [3], agriculture [4], and environmental monitoring [5], [6]. As the number of UAVs increases, their safety and reliability requirements also increase rapidly. Compared to manned aircraft, the UAV faces more severe safety and reliability challenges, which also leads to its higher accident rate [7], [8]. During the flight, the flight data records various UAV parameters, such as altitude, speed, angular velocity, etc. These parameters can effectively reflect the flight state of the UAV. By analyzing these flight data, the abnormal state of the UAV can be found in time, and timely measures can be taken to reduce the occurrence of accidents [9]. Therefore, it is necessary to develop anomaly detection methods for UAV flight data. Anomaly detection refers to identifying data that significantly deviate from the normal distribution [10], such as sensor failure, abnormal manipulation behavior, and control system failure. The anomaly detection methods for UAVs are classified into knowledge-based, model-based, and data-driven approaches [8]. As UAV systems become increasingly complex and the volume of generated data increases, data-driven approaches have gained prominence in flight data analysis because they rely minimally on prior domain knowledge, allowing flexible pattern learning directly from historical data. There have been many research results on data-driven methods. Most of the early approaches focused on machine learning algorithms such as k-nearest neighbors (KNNs) [11], [12], kernel principal component analysis (KPCA) [13]–[15], recursive leat squares (RLS) [9], isolation forest (IForest) [16] and support vector machine (SVM) [17]. However, due to the complex meteorological conditions and diverse mission requirements in UAV operational environments, these methods may show limited flexibility and adaptability in processing complex flight data.

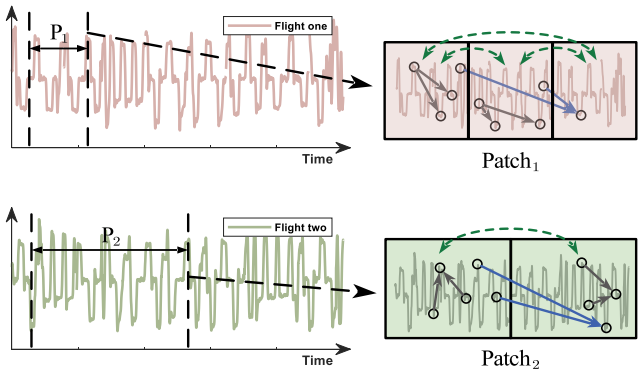

Fig. 1. **Left**: The variation curves of UAV's flight speed. Pink and green represent two samples, respectively. **Right**: Flight data are divided into patches of varying sizes as temporal resolution. Grey and blue arrows represent local and global correlations. Green arrows represent the neighborhood relationships among patches.

Existing research indicates that improving the performance of model's anomaly detection relies on achieving more effective feature extraction from flight data [18]. In recent years, deep learning methods have attracted much attention because of their powerful feature extraction capabilities. For example, convolutional neural network (CNN) [19], [20], long short-term memory networks (LSTM) [20]–[23] and Transformer [20], [24] have all been used for UAV anomaly detection with flight data. However, these methods mainly

Corresponding author: Zhengxin Li (zhengxinli@nwpu.edu.cn)

Gang Hu, Zhongliang Zhou, Zhengxin Li, Jiayong Fang, and Yu Zhao are with the College of Equipment Management and Unmanned Aerial Vehicle Engineering, Air Force Engineering University, Xi'an, 710043, China (E-mail: gangh24@163.com (Gang Hu)).

Chuhan Zhou is with the College of Air Traffic Control and Navigation, Air Force Engineering University, Xi'an, 710043, China(E-mail: chuhan_zhou_edu@163.com).

Zheng Dong is with the Beijing Bytedance Technology Co Ltd, Beijing, 200000, China (E-mail: zhdong0@outlook.com).

focus on single-scale feature extraction of flight data and lack multi-scale feature extraction, which has not been effectively explored so far. Affected by the environment and task, UAV flight data have significant characteristics of temporal variation. In the left part of Fig. 1, two flight speed samples are selected from a real UAV flight dataset [25], and in the right part, the samples are divided at different scales, which show different feature patterns. Furthermore, different types of anomalies in UAV flight data also have various temporal scale variations. For example, when actuator failures occur in the UAVs, the elevator failure and the aileron failure present different characteristics of temporal scale variations in the flight data. Therefore, multi-scale modeling is required to extract features at different scales from UAV flight data. Nevertheless, as the optimal scales of each sample are different, it is not appropriate to use fixed multi-scale feature extraction for all samples. This not only significantly increases the computational cost, but may also bring performance degradation. Chen et al. [26] introduced the multi-scale Transformers with adaptive pathways model for time series forecasting (Pathformer), which achieve adaptive multi-scale modeling while keeping computational costs relatively low. It uses a dual attention mechanism to extract intra- and inter-patch information but ignores the neighborhood relationships among patches. As shown in the right part of Fig. 1, the temporal correlations include not only the intra- and inter-patch correlations but also the neighborhood relationships among the patches. All of these temporal correlations are important for accurate anomaly detection [27], [28].

In summary, effective multi-scale modeling of UAV flight data for accurate anomaly detection requires addressing two key challenges: temporal scale variations and temporal correlations. To address these issues, the multi-scale Transformers with contrastive learning model for UAV anomaly detection (**MTCL-UAV**) is proposed. The model is built based on the sparse mixture-of-experts (MoE) architecture [29]–[31], and each sparse MoE block contains three components: (1) **Router**, (2) **Expert Networks** (ENs), and (3) **Aggregator**. The Router adaptively selects the key scales for each input sample and routes them into the corresponding ENs. Each EN corresponds to a specific scale and learns the temporal correlation at this scale. To fully identify temporal correlations, a dual attention mechanism enhanced by contrastive learning (**CL-DAM**) is proposed, which captures not only intra- and inter-patch correlation relationships but also neighborhood relationships between patches. Finally, the Aggregator fuses the outputs of the various ENs into an integrated representation. Our contributions are given below.

1) An unsupervised anomaly detection framework, called multi-scale Transformers with contrastive learning for UAV Flight Data, is introduced to perform adaptive multi-scale modeling according to temporal scale variations in flight data. It can extract and fuse the features of multiple time scales of flight data to better reconstruct the input data.
2) A dual attention mechanism enhanced by contrastive learning is proposed to extract the temporal correlations from flight data in each expert network. CL-DAM not only captures intra- and inter-patch correlation relationships, but also enhances the model's representation of similarity relationships among patches through contrastive learning, thus further improving the model's feature extraction capability.
3) The experimental results demonstrate that the proposed model not only outperforms existing methods but also exhibits strong robustness across varying noise levels. Moreover, visualization further validates the necessity of employing multi-scale modeling in UAV flight datasets and the effectiveness of the proposed model.

The remainder of this paper is organized as follows. Section II reviews related work on UAV anomaly detection and multi-scale modeling. Section III introduces the MTCL-UAV anomaly detection framework, detailing data preprocessing, the MTCL model, and the detection strategy. Section IV outlines the experiments and presents the results. Finally, Section VI summarizes the study and suggests directions for future research.

II. Related Work

This section briefly reviews UAV anomaly detection methods and the application of multi-scale modeling techniques in time series analysis.

A. UAV Anomaly Detection

UAV anomaly detection methods are classified into knowledge-based, model-based, and data-driven approaches [8]. Early techniques primarily utilized statistical analysis to identify abnormal data, but they struggled with complex temporal dependencies and high dimensionality [8]. In contrast, data-driven approaches include machine learning and deep learning methods [32], which extract patterns directly from historical flight data. Machine learning methods are valued for their simplicity and interpretability, but often require complex feature engineering [33]–[35].

Recently, deep learning methods have gained prominence due to their powerful feature extraction and modeling capabilities. Wang et al. [21] proposed an LSTM-based approach to detect abnormalities in UAV systems. Dhakal et al. [22] trained an autoencoder (AE) on normal data to identify anomalies based on reconstruction errors. Bae et al. [23] developed a distributed LSTM-AE model that utilized state, position, and velocity data, achieving faster detection with lower computational costs. He et al. [36] proposed a conditional generative adversarial network-based collaborative intrusion detection (CGAN) with distributed federated learning powered by blockchain to address challenges related to small sample sizes and imbalanced data. Yang et al. [19] proposed a model based on timestamp slicing and multi-split convolutional neural network (TS-MSCNN), which extracts and fuses UAV temporal domain information to learn key features from flight data. Jiang et al. [20] proposed a robust spatial-temporal autoencoder (RSTAE) model, accompanied by an enhanced loss function based on maximum correntropy for improved robustness and performance. Ahmad et al. [24] introduced a

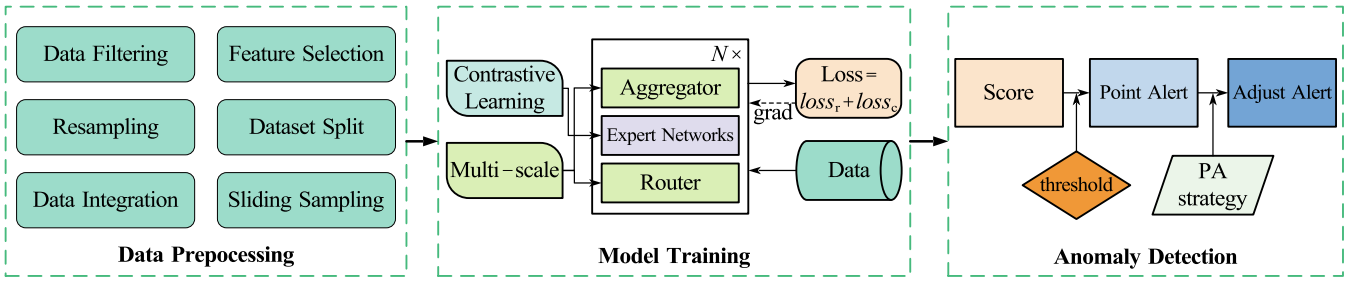

Fig. 2. Overview of proposed framework. It includes three parts: data prepocessing, model training, and anomaly detection.

Transformer-based framework for predicting and classifying UAV sensor faults, utilizing an innovative attention mechanism and deep learning techniques for early diagnosis and proactive prevention of sensor failures, thereby enhancing UAV flight safety and reliability.

Existing research shows that unsupervised anomaly detection methods for UAVs have produced significant results. However, certain issues still need to be addressed, particularly the challenge of temporal scale variations. To address this challenge, multi-scale modeling techniques need to be employed, which has been neglected in UAV anomaly detection.

B. Multi-Scale Modeling for Time Series

Multi-scale modeling has emerged as an important technique in time series analysis, such as wavelet transforms [37] and pyramid structures [38]. It extracts features from multiple time scales of a time series to obtain a richer representation than a single scale. Recently, many deep learning models have successfully integrated multi-scale techniques. Nie et al. [39] introduced PatchTST, a channel-independent patch time series Transformer that retains local semantic information while reducing computational complexity. The patching operation addresses the issue of insufficient semantic information and provides the foundation for multi-scale modeling. Wu et al. [40] proposed TimesNet, a CNN-based model that manages multi-periodicity in time series by decomposing complex temporal patterns into intraperiod and interperiod variations. Zhou et al. [41] introduced the frequency-enhanced decomposed Transformer (FEDformer), which combines Transformers with seasonal trend decomposition to capture both global profiles and detailed structures. Zhang et al. [42] presented the multi-resolution time series Transformer (MTST), featuring a multi-branch architecture that simultaneously models diverse temporal patterns at different resolutions.

However, most multi-scale models rely on fixed scales and cannot adapt to different data. Chen et al. [26] introduced Pathformer to address adaptive multi-scale modeling, but it overlooks the neighborhood relationships among patches, which is important for learning the temporal correlations [27]. Therefore, the multi-scale Transformers with contrastive learning model is proposed for UAV anomaly detection. The proposed model achieves adaptive multi-scale modeling, and can effectively extract the temporal correlations in flight data.

