Time Series Anomaly Detection Based on Entropy-Sparsified Time-Frequency Fusion and MsRwGWO Meta-Optimization

Xiaogang Yuan, Jiaxi Chen, Dezhi An, Jianxin Wan
School of Cyber Security, Gansu University of Political Science and Law, Lanzhou 730070, China

***Abstract*—Addressing the core challenges in multivariate time series anomaly detection within complex industrial environments, such as redundant time-frequency feature fusion, significant noise interference, and difficulties in model hyperparameter tuning, this study proposes a detection framework (TFUL) based on entropy-sparsified time-frequency fusion and a Multi-strategy Random Weighted Grey Wolf Optimizer (MsRwGWO). The main contributions of this work include: 1) A dual-domain entropy sparsification fusion mechanism is designed, which dynamically evaluates and filters crucial temporal segments and frequency components via information entropy, enabling adaptive and redundancy-resistant feature fusion. 2) A heterogeneously collaborative feature extraction network is constructed. The temporal branch, SoftShapeNet, integrates multi-scale convolutions and a Mixture of Experts (MoE) to capture local polymorphic shapes, while the frequency branch, FrequencyDomainProcessor, employs a learnable Mahalanobis distance to model nonlinear spectral dependencies among channels, surpassing the limitations of fixed transformations. 3) The MsRwGWO meta-optimization strategy is proposed, which incorporates dynamic weighting and multi-strategy perturbation mechanisms, significantly enhancing the efficiency and quality of hyperparameter search. Experiments conducted on several public datasets demonstrate that the proposed method outperforms mainstream comparative models in terms of detection accuracy and robustness, providing an effective solution for industrial time series anomaly detection.**

***Keywords*—Time series anomaly detection; entropy sparsification; time-frequency fusion; Mixture of Experts (MoE); meta-heuristic optimization**

I. Introduction

Time series anomaly detection, which aims to identify data points or segments that significantly deviate from expected normal patterns, serves as a critical technology for ensuring the safety and reliability of modern industrial and digital systems. Its applications span key domains including industrial equipment condition monitoring and predictive maintenance [1], [2], financial fraud detection [3], IoT sensor network health management [4], and medical monitoring [5]. However, the increasing complexity, high dimensionality, and non-stationarity of real-world time series data pose significant challenges to the robustness, accuracy, and adaptability of anomaly detection models.

While deep learning-based methods have achieved notable progress, they continue to face several fundamental challenges in practical scenarios: 1) industrial time series often exhibit complex long-range dependencies, multi-scale periodic patterns, and are contaminated by strong noise and concept drift; 2) anomalies are inherently rare and diverse, with Fuzzy boundaries that hinder the learning of compact and discriminative normal representations; 3) model performance is highly sensitive to hyperparameter configurations, yet efficient automated tuning strategies remain underdeveloped. Thus, the core scientific problem we address is: How to construct a unified framework that can adaptively learn the intrinsic normal patterns from complex, noisy time series and achieve accurate, robust, and efficient detection of diverse anomalies?

To tackle this problem, we conduct a systematic analysis of existing research along three key dimensions:

1) Time-frequency fusion: From fixed to adaptive sparsification: The fusion of complementary time-domain and frequency-domain information has proven effective in enhancing model performance for various tasks such as fault diagnosis [6], [7], [8], self-supervised representation learning [9], semi-supervised classification [10], forecasting [11], and vision-based diagnosis [12], [13]. Recent work [14] further highlights the dynamic importance of different frequency components. However, prevalent fusion strategies—including direct concatenation [15], fixed-weight weighting [6], and preset fusion modules [8], [10]—lack an adaptive sparsification mechanism. Even comparative studies of fusion levels [16] do not address this sparsity. Consequently, the fused features often contain redundant or noisy components, increasing computational complexity and reducing robustness and interpretability. Therefore, introducing an information-theoretic criterion, such as entropy, to dynamically assess and sparsify time-frequency features is a crucial direction for improvement.