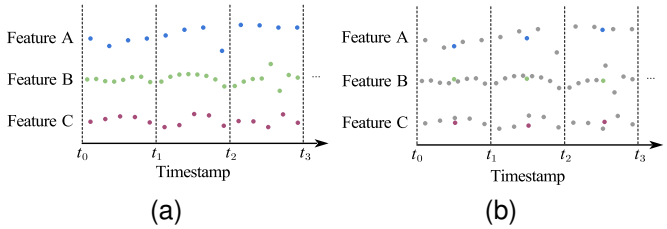

Fig. 3. Resampling of flight data. (a) Distribution of samples before sampling. (b) Sample distribution after average sampling.

III. Methodology

A. Overview of the Proposed Framework

The problem is considered as follows: Let $\mathbf{FD} \coloneqq \{\mathbf{F}_i\}_{i=1}^{n}$ represent a UAV flight dataset, where each flight $\mathbf{F}_i \in \mathbb{R}^{m \times t_i}$ consists of m time-dependent variables measured over t_i timestamps. The temporal dimension t_i varies across flights due to different durations. The dataset contains normal and abnormal flights. As shown in Fig. 2, the proposed framework comprises three main steps: data preprocessing, model training, and anomaly detection. During preprocessing, the raw flight data undergoes filtering, resampling, data integration, and feature selection to ensure high quality model input [7], [20]. Then, following the standard unsupervised anomaly detection procedure, the training set includes only normal data, while the test set includes normal and abnormal data. Thus, normal flights and parts of anomalous flights before faults can be selected as the training set as follows:

$$\mathbf{X}_{\text{train}} = \text{concat}(\mathbf{F}_1, \mathbf{F}_2, \cdots, \mathbf{F}_{n_{\text{train}}}), \tag{1}$$

where n_{train} is the number of flights in the training set. After concatenating, the training set can be written as $\mathbf{X}_{\text{train}} = \{\mathbf{X}_{\text{train}}^{i}\}_{i=1}^{T_{\text{train}}}$, where $\mathbf{X}_{\text{train}}^{i} \in \mathbb{R}^{m}$ is the value of the i-th timestamp and T_{train} is the number of timestamps in the training set. Similarly, the test set can be represented as $\mathbf{X}_{\text{test}} = \{\mathbf{X}_{\text{test}}^{i}\}_{i=1}^{T_{\text{test}}}$.

After performing data preprocessing, the model learns the distribution of normal data in the training set and performs anomaly detection in the test set. In anomaly detection, the output of our model is a set of binary labels $\mathbf{Y}_{\text{pred}} = \{y_t\}_{t=1}^{T_{\text{test}}}$ indicating whether each tick at the test time is an anomaly or not, that is, $y_t \in \{0, 1\}$, where $y_t = 1$ indicates that time t is anomalous.

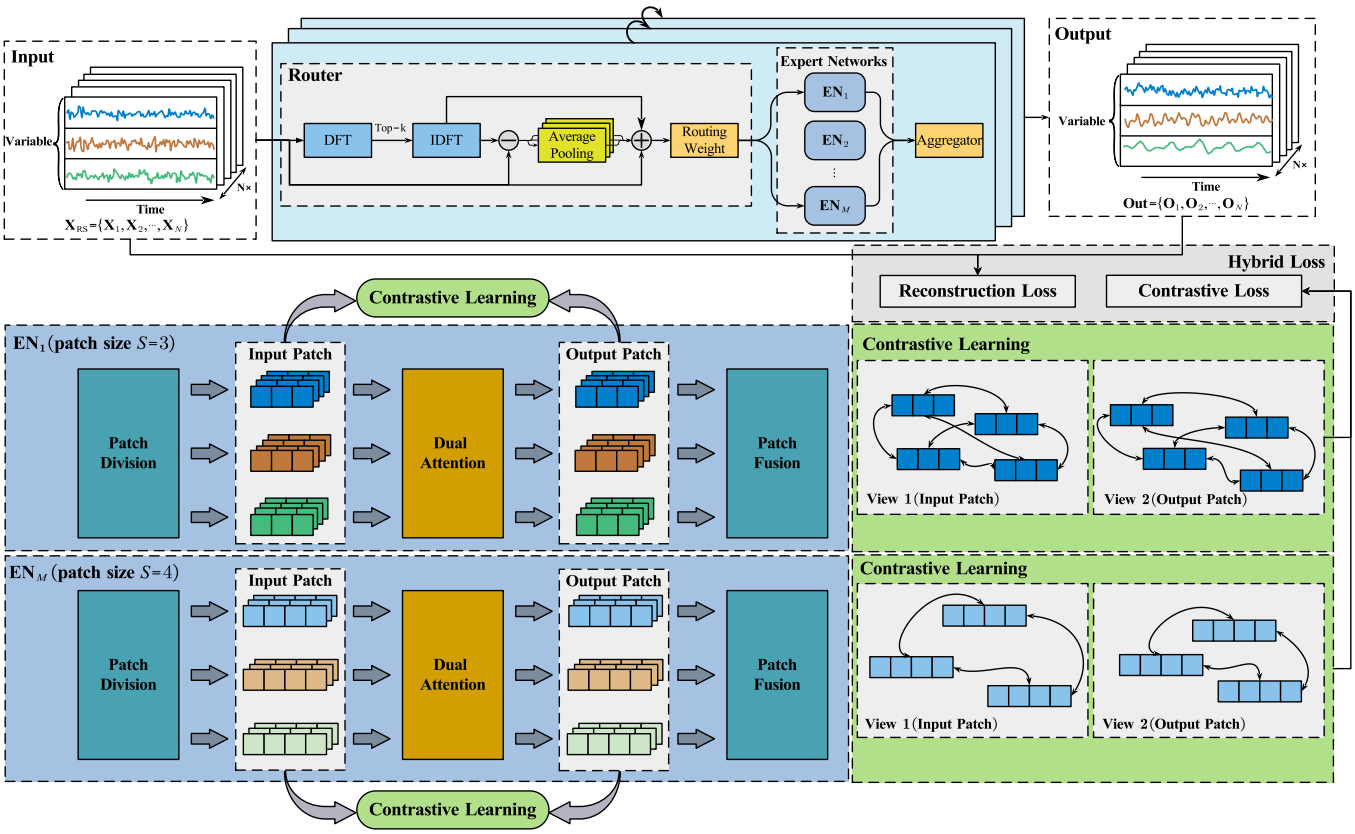

Fig. 4. The architecture of MTCL-UAV. A sparse MoE block comprises **Router**, **Expert Networks**, and **Aggregator**. The **Router** performs seasonal and trend decomposition on the input data, and adds the decomposed results to the input to enhance its periodicity and trend characteristics. After that, the optimal routing weights are obtained through the routing function. In each ENs with a specific scale, CL-DAM is used to extract temporal correlations from the data. The **Aggregator** performs a weighted sum based on the routing weights and outputs of ENs to obtain the output of MoE.

B. Data Preprocessing

Typically, a UAV system can generate a large amount of raw flight data. This raw data has the characteristics of high dimensionality, multivariate, high noise, missing data, heterogeneous data and irregular sampling, which cannot be fed directly into the model for training. Therefore, it is necessary to perform data preprocessing and feature engineering to enhance the quality of data and convert raw data into standard format data, which mainly includes:

1) Data Filtering: This process involves selecting data from the raw dataset that meets specific criteria. Specific variables unique to individual unmanned aerial vehicles were excluded to ensure compatibility with a wide range of UAV systems [43].

2) Resampling: Resampling is utilized on the data at a frequency of 5 Hz because the sampling frequency of the selected variables is inconsistent [20]. An average aggregation method is employed to perform resampling in each sensor data table to align the timestamps. By averaging multiple data points, the effect of random fluctuations and noise can be reduced, making the data trend smoother and more precise. The process is shown in Fig. 3.

3) Data Integration: Since the raw data is collected from multiple sensors, it is necessary to combine the data from different sources into a consistent dataset. Each table has a timestamp index, and the resampled timestamp is used as an index to combine each table. This merging is implemented in the form of a union.

4) Feature Selection: Flight data are high-dimensional multivariate time series collected from various sensors, but not all features are essential for model training. Thus, feature selection is crucial as it enhances model performance by reducing overfitting, improving interpretability, and accelerating training. In the experiments, the flight parameters adopted in [43] and [44] are applied to two flight datasets, respectively.

5) Dataset Split: Typically, anomaly detection models based on reconstruction loss are trained using normal data. Therefore, the training set contains only normal data to enable the model to learn normal patterns, and the test set contains normal and anomalous data with reconstruction loss as the anomaly score. In the paper, the UAV flight dataset **FD** is divided into a training set $\mathbf{X}_{\text{train}}$ and a test set $\mathbf{X}_{\text{test}}$, with the anomalous samples then removed from the training set. Considering cross-validation requirements, a part of the training set is selected as the validation set.

6) Sliding sampling: Sliding sampling involves moving a sampling window over the data, allowing for the continuous capture of data streams. A key advantage of sliding sampling is its ability to reduce computational load while still providing a strong representation of the underlying data. The sliding

window is applied to the training set $\mathbf{X}_{\text{train}}$ and the test set $\mathbf{X}_{\text{test}}$. The reconstructed training set can be expressed as follows,

$$\mathbf{X}_{\text{RS}} = \{\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_N\}, \mathbf{X}_i \in \mathbb{R}^{m \times L}, \tag{2}$$

where m is the number of variables, L represents the window size, and N indicates the total number of reconstructed samples gathered from all flight data. In the following, $\mathbf{X}$ is used to represent the i-th sample $\mathbf{X}_i$ in order to maintain a concise symbolic description.

C. Multi-Scale Transformers with Contrastive Learning

MTCL-UAV is built based on sparse mixture-of-experts (MoE) blocks, as illustrated in Fig. 4. Each MoE block consists of three primary components: **Router**, **Expert Networks**, and **Aggregator**. The Router begins by performing seasonal and trend decomposition on the input data, effectively extracting periodicity and trend patterns and incorporating them into the input. Based on the data characteristics, the **Router** selects the optimal scales and routes the input sample into the corresponding ENs. Within each EN, a dual attention mechanism is employed to capture both inter-patch and intra-patch correlation relationships. To enhance the model's ability to learn important patterns in flight data, a contrastive learning approach is introduced, effectively capturing neighborhood relationships among patches. Finally, the Aggregator combines the results of ENs to generate the final output of the MoE block. The details of each component within the MoE block are described below.

1) Router: Recently, patching time series has emerged as a prominent approach for effective segmentation and enhanced feature extraction. The choice of patch size is critical, as it directly impacts model efficiency and the ability to capture meaningful temporal patterns. Flight data are multivariate time series, and exhibit diverse variations and fluctuations across different temporal scales. To achieve adaptive multi-scale modeling, a Router based on frequency domain analysis techniques is employed to dynamically select optimal patch sizes for each sample and assign them to the corresponding experts, as shown in Fig. 4. The Router achieves automatic routing of data by optimizing patch sizes, enabling efficient multi-scale modeling and facilitating the analysis of information across various scales. This module performs seasonal and trend decomposition to transform the data from the temporal domain to the frequency domain to extract periodic patterns and improve the model's ability to analyze temporal dynamics.

Firstly, the discrete Fourier transform (DFT), denoted $\text{DFT}(\cdot)$, is performed to decompose the sample $\mathbf{X} \in \mathbb{R}^{m \times L}$ into Fourier basis, and the k_f basis with the largest amplitudes is selected as follows:

$$\mathbf{A}, \Phi = \text{Amp}\left(\text{DFT}\left(\mathbf{X}\right)\right), \text{Phase}\left(\text{DFT}\left(\mathbf{X}\right)\right), \tag{3}$$

where $\text{Amp}(\cdot)$ and $\text{Phase}(\cdot)$ denote the calculation of amplitude values and phase values, respectively. $A \in \mathbb{R}^{m \times L}$ and $\Phi \in \mathbb{R}^{m \times L}$ represent the amplitude and the phase of each frequency for each variable, respectively. The k_f basis with the largest amplitudes, $\{f_1, \cdots, f_{k_f}\}$, is selected to maintain the sparsity of the frequency domain. Each $f_i \in \mathbb{R}^{k_f}$ represents the relevant frequency of each variable. To obtain the seasonal pattern, the inverse DFT (IDFT) is used to transform the Fourier coefficients as follows:

$$\mathbf{X}_{\text{sea}} = \text{IDFT}\left(\left\{f_1, \ldots, f_{k_f}\right\}, \mathbf{A}, \Phi\right). \tag{4}$$

The trend patterns are extracted by the various kernel sizes of average pooling from the remaining data after seasonal decomposition $\mathbf{X}_{\text{rem}} = \mathbf{X} - \mathbf{X}_{\text{sea}}$. A weighted operation is used to represent the trend component based on the outcomes from different kernel sizes:

$$\mathbf{X}_{\text{trend}} = \text{Softmax}(L(\mathbf{X}_{\text{rem}})) \cdot (\text{Avg}_1(\mathbf{X}_{\text{rem}}), \cdots, \text{Avg}_C(\mathbf{X}_{\text{rem}})), \tag{5}$$

where $\text{Softmax}(z_i) = \exp(z_i)/\sum_j \exp(z_j)$, z_i represents the i-th element of the input vector z, $\text{Avg}_i(\cdot)$ is the average pooling function of the i-th kernel, C is the number of kernels, $L(\cdot)$ represents a linear layer. The output of the time decomposition module can then be obtained as follows [26],

$$\mathbf{X}_{\text{trans}} = L(\mathbf{X} + \mathbf{X}_{\text{sea}} + \mathbf{X}_{\text{trend}}), \tag{6}$$

where $\mathbf{X}_{\text{trans}} \in \mathbb{R}^L$.

Finally, the softmax function is employed to generate the routing weight of result $\mathbf{X}_{\text{trans}}$, which determines the ENs to be chosen for the current data. To enhance model generalization, a noise term is introduced during weight calculation to increase randomness. The process of generating weights can be formulated as follows,

$$R\left(\mathbf{X}_{\text{trans}}\right) = \text{Softmax}\left(\mathbf{X}_{\text{trans}} W_r + \epsilon \cdot \text{Softplus}\left(\mathbf{X}_{\text{trans}} W_n\right)\right), \tag{7}$$

where $\text{Softplus}(x) = \ln(1 + e^x)$, x is the input, $\epsilon \sim \mathcal{N}(0, 1)$, $\mathcal{N}(\cdot)$ is gaussian distribution, $R(\cdot)$ is the routing function, $\mathbf{W}_r$ and $\mathbf{W}_n \in \mathbb{R}^{L \times M}$ represent learnable parameters for weight generation, M is the number of expert networks. To introduce sparsity in the routing and encourage the selection of critical scales, the Top-k selection is performed on the routing weight, keeping the Top-k routing weight and setting the remaining weights at 0. The final result of Router is denoted as $\bar{R}\left(\mathbf{X}_{\text{trans}}\right)$.

2) Expert Networks: The Router selects optimal k scales for each sample and assigns it to the corresponding k ENs. Within each EN, a dual attention mechanism is employed to learn both intra- and inter-patch correlations. Additionally, a contrastive learning method is introduced to enhance the model's ability to detect subtle differences between patches. Specifically, the similarity relationships among patches in the input and output of the EN are treated as two views (View 1 and View 2). By comparing and aligning the consistency differences between these two views, the model is guided to capture more precise and meaningful features across patches.

Let M expert networks $\mathbf{E} = \{\text{EN}_1, \text{EN}_2, \cdots, \text{EN}_M\}$, and let the patch size of i-th EN be S_i. The collection of patch sizes $\{S_1, S_2 \cdots, S_M\}$ can be defined based on prior knowledge. For the i-th expert network, the patch size S_i is denoted as S to simplify the notation. Since channel-independent modeling is adopted in expert networks, embedding is performed on the v-th variable $\mathbf{X}[v, :]$, and it is denoted as $\mathbf{X}_v \in \mathbb{R}^{d \times L}$, where d is the dimension of features.

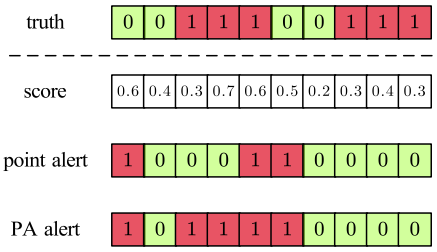

Fig. 5. Depiction of the point adjust strategy. Red for abnormal points and green for normal points.

Through the patch division operation with the patch size S, the input data $\mathbf{X}_v$ can be divided into P ($P = L/S$) patches as follows [39],

$$(\mathbf{X}_v^1, \mathbf{X}_v^2, \cdots, \mathbf{X}_v^P) = \text{Patch}_S(\mathbf{X}_v), \tag{8}$$

where $\mathbf{X}_v \in \mathbb{R}^{P\times S\times d}$), each patch ($\mathbf{X}_v^i \in \mathbb{R}^{S\times d}$) with S timestamps.

Intra-patch Attention. For the i-th patch, denoted as $\mathbf{X}_v^i$, the process begins with embedding the patch along the feature dimension d, resulting in $\mathbf{X}_{\text{intra}}^i \in \mathbb{R}^{S\times d_m}$, where d_m represents the dimensionality of the embedding. A trainable linear transformation is then applied to $\mathbf{X}_{\text{intra}}^i$ to obtain the key and value components required for the computation of attention, denoted as K_{intra}^i and V_{intra}^i. In addition, a query matrix $Q_{\text{intra}}^i \in \mathbb{R}^{1\times d_m}$ is used to aggregate the context of the patch. The cross-attention is subsequently computed among Q_{intra}^i, K_{intra}^i, and V_{intra}^i to extract the local features within the i-th patch. This attention mechanism can be expressed mathematically as follows:

$$\text{Attn}_{\text{intra}}^i = \text{Softmax}\left(Q_{\text{intra}}^i \left(K_{\text{intra}}^i\right)^T / \sqrt{d_m}\right) V_{\text{intra}}^i . \tag{9}$$

Upon completing the intra-patch attention process, each patch transforms from its initial length of S to a length of 1. The attention outputs from all patches are then concatenated to form the intra-attention output across the divided patches, represented as $\text{Attn}_{\text{intra}} \in \mathbb{R}^{P\times d_m}$, which encapsulates the local details from adjacent time steps within the time series:

$$\text{Attn}_{\text{intra}} = \text{Concat}\left(\text{Attn}_{\text{intra}}^1, \ldots, \text{Attn}_{\text{intra}}^P\right). \tag{10}$$

Inter-patch Attention. The inter-patch attention mechanism is structured to establish relationships between patches, thereby facilitating the capture of global correlations. For the time series segmented into patches, defined as $\mathbf{X}_v \in \mathbb{R}^{P\times S\times d}$, the embedding of the feature is executed from dimension d to d_m before the data is rearranged to combine the dimensions of the patch count S and feature embedding d_m. This results in $\mathbf{X}_{\text{inter}} \in \mathbb{R}^{P\times d_m^{'}}$, where $d_m^{'} = S \cdot d_m$. This embedding and rearrangement process enables us to integrate time steps within each patch and consequently apply self-attention to $\mathbf{X}_{\text{inter}}$ to model inter-patch correlations. To incorporate the sequential order of the patches, a learnable positional encoding is added to the transformed embedding. Specifically, a trainable positional encoding matrix $\mathbf{W}_{\text{pos}} \in \mathbb{R}^{P\times d_m'}$ is generated and added to $\mathbf{X}_{\text{inter}}$ as:

$$\mathbf{X}_{\text{inter}} = \mathbf{X}_{\text{inter}} + \mathbf{W}_{\text{pos}}. \tag{11}$$

This embedding allows the model to effectively capture the global temporal relationship among patches. Following the standard self-attention framework [45], the query, key, and value matrices are derived from $\mathbf{X}_{\text{inter}}$, denoted as Q_{inter}, K_{inter}, and $V_{\text{inter}} \in \mathbb{R}^{P\times d_m'}$. Subsequently, the attention $\text{Attn}_{\text{inter}}$ is computed, capturing interactions between patches and reflecting the global correlations of the time series:

$$\text{Attn}_{\text{inter}} = \text{Softmax}\left(Q_{\text{inter}} \left(K_{\text{inter}}\right)^T / \sqrt{d_m'}\right) V_{\text{inter}} . \tag{12}$$

Fusion. The global and local correlations acquired through the dual attention mechanism are fused by first rearranging the outputs from intra-patch attention into $\text{Attn}_{\text{intra}} \in \mathbb{R}^{P\times S\times d_m}$. This step entails performing linear transformations on the patch size dimension, transitioning from 1 to S to amalgamate the time steps within each patch. Finally, the output of inter-patch attention $\text{Attn}_{\text{inter}} \in \mathbb{R}^{P\times S\times d_m}$ is added to the transformed intra-patch attention results, yielding the final output of the dual attention mechanism, denoted as follows,

$$\mathbf{X}_{\text{out}} = \text{Attn}_{\text{inter}} + \text{Attn}_{\text{intra}}, \tag{13}$$

where $\mathbf{X}_{\text{out}} \in \mathbb{R}^{P\times S\times d_m}$, and i-th output patch $\mathbf{X}_{\text{out}}^i \in \mathbb{R}^{S\times d_m}$.

Contrastive Learning. The dual attention mechanism as an encoder for patches, enabling simultaneous attention to global and local data correlations. However, when processing highly complex or noisy data, the dual attention mechanism may overly depend on specific patterns in the training set, leading to model overfitting. To address this issue, a contrastive learning method is proposed to improve the model's generalization. The approach can be summarized as treating the dual attention mechanism as an encoder while maintaining the consistency of neighborhood relationships among patches before and after encoding. The details of this process are described below.

Considering the collection of input patches $(\mathbf{X}_v^1, \mathbf{X}_v^2, \ldots, \mathbf{X}_v^P)$, the pairwise similarity between these patches is determined and used as the weights in the following manner:

$$W_{pq} = \exp\left(\text{sim}\left(\mathbf{X}_v^i, \mathbf{X}_v^j\right)/t\right), \tag{14}$$

where $\text{sim}(\mathbf{X}_v^p, \mathbf{X}_v^q)$ represents the similarity function such as Euclid distance (ED), cosine distance (CD) and dynamic time warping (DTW), t is the temperature parameter.

After encoding with the dual attention mechanism, the set of output patches $(\mathbf{X}_{\text{out}}^1, \mathbf{X}_{\text{out}}^2, \cdots, \mathbf{X}_{\text{out}}^P)$ is obtained. Thus, the contrastive loss of j-th EN in i-th block can be defined as

$$Loss_{\text{c}}^{ij} = \text{Avg}(\sum_{p=1}^{P} \sum_{q=1, p\neq q}^{P} \text{sim}(\mathbf{X}_{\text{out}}^p, \mathbf{X}_{\text{out}}^q) W_{pq}). \tag{15}$$

The contrastive loss of our model $Loss_{\text{c}}$ is the average of all $Loss_{\text{c}}^{ij}$ for each EN in layers.

3) Aggregator: The Router selects k ENs based on the routing result $\bar{R}(\mathbf{X}_{\text{trans}}) \in \mathbb{R}^M$, which divides the input $\mathbf{X}$ according to its temporal characteristics. Here, $\bar{R}(\mathbf{X}_{\text{trans}})_i$ represents the weight of the i-th dimension corresponding to a specific expert network, and $\bar{R}(\mathbf{X}_{\text{trans}})_i > 0$ indicates that the sample is assigned to the i-th expert network. Subsequently, the Aggregator performs a weighted aggregation of the outputs

TABLE I
DETAILED INFORMATION OF ALFA DATASET.