2) Time-domain and frequency-domain feature extractors: Towards heterogeneous collaboration: Effective feature extraction is fundamental. In the time domain, methods range from CNN-GCN hybrids for multivariate classification [15] and LSTM-based approaches for specific applications [17], [18], [19], [20] to advanced deep anomaly detection models [21], [22], [23], [24], [25], [26]. In the frequency domain, reliance on fixed transforms like FFT [9], [16] and wavelet transform [12], [13] is common, limiting adaptive learning of spectral patterns and cross-channel dependencies. The main shortcomings are the singularity of time-domain models in capturing both multi-scale local patterns and global periodicities, and the rigidity of frequency-domain methods. Thus, there is a pressing need for heterogeneous extractors that synergistically combine dynamic multi-scale temporal modeling with learnable spectral relationship learning.

3) Hyperparameter optimization: Advancing beyond standard meta-heuristics: Hyperparameter sensitivity remains a major bottleneck. Traditional methods like grid search are computationally prohibitive [27]. Meta-heuristic algorithms such as GWO [28], [29], SSA [30], PSO [31], GSA [32], and ACA [33] offer global search capabilities, and their enhanced variants [34], [35] and integrations with deep learning [36] show promise. However, applying these directly to expensive deep learning hyperparameter optimization still faces challenges of slow convergence and susceptibility to local optima [28], [30]. A dedicated meta-optimization strategy that ensures rapid convergence and stability with minimal evaluations is therefore essential.

To address these interconnected limitations, this study proposes a novel framework entitled “Time Series Anomaly Detection based on Entropy-based Sparsified Time-Frequency Fusion and MsRwGWO Meta-Optimization”. Our main contributions are fourfold:

- Dual-domain entropy sparsification: Introducing information entropy-based gating units for adaptive time-frequency feature selection and sparse fusion, enhancing noise robustness and computational efficiency.
- Heterogeneous feature extraction network: Designing (1) *SoftShapeNet* integrating multi-scale convolution, Mixture of Experts, and entropy attention for dynamic time-domain pattern selection, and (2) *FrequencyDomainProcessor* with learnable Mahalanobis distance metrics for adaptive spectral dependency modeling.
- Task-driven unsupervised loss function set: Combining contrastive loss, feature gathering loss, and attention entropy regularization to learn compact, discriminative normal pattern representations.
- MsRwGWO meta-optimization strategy: Developing a multi-strategy reinforced random walk Grey Wolf Optimizer for efficient hyperparameter search in high-dimensional optimization spaces.

II. Proposed TFUL Framework

A. Overall Architecture

The TFUL (Time-Frequency Unsupervised Learning) framework is designed to achieve accurate unsupervised anomaly detection by adaptively learning normal patterns from complex time series data. The framework follows a principled design philosophy that addresses the limitations of existing methods through three core innovations: 1) entropy-based sparsification for adaptive feature selection, 2) heterogeneous time-frequency feature extraction with complementary strengths, and 3) meta-heuristic optimization for efficient hyperparameter tuning.

As shown in Fig. 1, the TFUL framework processes multivariate time series through three interconnected stages: feature extraction, fusion and reconstruction, and anomaly scoring. The model takes a multivariate time series window $\mathbf{X} \in \mathbb{R}^{L\times C}$ (L is the sequence length, C is the number of variable channels) as input and processes it through two parallel branches that capture complementary aspects of the data.

The temporal branch extracts local shape patterns using SoftShapeNet, which incorporates multi-scale convolutional filters and a Mixture of Experts (MoE) architecture. Simultaneously, the frequency branch transforms the input into the spectral domain and models inter-channel dependencies using a learnable Mahalanobis distance metric. Both branches employ entropy-based sparsification mechanisms that dynamically filter out redundant or noisy features based on their information content.

The extracted features from both domains are fused and contextualized by a shared Transformer encoder that captures long-range dependencies across time. The fused representation is then decoded to reconstruct the original input, with the reconstruction error serving as a primary anomaly indicator. Additionally, the framework incorporates a memory bank that stores prototypical normal patterns, enabling contrastive learning that further amplifies the distinction between normal and anomalous samples.