Fault Type	Number of Flight	Time Before Fault(s)	Time After Fault(s)
Engine Full Power Loss	23	2282	362
Rudder Stuck to Left	1	60	9
Rudder Stuck to Right	2	107	32
Elevator Stuck at Zero	2	181	23
Left Aileron Stuck at Zero	3	228	183
Right Aileron Stuck at Zero	4	442	231
Both Ailerons Stuck at Zero	1	66	36
Rudder & Aileron at Zero	1	116	27
No Fault	10	558	–
Total	47	3935	777

from the selected ENs using the routing weights. This yields the final output of the block:

$$\mathbf{X}_{\text{out}} = \sum_{i=1}^{M} \bar{R}\left(\mathbf{X}_{\text{trans}}\right)_i \mathbf{X}_{\text{out}}^i, \tag{16}$$

where $\mathbf{X}_{\text{out}}^i$ is the output of i-th EN with the patch size S_i.

For each sample $\mathbf{X}$, the final output of our model is denoted as $\mathbf{O} \in \mathbb{R}^{m \times L}$, which is also the output of the last sparse MoE block after residual connection. The reconstruction loss of the model can be defined as

$$Loss_r = \text{MSE}(\mathbf{X}, \mathbf{O}), \tag{17}$$

where $\text{MSE}(\cdot)$ is the mean squared error function.

4) Hybrid Loss Function: The loss function employed in MTCL-UAV consists of two fundamental components: reconstruction loss and contrastive loss. Reconstruction loss quantitatively assesses the model's ability to accurately reconstruct input data, ensuring the preservation of structural integrity and critical information, which is essential for capturing intricate details. Meanwhile, contrastive loss considerably enhances the model's discriminative capacity by enabling the learning of relationships among samples, thus promoting the development of a robust and easily identifiable feature representation.

This hybrid loss function can be represented mathematically as follows:

$$Loss = Loss_{\text{r}} + \lambda \cdot Loss_{\text{c}}, \tag{18}$$

where λ is a hyperparameter that regulates the trade-off between reconstruction loss and contrastive loss, enabling a nuanced adjustment based on the specific requirements of the task.

By harmonizing the goals of reconstruction and contrastive learning, the hybrid loss function enhances the model's performance while facilitating a balanced approach to maintaining global structure and distinguishing local features.

D. Anomaly Detection

Anomaly detection methods are based on a fundamental assumption: the distribution of abnormal samples differs significantly from that of normal samples. Consequently, the model is trained using normal data to facilitate unsupervised anomaly detection. In the trained model, the reconstruction loss in Eq. 17 serves as the anomaly score for each sample in the test set $\mathbf{X}_{\text{test}}$. Subsequently, the anomaly threshold is determined based on the distribution of anomaly scores:

$$\tau = \text{Percentile}(\Lambda, p), \tag{19}$$

where $\Lambda = \{\alpha_i\}_{i=1}^{T_{\text{test}}}$ represents the set of anomaly scores, $\text{Percentile}(a, b)$ is a function that calculates the bth percentile of the set a. The output of the model can be computed as follows:

$$\mathbf{Y}_{\text{pred}} = \{y_t\}_{t=1}^{T_{\text{test}}}, y_t = 0 \; if \; \alpha_t < \tau \; else \; 1. \tag{20}$$

In practical applications, human operators generally do not focus on point-wise metrics. An algorithm may trigger an alert at any point in a contiguous anomaly segment, as long as the delay is reasonable. Therefore, the point-by-point detection strategy needs to be improved to make it more realistic for evaluating the model. Specifically, a point adjustment (PA) strategy is adopted: If a chosen threshold can detect any point within an anomaly segment in the ground truth, that segment is labeled as correctly detected, with all points within this segment considered identifiable by this threshold. Meanwhile, points outside the anomaly segments are treated as usual. This approach is illustrated in Fig. 5. The top row represents the ground truth, which consists of 10 consecutive points with two highlighted anomaly segments enclosed in shaded squares. In the second row, the scores from the detector are displayed below. The third row presents the results of point-wise detection using a threshold value of 0.5. After adjustments, the fourth row illustrates the modified detector outcomes. Consequently, a precision of 0.6 and a recall of 0.5 are calculated. In particular, from the third row, it can be seen that the alert delay for the initial segment is 1 interval. With the PA strategy, the output of the test set is adjusted to obtain the adjustment output $\tilde{\mathbf{Y}}_{\text{pred}}$.

E. Complexity analysis

The MTCL-UAV model leverages a mixture-of-experts (MoE) architecture, where Expert Networks (ENs) can be computed in parallel to reduce the overall runtime. However, the computational complexity is still heavily influenced by the operations within each EN. Among these, the inter-patch attention mechanism dominates, with a quadratic complexity $O(P^2)$ in terms of the number of patches P (where $P = L/S$). This quadratic dependency becomes particularly significant when processing long sequences, making global self-attention a primary bottleneck. Additionally, in the contrastive learning module, the computation of pairwise similarities between patches $\mathbf{X}_v^i$ and $\mathbf{X}_v^j$, such as using ED, further adds to the computational burden, also scaling with $O(P^2)$.

IV. EXPERIMENTS

In this section, a real UAV flight dataset is used to verify the performance of the proposed model. Some classical and state-of-the-art models are selected as baselines for the comparison experiment. In addition, the ablation study and the parameter sensitivity analysis are conducted to comprehensively evaluate the robustness of the proposed model. Finally, a visualization

TABLE II
THE COMPOSITION OF DATASETS.

Dataset	Subset	Flights	Timestamps	Test Set Composition
D1	Training	28	10512	6 Eng., 1 Elev., 6 Ail.
	Valid	5	1985	
	Test	13	6286	
D2	Training	31	11401	5 Eng., 3 Rud., 2 Ail.
	Valid	5	1208	
	Test	10	4550	
D3	Training	32	11808	7 Eng., 1 Elev., 1 Ail.
	Valid	5	1309	
	Test	9	4s095	
D4	Training	28	10687	11 Eng., 1 Rud., 1 Ail.
	Valid	5	1029	
	Test	13	6238	
D5	Training	28	10953	10 Eng., 2 Rud., 2 Ail.
	Valid	4	846	
	Test	14	5730	
U1	Training	7	107968	July 17, July 21, July 23
	Valid	1	26991	
	Test	3	95780	
U2	Training	7	99025	July 3, July 10, July 13
	Valid	1	24756	
	Test	3	89370	

• Eng.: Engine, Elev.: Elevator, Rud.: Rudder, Ail.: Aileron.

TABLE III
MODEL PARAMETERS CONFIGURATION.

Type	Parameter	Value
Learning Parameters	Learning rate	1E-04
	Epoch	100
	Patience	10
	Batch size	128
	Dropout	0.1
Structure Parameters	Window size L	96
	Dimension of d_m	16
	Number of blocks	3
	K largest fourier basis	3
	Trend kernel size	[4,8,12]
	Patch pool	[2,4,6,8,12,16,32]
	Number of ENs	4
	Top-k ENs selected	2
	Temperature t	200
	Anomaly ratio p	5.0%
	Loss weight λ	1

is performed to validate the necessity of employing multi-scale modeling and the effectiveness of the proposed model.

A. Data Description

In the experiments, two flight datasets, the AirLab Failure and Anomaly (ALFA) dataset [25] and the Fixed-Wing UAV (FW-UAV) dataset [44], are used to evaluate our model and others. The descriptions of two datasets are as follows.

The ALFA dataset is presented by Keipour et al. [25], [46] and was collected from a fixed-wing Carbon-Z T-28 UAV at the experimental site of an airport near Pittsburgh, Pennsylvania. It includes processed data from 47 autonomous flights with scenarios for eight different types of control surface failure (actuator and engine), totaling 66 minutes of flight under normal conditions and 13 minutes of post-failure flight time. TABLE I shows the detailed information about the ALFA dataset. Each flight has a record of whether the UAV is in a fault state at each timestamp as a ground truth. In addition, the first flight has no ground truth information, so there are 46 flights available. In the dataset, faults on different control surfaces have different effects on the state of the UAV, some faults may be present for a shorter period of time and others for a longer period of time. In addition, there are scale variations in normal flight data that are affected by the environment and mission.

The FW-UAV dataset is presented by Bronz [44] and was collected from a small fixed-wing UAV. It includes 11 flights and the faults were manually injected into the actual task to simulate the faults occurring in a real flight. The actuator fault model proposed by Tandale [47] was used in the experiment:

$$\begin{bmatrix} u_{\mathrm{aR}} \\ u_{\mathrm{aL}} \end{bmatrix} = \begin{bmatrix} d_{\mathrm{R}}, 0 \\ 0, d_{\mathrm{L}} \end{bmatrix} \begin{bmatrix} u_{\mathrm{cR}} \\ u_{\mathrm{cL}} \end{bmatrix} + \begin{bmatrix} e_{\mathrm{R}} \ e_{\mathrm{L}} \end{bmatrix}, \quad (21)$$

where the subscripts R and L represent the right and left control surfaces, respectively, u_{a} is the final control deviation, u_{c} is intended deviation of surface control, e is a noise term, d is the control plane efficiency failure.

To evaluate model performance, the dataset is divided into training and test sets, with a portion of the training set serving as the validation set. Unsupervised anomaly detection methods are typically trained on normal data and tested on test set containing both normal and abnormal samples. Traditional cross-validation randomly divides samples into training or testing sets, which may result in insufficient normal samples in the training set or insufficient abnormal samples in the test set. Thus, a modified cross-validation approach is adopted: a portion of abnormal flights is randomly selected as the test set, while the remaining abnormal flights and all normal flights form the training and validation sets. Abnormal samples are excluded from both the training and validation sets. In this way, seven datasets are obtained as shown in TABLE II. These datasets are all complete datasets rather than subsets, differing only in the sample distributions of their training and test sets. This approach enables a more comprehensive evaluation of model performance on the anomaly detection task. D1 to D5 are from the AFLA dataset, and U1 and U2 are from the FW-UAV dataset. The composition of the ALFA dataset test set gives the type of abnormal sample fault in the test set, while the FW-UAV gives the flight date of the test set sample.

B. Implement Details and Evaluation Metrics

1) Implement Details: Experiments are conducted using PyTorch 2.1.2 and Python 3.10 on an NVIDIA 3090 24GB GPU. MTCL-UAV employs the Adam optimizer [48] with a learning rate of 10^{-4} and incorporates early stopping within 10 training epochs. The details of the related parameters are given in TABLE III. The trend kernel size is a sequence that indicates the pooling kernel sizes used for detrending. In our model, each sparse MoE block includes four ENs with patch sizes selected from the patch pool.

TABLE IV
THE ANOMALY DETECTION RESULTS IN THE ALFA DATASET.

Dataset	Metrics	MTCL-UAV	DCdetector	Anomaly*	TimesNet	PatchTST	FEDformer	Autoformer	CGAD	AutoEncoder	XGBoost	IForest	OCSVM
D1	*Accuracy*	0.9384	0.9343	**0.9470**	0.9070	0.9151	0.8758	0.8906	0.8862	0.9035	0.9042	0.8864	0.8026
	Precision	**0.9244**	0.8717	0.8788	0.8915	0.9057	0.8808	0.8870	0.7753	0.8490	0.8514	0.8229	0.8560
	Recall	0.8417	**0.8872**	0.8410	0.7460	0.7648	0.6241	0.6813	0.8225	0.7867	0.7867	0.7444	0.3338
	F1-score	**0.8811**	0.8794	0.8595	0.8123	0.8293	0.7306	0.7707	0.7982	0.8167	0.8178	0.7817	0.4803
D2	*Accuracy*	0.9470	0.9135	0.9409	0.9266	**0.9512**	0.9046	0.9119	0.9078	0.9315	0.9308	0.9228	0.9152
	Precision	**0.7481**	0.6799	0.7189	0.7385	0.7430	0.7198	0.7428	0.6310	0.7111	0.7083	0.7087	0.7293
	Recall	**0.9152**	0.6950	0.9017	0.7177	0.8521	0.4954	0.5449	0.8653	0.8811	0.8811	0.7852	0.6505
	F1-score	**0.8233**	0.6874	0.8000	0.7279	0.7938	0.5869	0.6287	0.7298	0.7870	0.7853	0.7450	0.6876
D3	*Accuracy*	**0.9299**	0.9148	0.9263	0.9072	0.9129	0.8797	0.8828	0.8247	0.9215	0.8676	0.8674	0.8649
	Precision	**0.8047**	0.7310	0.7358	0.7482	0.7752	0.7331	0.7379	0.4870	0.7265	0.6238	0.6229	0.6118
	Recall	0.7136	0.7370	0.8619	0.6296	0.6386	0.3856	0.4104	**1.0000**	0.7160	0.5086	0.5086	0.5086
	F1-score	0.7564	0.7340	**0.7939**	0.6838	0.7003	0.5054	0.5274	0.6550	0.7212	0.5603	0.5599	0.5554
D4	*Accuracy*	0.9256	**0.9261**	0.8981	0.8991	0.9076	0.8599	0.8726	0.8755	0.8906	0.8385	0.8379	0.8576
	Precision	**0.8404**	0.8213	0.7882	0.8216	0.8260	0.8017	0.8157	0.7214	0.7214	0.6332	0.6343	0.7005
	Recall	**0.7965**	0.7938	0.7137	0.6732	0.7181	0.4559	0.5202	0.7004	0.7331	0.6152	0.6053	0.6053
	F1-score	**0.8178**	0.8073	0.7491	0.7400	0.7683	0.5813	0.6353	0.7107	0.7636	0.6241	0.6195	0.6494
D5	*Accuracy*	**0.9372**	0.9174	0.9187	0.9043	0.9229	0.8758	0.8899	0.7923	0.8792	0.8745	0.8884	0.8615
	Precision	**0.7737**	0.7109	0.6959	0.6925	0.7582	0.6309	0.6955	0.4184	0.5864	0.5855	0.6164	0.5437
	Recall	0.7946	0.7325	0.7800	0.6200	0.6931	0.3602	0.4381	**0.9615**	0.6734	0.5737	0.6876	0.5059
	F1-score	**0.7840**	0.7215	0.7355	0.6542	0.7242	0.4586	0.5376	0.5830	0.6269	0.5795	0.6500	0.5241
Avg F1		**0.8125**	0.7682	0.7876	0.7236	0.7632	0.5726	0.6199	0.6953	0.7431	0.6734	0.6712	0.5794

• **Anomaly*** denotes Anomaly Transformer.

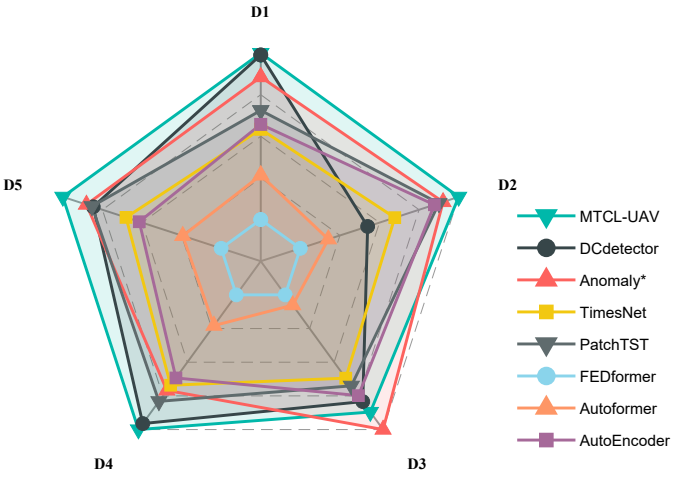

Fig. 6. Radar chart for normalized F1 scores.

2) Evaluation Metrics: To comprehensively evaluate the performance of anomaly detection models, four widely-recognized evaluation metrics are employed: *Accuracy*, *Precision*, *Recall*, and *F1-score*. In addition, the receiver operating characteristic (ROC) and area under the curve (AUC) are provided in the comparison experiment. These metrics provide complementary perspectives on the anomaly detection capabilities and collectively offer a thorough assessment of the effectiveness of the model. Each metric captures different aspects of the detection performance, allowing for a balanced and comprehensive evaluation of the methods.

C. Baselines

To verify the performance of the proposed model, some time series anomaly detection models, including deep learning methods and machine learning methods, are selected as baselines. Some of these have been used in UAV anomaly detection, such as AE [22], XGBoost, IForest [16] and OCSVM [49], [50]. The following is an overview of other models.

- **DCdetector [27]** uses a dual attention asymmetric design and pure contrastive loss to learn a permutation invariant representation with strong discrimination capabilities.
- **Anomaly Transformer [28]** introduces an anomaly attention mechanism for association discrepancy and employs a minimax strategy to enhance the distinction between normal and abnormal.
- **TimesNet [40]** transforms 1D time series into 2D tensors with TimesBlock, capturing intra- and inter-period variations for efficient anomaly detection through multi-scale feature extraction.
- **PatchTST [39]** is a patch-based method for multivariate time series that preserves local semantics and improves long-term dependency capture, reducing complexity and enhancing modeling effectiveness.
- **FEDformer [41]** integrates Transformers with seasonal-trend decomposition, allowing the decomposition method to capture the overall time series profile while Transformers focus on detailed structures.
- **Autoformer [51]** employs auto-correlation and decomposition to model time series, capturing long-term dependencies and seasonal patterns effectively.
- **CGAD [52]** utilizes transfer entropy to construct graph structures that unveil the underlying causal relationships among time series data, and models both the causal graph

TABLE V
THE ANOMALY DETECTION RESULTS IN THE FW-UAV DATASET.

Dataset	Metrics	MTCL-UAV	DCdetector	Anomaly*	TimesNet	PatchTST	FEDformer	Autoformer	CGAD	AutoEncoder	XGBoost	IForest	OCSVM
U1	*Accuracy*	0.9807	0.9662	0.9714	0.9555	0.9699	0.9793	0.9709	**0.9986**	0.9875	0.9190	0.9823	0.9754
	Precision	0.9641	0.9437	0.9418	0.9253	0.9643	**0.9823**	0.9804	0.9733	0.9737	0.8509	0.9630	0.9495
	Recall	0.9954	0.9858	**1.0000**	0.9833	0.9708	0.9728	0.9562	0.9826	0.9831	**1.0000**	0.9931	0.9945
	F1-score	**0.9795**	0.9643	0.9700	0.9534	0.9675	0.9775	0.9682	0.9779	0.9784	0.9194	0.9778	0.9715
U2	*Accuracy*	0.9583	0.9546	0.9739	0.9296	0.9474	0.9432	0.9429	0.9540	0.9399	**0.9761**	0.8934	0.8879
	Precision	0.8859	0.8906	0.9196	0.8233	0.8668	0.8956	0.8928	0.9124	0.8323	**0.9259**	0.7368	0.7270
	Recall	0.9878	0.9669	0.9463	0.9733	0.9735	0.9168	0.9192	0.9358	**1.0000**	0.9153	**1.0000**	**1.0000**
	F1-score	**0.9341**	0.9272	0.9328	0.8920	0.9171	0.9061	0.9058	0.9240	0.9085	0.9206	0.8485	0.8419
Avg F1		**0.9568**	0.9458	0.9514	0.9227	0.9423	0.9418	0.9370	0.9509	0.9434	0.9200	0.9131	0.9067

• **Anomaly*** denotes Anomaly Transformer.

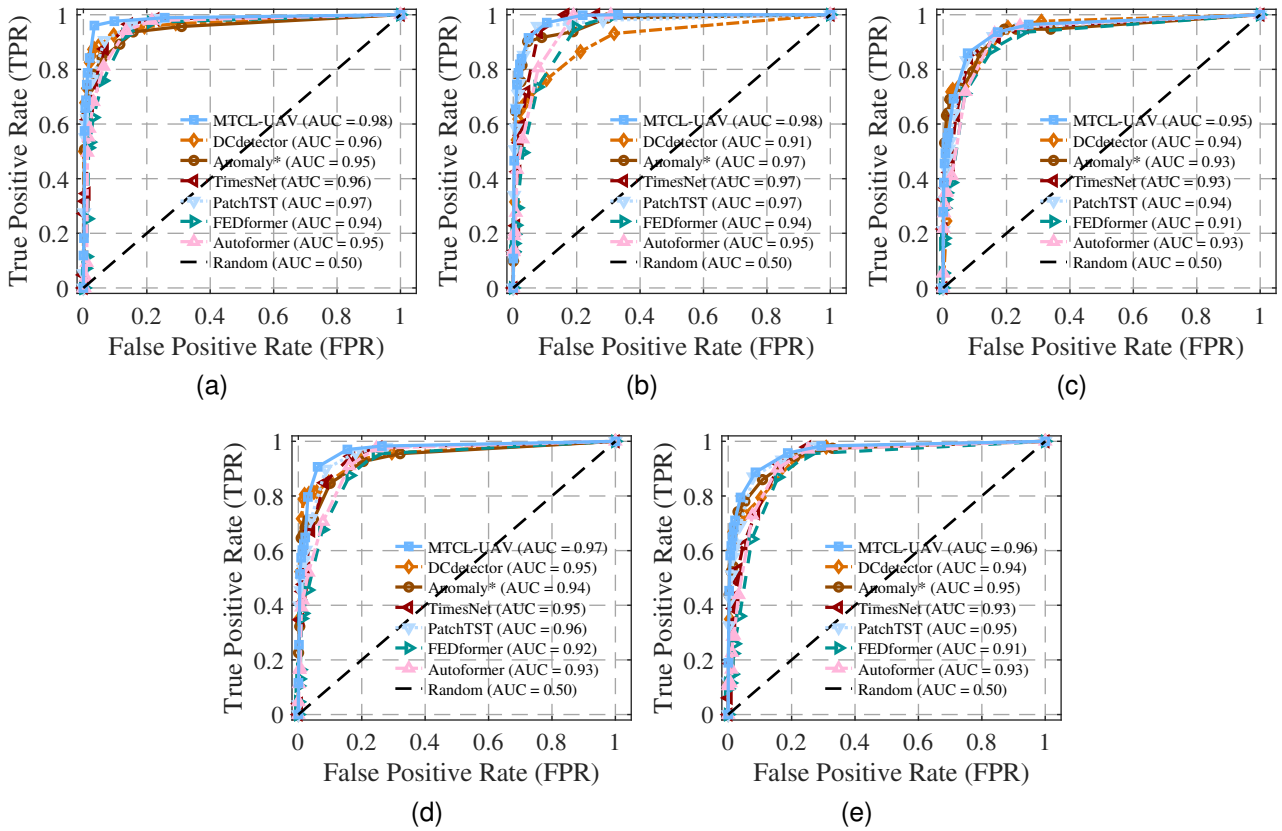

Fig. 7. ROC curves (horizontal-axis: false-positive rate; vertical-axis: true-positive rate) for D1-D5 datasets. A higher AUC value (area under the ROC curve) indicates better performance. The predefined threshold percentile p is in $\{0.1\%, 0.5\%, 1.0\%, 1.5\%, 2.0\%, 2.5\%, 3.0\%, 5.0\%, 10.0\%, 20.0\%, 30.0\%\}$

structures and the temporal patterns within multivariate time series data.

D. Performance Comparison

The effectiveness of MTCL-UAV was evaluated by comparing it against several competitive baselines in seven UAV flight datasets. The results are summarized in TABLE IV and TABLE V, with the best results highlighted in bold and the second best highlighted. To assess overall model performance, the F1 scores from the D1-D5 dataset are normalized and visualized on the radar chart shown in Fig. 6. Furthermore, all deep learning models were provided with the same input as MTCL-UAV. The results demonstrate that MTCL-UAV achieves superior performance in most datasets, recording the highest average F1 score of 0.8125 in D1-D5 datasets, significantly outperforming traditional and other deep learning-based anomaly detection methods. This highlights the robustness of our approach in handling UAV flight data with diverse distribution characteristics. For a more comprehensive evaluation, ROC curves are plotted and the value of AUC is given in Fig. 7. In these figures, the horizontal and vertical axes represent the false positive rate and true positive rate at various thresholds, respectively. AUC is a comprehensive metric, with higher values indicating better performance of the model. It can be seen that MTCL-UAV achieves the highest AUC values in all five datasets, demonstrating its effectiveness in balancing false positive and true positive rates under varying thresholds. Furthermore, MTCL-UAV maintains a low missed

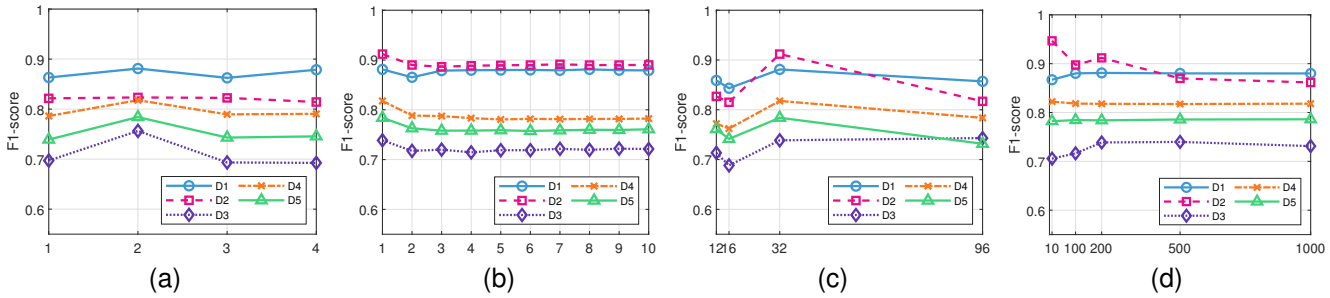

Fig. 8. Parameter sensitivity results. (a) The number of Top-k ENs. (b) The loss weight λ. (c) The maximum patch size $S_{\max}$. (d) The temperature t.

TABLE VI
RESULTS OF D1 WITH NOISY DATA.

Noise type	Metrics	1.0%	5.0%	10.0%
Gaussian	*Accuracy*	0.9279	0.9248	0.9222
	Precision	0.7930	0.7850	0.7842
	Recall	0.9935	0.9938	0.9841
	F1-score	0.8820	0.8771	0.8729
Noise type	**Metrics**	**[-0.5,0.5]**	**[-1,1]**	**[-5,5]**
Uniform	*Accuracy*	0.9335	0.9335	0.9242
	Precision	0.9238	0.9238	0.7854
	Recall	0.8227	0.8225	0.9916
	F1-score	0.8703	0.8702	0.8765

detection rate while achieving high precision, underscoring its reliability and practicality in UAV anomaly detection tasks.

In D1-D5 datasets, deep learning methods generally outperform traditional machine learning methods, mainly due to their powerful feature extraction capabilities and the ability to capture complex spatio-temporal dependencies. While OCSVM and ensemble learning-based methods (XGBoost and IForest) perform well in some tasks, they often struggle with challenges such as complex feature engineering and limited generalization when dealing with high-dimensional and complex time series data. In contrast, deep learning methods automatically learn and extract high-order features through multi-layer neural networks, significantly improving the recognition of abnormal patterns and achieving higher accuracy and robustness in UAV anomaly detection. From TABLE IV, it is evident that FEDformer and Autoformer have lower recall rates, indicating higher missed detection rates due to overfitting in normal data. CGAD combines causal convolutional and graph convolutional networks to effectively capture dynamic patterns in flight data, and has a good performance in recall. TimesNet, which combines multi-scale modeling with CNN to capture intra- and inter-period variations, effectively captures multi-level temporal dependencies but falls short in capturing global dependencies compared to Transformer models. PatchTST addresses the limitations of single-point modeling in traditional Transformers by using patching to enhance the granularity of feature extraction. However, it lacks multi-scale modeling to capture the complex patterns in flight data. The strong performance of Anomaly Transformer and DCdetector can be attributed to their implementation of contrastive learning, which improves feature discrimination.

In the U1 and U2 datasets, MTCL-UAV also achieves a balance between high precision and high recall compared to other models and achieves the highest F1 score of 0.9568. It should be noted that machine learning methods show performance comparable to that of deep learning methods in both datasets, and their recall rates are generally higher. The reason is that the control surface effectiveness reduction type faults are injected in the FW-UAV dataset, and they are injected in a linear manner. This leads to noticeable changes in angular velocity and acceleration, making it easy for anomaly detection methods to catch such anomalies.

The results of comparative experiments demonstrate that MTCL-UAV outperforms existing methods, highlighting its ability to effectively capture critical patterns in flight data and achieve accurate modeling. Unlike other multi-scale modeling approaches, MTCL-UAV adaptively selects optimal scales based on the temporal variation characteristics of the samples, enabling efficient multi-scale modeling. Furthermore, the incorporation of contrastive learning improves the model's ability to extract comprehensive temporal correlations. In general, MTCL-UAV achieves superior accuracy in UAV anomaly detection.

E. Sensitivity and Robustness Analysis

1) Sensitivity Analysis: Fig. 8 presents the results of our sensitivity analysis on four key hyperparameters in the ALFA dataset: the number of Top-k Expert Networks (ENs), the loss weight λ, the maximum patch size $S_{\max}$, and the temperature t. Our analysis shows that the use of two experts achieves the best overall performance, with further increases in k yielding only marginal improvements or slight declines. This suggests that a moderate number of experts effectively capture the diversity in UAV flight patterns. For the loss weight λ, the model remains stable across a wide range of values, with optimal performance at $\lambda = 1$, emphasizing the importance of balancing the loss components. Regarding the maximum patch size, $S_{\max} = 32$ delivers an F1 score of 0.9119, striking a balance between temporal segmentation and model accuracy. The temperature parameter t exhibits significant sensitivity at lower values; at $t = 10$, the model performs consistently well in datasets, with diminishing returns as t increases. The complexity of the model was evaluated by counting the time

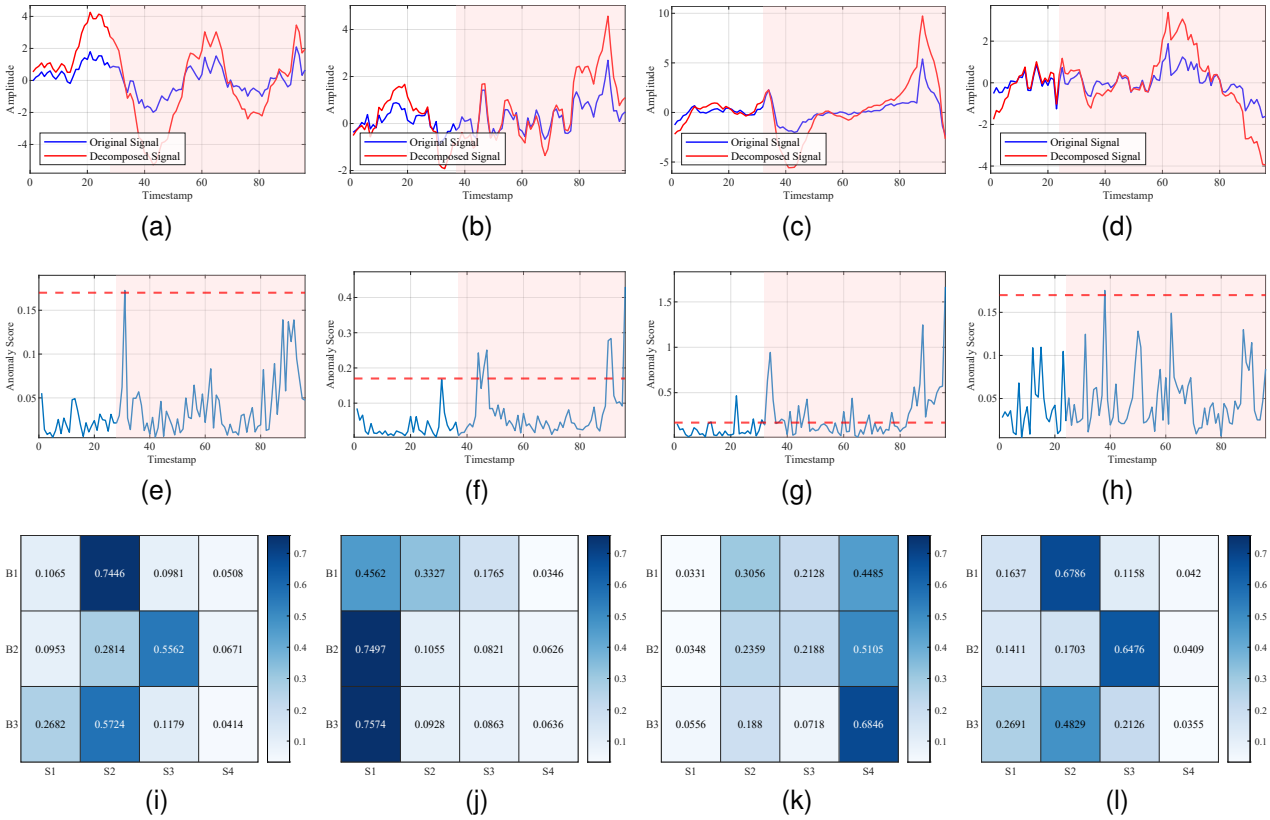

Fig. 9. Visual comparison of ground truth anomalies, anomaly scores and routing weights for different types of anomalies in the D1 dataset. (a)-(d): original and decomposed data. (e)-(h): normalized anomaly scores. (i)-(l): routing weight. The four types of anomalies are engine, elevator, rudder, and aileron. The light red areas represent anomalous areas.

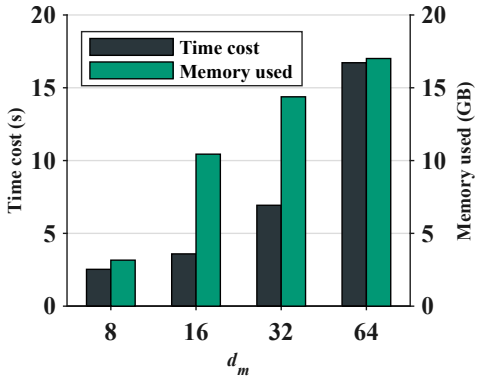

Fig. 10. The averaged GPU memory cost and the averaged running time of 100 iterations during training with different d_{model}.

cost and memory cost for different d_m, as shown in Fig. 10. Based on these findings, an embedding dimension of $d_m = 16$ was selected to optimize the trade-off between performance and complexity, ensuring efficient operation with minimal memory consumption.

In general, although performance variations across different parameters were observed, these differences were not statistically significant. In practical applications, adjusting key hyperparameters such as the maximum patch size S_{max} and the temperature t based on the characteristics of the data is crucial for optimal results.

2) Robustness Analysis: UAVs often operate in complex and dynamic environments, where the sensors onboard are exposed to harsh conditions, leading to noisy flight data. To evaluate the robustness of the model under such conditions, two types of noise, Gaussian and uniform, were introduced into the D1 dataset. Gaussian noise was tested with standard deviations of 1.0%, 5.0% and 10.0%, while uniform noise was applied with intervals of ± 0.5, ± 1.0 and ± 5.0. As shown in TABLE VI, while the accuracy of MTCL-UAV decreases slightly as the noise levels increase, it maintains strong performance. Even at the highest level of Gaussian noise (10. 0%), the F1 score drops by only 0.82%, demonstrating the robustness of the model in UAV anomaly detection tasks under noisy conditions.

F. Ablation Study

To evaluate the contributions of various modules in MTCL-UAV, ablation studies were conducted on the multi-scale mechanism, the Router, and the CL-DAM. The "W/O multi-scale mechanism" configuration employs single-scale modeling, utilizing a single expert network (with the largest patch size) for feature extraction. The "W/O Router" setup removes adaptive multi-scale selection, instead activating all expert networks (ENs) in each sparse MoE block for every dataset. The "W/O CL-DAM" variant excludes contrastive learning, retaining only the dual attention mechanism within the expert

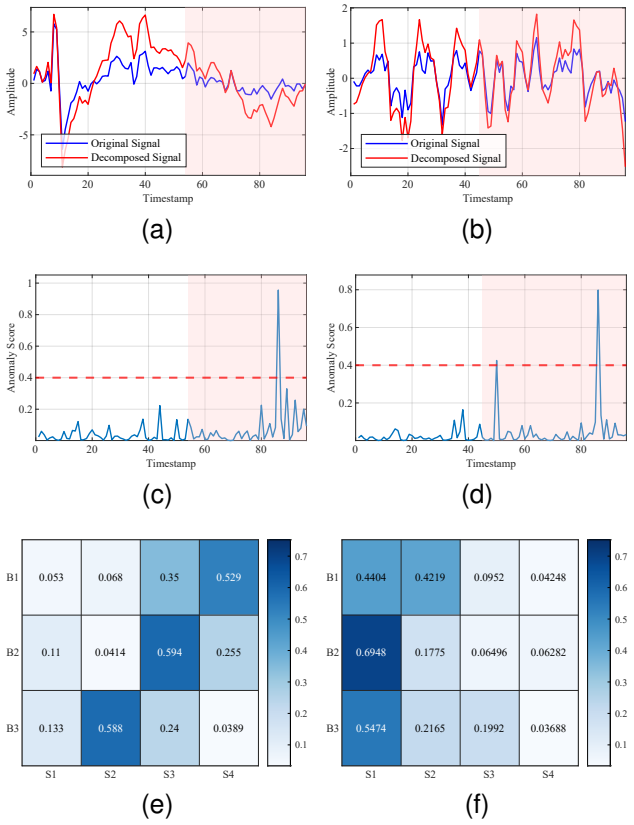

Fig. 11. Visual comparison of ground truth anomalies, anomaly scores and routing weights for different samples in the U1 dataset. (a)-(b): original and decomposed data. (c)-(d): normalized anomaly scores. (e)-(f): routing weight. The light red areas represent anomalous areas.

network. TABLE VII summarizes the results, highlighting the unique contributions of each module.

The ablation results reveal that the multi-scale mechanism significantly enhances performance. However, simple multi-scale modeling without the Router does not yield notable improvements, emphasizing the importance of frequency domain analysis in selecting appropriate scales. MTCL-UAV's adaptive multi-scale modeling ensures efficient feature extraction by dynamically activating expert networks based on flight data characteristics, even selectively engaging specific networks for anomaly detection when necessary. Furthermore, the model's performance declines when the dual attention mechanism operates without contrastive learning, confirming that CL-DAM effectively captures neighborhood relationships among patches and plays a critical role in improving model performance.

G. Visualization

To illustrate the need for multi-scale modeling, four types of anomaly samples were selected from the D1 dataset. In addition, two abnormal samples were randomly selected from the U1 dataset to more fully show that the temporal scale variations exist in the UAV flight datasets. Fig. 9 and Fig. 11 visualize the original signal, the decomposed signal, the normalized anomaly score, and routing weight matrix. In these figures, the light red region denotes the window in which the

TABLE VII
ABLATION STUDY RESULTS.

Dataset	Metrics	W/O Multi*	W/O Router	W/O CL-DAM	MTCL
D1	*ACC*	0.9337	0.9283	0.9273	**0.9384**
	Precision	0.9239	0.9208	**0.9306**	0.9244
	Recall	0.8234	0.8049	0.7909	**0.8417**
	F1-score	0.8708	0.8589	0.8550	**0.8811**
D2	*ACC*	0.9416	0.9434	0.9423	**0.9470**
	Precision	0.7321	0.7444	0.7372	**0.7481**
	Recall	0.8950	0.8984	0.8898	**0.9152**
	F1-score	0.8054	0.8142	0.8064	**0.8233**
D3	*ACC*	0.9131	0.9158	0.9243	**0.9299**
	Precision	0.7729	0.7812	0.8016	**0.8047**
	Recall	0.6086	0.6220	0.6693	**0.7136**
	F1-score	0.6810	0.6925	0.7295	**0.7564**
D4	*ACC*	0.9112	0.9169	0.9105	**0.9256**
	Precision	0.8242	0.8379	0.8277	**0.8404**
	Recall	0.7329	0.7482	0.7239	**0.7965**
	F1-score	0.7759	0.7905	0.7723	**0.8178**
D5	*ACC*	0.9358	0.9290	0.9244	**0.9372**
	Precision	0.7701	0.7678	0.7466	**0.7737**
	Recall	0.7881	0.7245	0.7161	**0.7946**
	F1-score	0.7790	0.7455	0.7310	**0.7840**

• **Multi*** represents multi-scale mechanism.

anomaly occurs, and the red dashed line indicates the model threshold for the normalized anomaly score. Each row of the matrix of routing weights represents one sparse MoE block, and each column corresponds to a distinct expert network. S1 is the minimum scale weight and S4 is the maximum scale weight in this layer.

Fig. 9 illustrates four anomaly type samples, which are engine failure, elevator failure, rudder failure, and aileron failure. Fig. 11 shows two abnormal samples, both of which are brake faults. It can be seen that different samples present significant scale variations. For example, the elevator fault exhibits a shorter periodicity, while the aileron fault displays a more pronounced downward trend. In contrast, the engine fault has stronger periodic fluctuations and the rudder fault follows a gentler trend. In the first sample of FW-UAV datasets, MTCL-UAV employs longer-scale ENs for feature extraction; in the second sample, it uses shorter-scale ENs. These observations highlight the presence of time series scale variations in UAV flight data. MTCL-UAV adapts to multiple scales to achieve more effective feature extraction. Specifically, the Router isolates the seasonal and trend components of the original signal and then adds them back to emphasize periodicity and trend characteristics. It then identifies the optimal scale for each sample based on these decomposed signals, assigning the original signal to the most suitable EN. Within each EN, samples are divided into patches of a specific scale, and CL-DAM is utilized to extract temporal correlations. Finally, outputs from all ENs are fused according to routing weights to produce the final output. The normalized anomaly scores show that the proposed model effectively identifies abnormal segments in the samples after this multi-scale feature extraction.

V. DISCUSSION

Regarding the UAV anomaly detection, this study has two major limitations to consider:

On the hand, our method employs channel-independent modeling, which overlooks interactions between channels and may fail to capture spatial correlations among various features in flight data. In scenarios where multiple channels carry interrelated information, this limitation can lead to suboptimal outcomes. However, channel-independent approaches are effective in mitigating model overfitting, especially for small datasets, which is a common scenario in UAV anomaly detection. Although incorporating both temporal and spatial correlations could improve model performance, it is important to consider the computational costs. Models sensitive to both dimensions, particularly transformer-based architectures, may incur significant computational cost, limiting the application in the real world. On the other hand, our approach introduces additional computational overhead due to multi-scale feature extraction and fusion, which poses challenges for deployment on resource-constrained UAVs. This complexity can hinder real-time applications where quick decision-making is critical. Fortunately, the MoE architecture mitigates some of the additional time costs, as each expert network can be computed in parallel. However, the model does have limitations in terms of space efficiency.

In conclusion, UAV anomaly detection models should balance performance and computational efficiency. Future work could focus on optimizing the architecture to achieve this goal. Alternatively, techniques such as model distillation and model pruning can be considered to reduce model complexity without sacrificing performance.

VI. Conclusion

In this paper, multi-scale Transformers with contrastive learning for UAV anomaly detection (MTCL-UAV) is proposed. It is built based on the sparse mixture-of-experts (MoE) architecture. Each MoE block includes a Router, Expert Networks (ENs), and an aggregater. MTCL-UAV effectively solves the challenges of temporal variations and temporal correlations in UAV flight data. It employs seasonal and trend decomposition to adaptively identify optimal scales and aggregate outputs from ENs. To enhance representational capabilities, a dual attention mechanism enhanced by contrastive learning (CL-DAM) is proposed to extract temporal correlations in a comprehensive way. Experimental results demonstrate that MTCL-UAV achieves superior performance compared to existing methods and exhibits robustness at varying noise levels. The limitations of the proposed model and the directions of future research are also discussed.

In the future, multi-scale information fusion in temporal and spatial dimensions will be explored. The development of more efficient frameworks to satisfy the requirements for lightweight deployment in UAVs will also be considered.

References

[1] S. Li, B. Duo, M. D. Renzo, M. Tao, and X. Yuan, "Robust secure UAV communications with the aid of reconfigurable intelligent surfaces," *IEEE Trans. Wirel. Commun.*, vol. 20, no. 10, pp. 6402–6417, 2021. [Online]. Available: https://doi.org/10.1109/TWC.2021.3073746

[2] Y. Wan, Y. Zhong, A. Ma, and L. Zhang, "An accurate UAV 3-d path planning method for disaster emergency response based on an improved multiobjective swarm intelligence algorithm," *IEEE Trans. Cybern.*, vol. 53, no. 4, pp. 2658–2671, 2023. [Online]. Available: https://doi.org/10.1109/TCYB.2022.3170580

[3] R. W. L. Coutinho and A. Boukerche, "UAV-mounted cloudlet systems for emergency response in industrial areas," *IEEE Trans. Ind. Informatics*, vol. 18, no. 11, pp. 8007–8016, 2022. [Online]. Available: https://doi.org/10.1109/TII.2022.3174113

[4] A. Menshchikov, D. Shadrin, V. Prutyanov, D. Lopatkin, S. Sosnin, E. V. Tsykunov, E. Iakovlev, and A. Somov, "Real-time detection of hogweed: UAV platform empowered by deep learning," *IEEE Trans. Computers*, vol. 70, no. 8, pp. 1175–1188, 2021. [Online]. Available: https://doi.org/10.1109/TC.2021.3059819

[5] K. Liu and J. Zheng, "UAV trajectory optimization for time-constrained data collection in uav-enabled environmental monitoring systems," *IEEE Internet Things J.*, vol. 9, no. 23, pp. 24 300–24 314, 2022. [Online]. Available: https://doi.org/10.1109/JIOT.2022.3189214

[6] M. Pan, C. Chen, X. Yin, and Z. Huang, "UAV-aided emergency environmental monitoring in infrastructure-less areas: Lora mesh networking approach," *IEEE Internet Things J.*, vol. 9, no. 4, pp. 2918–2932, 2022. [Online]. Available: https://doi.org/10.1109/JIOT.2021.3095494

[7] P. Xiong, Z. Li, Y. Li, S. Huang, C. Liu, and F. Gu, "Fault diagnosis of uav based on adaptive siamese network with limited data," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1–11, 2023.

[8] L. Yang, S. Li, C. Li, A. Zhang, and X. Zhang, "A survey of unmanned aerial vehicle flight data anomaly detection: Technologies, applications, and future directions," *Science China Technological Sciences*, vol. 66, no. 4, pp. 901–919, Apr. 2023.

[9] A. Keipour, M. Mousaei, and S. A. Scherer, "Automatic real-time anomaly detection for autonomous aerial vehicles," in *ICRA*. IEEE, 2019, pp. 5679–5685. [Online]. Available: https://doi.org/10.1109/ICRA.2019.8794286

[10] D. Ge, Z. Dong, Y. Cheng, and Y. Wu, "An enhanced spatio-temporal constraints network for anomaly detection in multivariate time series," *Knowledge-Based Systems*, vol. 283, p. 111169, 2024.

[11] Y. Liu and W. Ding, "A knns based anomaly detection method applied for uav flight data stream," in *2015 Prognostics and System Health Management Conference (PHM)*, 2015, pp. 1–8.

[12] A. Alos and Z. Dahrouj, "Detecting contextual faults in unmanned aerial vehicles using dynamic linear regression and k-nearest neighbour classifier," *Gyroscopy and Navigation*, vol. 11, no. 1, pp. 94–104, Jan. 2020.

[13] D. Yong, Z. Yuanpeng, X. Yaqing, P. Yu, and L. Datong, "Unmanned aerial vehicle sensor data anomaly detection using kernel principle component analysis," in *2017 13th IEEE International Conference on Electronic Measurement & Instruments (ICEMI)*, 2017, pp. 241–246.

[14] D. Pan, "Hybrid data-driven anomaly detection method to improve uav operating reliability," in *2017 Prognostics and System Health Management Conference (PHM-Harbin)*, 2017, pp. 1–4.

[15] L. Liu, M. Liu, Q. Guo, D. Liu, and Y. Peng, "Mems sensor data anomaly detection for the uav flight control subsystem," in *2018 IEEE SENSORS*, 2018, pp. 1–4.

[16] A. B. Mohammed, L. Chaari Fourati, and A. M. Fakhrudeen, "Isolation forest algorithm against UAV's GPS spoofing attack," in *CPSCom*, 2024, pp. 459–463.

[17] E. Baskaya, M. Bronz, and D. Delahaye, "Fault detection & diagnosis for small uavs via machine learning," in *2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC)*, 2017, pp. 1–6.

[18] L. Yang, S. Li, Y. Zhang, C. Zhu, and Z. Liao, "Deep learning-assisted unmanned aerial vehicle flight data anomaly detection: A review," *IEEE Sensors Journal*, vol. 24, no. 20, pp. 31 681–31 695, 2024.

[19] T. Yang, J. Chen, H. Deng, and Y. Lu, "UAV abnormal state detection model based on timestamp slice and multi-separable CNN," *Electronics*, vol. 12, no. 6, p. 1299, 2023.

[20] G. Jiang, P. Nan, J. Zhang, Y. Li, and X. Li, "Robust spatial-temporal autoencoder for unsupervised anomaly detection of unmanned aerial vehicle with flight data," *IEEE Trans. Instrum. Meas.*, vol. 73, pp. 1–14, 2024. [Online]. Available: https://doi.org/10.1109/TIM.2024.3428649

[21] B. Wang, Z. Wang, L. Liu, D. Liu, and X. Peng, "Data-driven anomaly detection for UAV sensor data based on deep learning prediction model," in *2019 Prognostics and System Health Management Conference (PHM-Paris)*. IEEE, 2019, pp. 286–290.

[22] R. Dhakal, C. Bosma, P. Chaudhary, and L. N. Kandel, "UAV fault and anomaly detection using autoencoders," in *DASC*. IEEE, 2023, pp. 1–8.

[23] G. Bae and I. Joe, "UAV anomaly detection with distributed artificial intelligence based on LSTM-AE and AE," in *Advanced Multimedia and Ubiquitous Engineering*. Springer, 2020, pp. 305–310.
[24] M. W. Ahmad, M. U. Akram, M. M. Mohsan, K. Saghar, R. Ahmad, and W. H. Butt, "Transformer-based sensor failure prediction and classification framework for UAVs," *Expert Syst. Appl.*, vol. 248, p. 123415, 2024. [Online]. Available: https://doi.org/10.1016/j.eswa.2024.123415
[25] A. Keipour, M. Mousaei, and S. A. Scherer, "ALFA: A dataset for UAV fault and anomaly detection," *Int. J. Robotics Res.*, vol. 40, no. 2-3, 2021. [Online]. Available: https://doi.org/10.1177/0278364920966642
[26] P. Chen, Y. Zhang, Y. Cheng, Y. Shu, Y. Wang, Q. Wen, B. Yang, and C. Guo, "Pathformer: Multi-scale transformers with adaptive pathways for time series forecasting," in *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.
[27] Y. Yang, C. Zhang, T. Zhou, Q. Wen, and L. Sun, "DCdetector: Dual attention contrastive representation learning for time series anomaly detection," in *SIGKDD*. ACM, 2023, pp. 3033–3045.
[28] J. Xu, H. Wu, J. Wang, and M. Long, "Anomaly Transformer: Time series anomaly detection with association discrepancy," in *ICLR*, 2022. [Online]. Available: https://openreview.net/forum?id=LzQQ89U1qm_
[29] Y. Zhou, T. Lei, H. Liu, N. Du, Y. Huang, V. Y. Zhao *et al.*, "Mixture-of-experts with expert choice routing," in *NeurIPS*, 2022. [Online]. Available: http://papers.nips.cc/paper_files/paper/2022/hash/2f00ecd787b432c1d36f3de9800728eb-Abstract-Conference.html
[30] X. Ma, Y. Yu, H. Wu, and K. Zhou, "Efficient reflectance capture with a deep gated mixture-of-experts," *IEEE Trans. Vis. Comput. Graph.*, vol. 30, no. 7, pp. 4246–4256, 2024. [Online]. Available: https://doi.org/10.1109/TVCG.2023.3261872
[31] Y. Kim, H. Lim, and D. Han, "Scaling beyond the GPU memory limit for large mixture-of-experts model training," in *ICML*, 2024. [Online]. Available: https://openreview.net/forum?id=uLpyWQPyF9
[32] D. Shen, C. Qin, C. Wang, Z. Dong, H. Zhu, and H. Xiong, "Topic modeling revisited: A document graph-based neural network perspective," *Advances in neural information processing systems*, vol. 34, pp. 14 681–14 693, 2021.
[33] Q. Liu, Z. Dong, C. Liu, X. Xie, E. Chen, and H. Xiong, "Social marketing meets targeted customers: A typical user selection and coverage perspective," in *ICDM*. IEEE, 2014, pp. 350–359.
[34] Y. Ye, Z. Dong, H. Zhu, T. Xu, X. Song, R. Yu, and H. Xiong, "Mane: Organizational network embedding with multiplex attentive neural networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 4, pp. 4047–4061, 2022.
[35] J. Hang, Z. Dong, H. Zhao, X. Song, P. Wang, and H. Zhu, "Outside in: Market-aware heterogeneous graph neural network for employee turnover prediction," in *ICWSDM*. ACM, 2022, pp. 353–362.
[36] X. He, Q. Chen, L. Tang, W. Wang, and T. Liu, "CGAN-based collaborative intrusion detection for UAV networks: A blockchain-empowered distributed federated learning approach," *IEEE Internet Things J.*, vol. 10, no. 1, pp. 120–132, 2023. [Online]. Available: https://doi.org/10.1109/JIOT.2022.3200121
[37] M. Cheng, Q. Li, J. Lv, W. Liu, and J. Wang, "Multi-scale LSTM model for BGP anomaly classification," *IEEE Trans. Serv. Comput.*, vol. 14, no. 3, pp. 765–778, 2021. [Online]. Available: https://doi.org/10.1109/TSC.2018.2824809
[38] S. Liu, H. Yu, C. Liao, J. Li, W. Lin, A. X. Liu *et al.*, "Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting," in *ICLR*, 2022. [Online]. Available: https://openreview.net/forum?id=0EXmFzUn5I
[39] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A time series is worth 64 words: Long-term forecasting with Transformers," in *ICLR*, 2023. [Online]. Available: https://openreview.net/forum?id=Jbdc0vTOcol
[40] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "TimesNet: Temporal 2d-variation modeling for general time series analysis," in *ICLR*, 2023. [Online]. Available: https://openreview.net/forum?id=ju_Uqw384Oq
[41] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting," in *ICML*. PMLR, 2022, pp. 27 268–27 286. [Online]. Available: https://proceedings.mlr.press/v162/zhou22g.html
[42] Y. Zhang, L. Ma, S. Pal, Y. Zhang, and M. Coates, "Multi-resolution time-series transformer for long-term forecasting," in *AISTATS*, vol. 238. PMLR, 2024, pp. 4222–4230. [Online]. Available: https://proceedings.mlr.press/v238/zhang24l.html
[43] K. H. Park, E. Park, and H. K. Kim, "Unsupervised fault detection on unmanned aerial vehicles: Encoding and thresholding approach," *Sensors*, vol. 21, no. 6, p. 2208, 2021. [Online]. Available: https://doi.org/10.3390/s21062208
[44] M. Bronz, E. Baskaya, D. Delahaye, and S. Puech-Morel, "Real-time fault detection on small fixed-wing uavs using machine learning," in *DASC*, 2020.
[45] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 2017, pp. 5998–6008.
[46] A. Keipour, M. Mousaei, and S. Scherer, "Automatic real-time anomaly detection for autonomous aerial vehicles," in *2019 IEEE International Conference on Robotics and Automation (ICRA)*, May 2019, pp. 5679–5685.
[47] M. D. Tandale and J. Valasek, "Fault-tolerant structured adaptive model inversion control," *Journal of Guidance, Control, and Dynamics*, vol. 29, no. 3, pp. 635–642, 2006.
[48] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1412.6980
[49] A. Chriki, H. Touati, H. Snoussi, and F. Kamoun, "UAV-based surveillance system: an anomaly detection approach," in *ISCC*. IEEE, 2020, pp. 1–6. [Online]. Available: https://doi.org/10.1109/ISCC50000.2020.9219585
[50] A. Chriki, H. Touati, H. Snoussi, and F. Kamoun, "Deep learning and handcrafted features for one-class anomaly detection in UAV video," *Multim. Tools Appl.*, vol. 80, no. 2, pp. 2599–2620, 2021. [Online]. Available: https://doi.org/10.1007/s11042-020-09774-w
[51] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: decomposition transformers with auto-correlation for long-term series forecasting," in *NeurIPS*. Curran Associates Inc., 2024. [Online]. Available: https://proceedings.neurips.cc/paper/2021/hash/bcc0d400288793e8bdcd7c19a8ac0c2b-Abstract.html
[52] Y. Wan, D. Zhang, D. Liu, and F. Xiao, "Cgad: A novel contrastive learning-based framework for anomaly detection in attributed networks," *Neurocomputing*, p. 128379, 2024.

Gang Hu received the B.S. degree in Management Engineering from the Air Force Engineering University, Xi'an, China, in 2020, and the M.S. degree in Electronic Information from the same university in 2022. He is currently pursuing the Ph.D. degree in Control Science and Engineering at the Air Force Engineering University, Xi'an, China. His research interests include unmanned aerial vehicle (UAV) anomaly detection and multivariate time series analysis.

Zhongliang Zhou received his Ph.D. degree from Air Force Engineering University, China in 2007. He is currently a Professor with Air Force Engineering University. His research interests include design of evaluation, causal inference in statistics and observational study.

Zhengxin Li received the Ph.D. degree in Control Science and Engineering from Air Force Engineering University, China in 2011. He was a Postdoctoral Researcher with the School of Artificial Intelligence, Optics and Electronics (iOPEN), Northwestern Polytechnical University. His research interests mainly include time series data mining, machine learning, and artificial intelligence.

Zheng Dong received the master degree in computer science from the University of British Columbia in 2017 and the bachelor of science degree in electronic science and technology from the University of Science and Technology of China in 2015. He is currently a staff software engineer at ByteDance Inc. He has published prolifically in refereed journals and conference proceedings, including TKDE, ACM TOIS, VLDB, NeurIPS, WSDM and ICDM. He was the recipient of the Best-ranked Papers of WSDM 2022. He is a reviewer of many leading academic journals and has served regularly on the program committees of numerous conferences.

Jiayong Fang received his Ph.D. degree from Air Force Engineering University, China in 2011. He is currently an Associate Professor with Air Force Engineering University. His research interests include design of evaluation, causal inference in statistics and observational study.

Yu Zhao (Graduate student member, IEEE) received the Ph.D. degree from Eindhoven University of Technology in 2022. He is currently a Postdoc in Air Force Engineering University, China.

Chuhan Zhou received the B.S. degree in management engineering from Air Force Engineering University, China in 2020. He is currently pursuing the Ph.D. degree in control science and engineering with Air Force Engineering University. His research interests include UAVs formation control, event-triggered control of multi-agent systems, and resilient security control.