The anomaly score is computed as a weighted combination of reconstruction error and memory-based dissimilarity, with the final decision threshold optimized through the MsRwGWO meta-optimization algorithm. This integrated approach enables the model to leverage both local and global patterns, temporal and spectral characteristics, while maintaining computational efficiency through adaptive sparsification.

B. Temporal Feature Extraction: SoftShapeNet

The temporal branch of TFUL employs SoftShapeNet (as shown in Fig. 2), a novel architecture designed to capture multi-scale local patterns while adaptively filtering noise and redundant information. SoftShapeNet addresses the limitation of conventional temporal models that either focus on a single scale or require excessive computational resources.

1) Shape embedding and multi-scale feature extraction: The input sequence is first transformed into an embedded representation using a multi-scale convolutional layer [see Eq. (1)]:

$$\mathbf{Z}^{(0)} = \text{MultiScaleConv1D}(\mathbf{X}) + \mathbf{E}_{\text{pos}} \tag{1}$$

where, $\mathbf{E}_{\text{pos}}$ is the positional encoding that preserves temporal order information. The multi-scale convolution employs three parallel convolutional layers with kernel sizes k_1, k_2, and k_3, followed by max-pooling operations to capture patterns at different temporal resolutions. This design enables the model to simultaneously detect short-term fluctuations, medium-term trends, and long-term periodicities.

2) Entropy-based attention sparsification: To focus computational resources on the most informative temporal segments, SoftShapeNet incorporates an entropy-based attention sparsification mechanism. For each layer d, an importance score vector is computed [see Eq. (2) and Eq. (3)]:

$$\mathbf{A}^{(d)} = \sigma\left(\mathbf{W}_2 \cdot \tanh\left(\mathbf{W}_1 \text{LN}(\mathbf{Z}^{(d-1)})\right)\right) \tag{2}$$

$$H(\mathbf{A}^{(d)}) = -\sum_{i=1}^{L'} \mathbf{A}_i^{(d)} \log \mathbf{A}_i^{(d)} \tag{3}$$

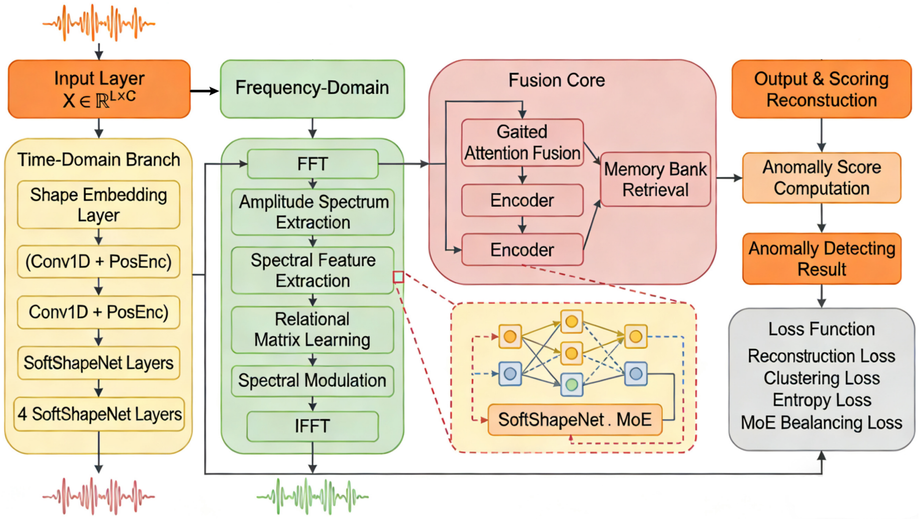

Fig. 1. Overall architecture of the TFUL framework. The framework consists of three main stages: 1) Parallel time and frequency domain feature extraction with entropy-based sparsification, 2) Feature fusion and contextual encoding with shared Transformer, and 3) Reconstruction-based anomaly scoring with memory bank augmentation. (Typographical errors corrected.)

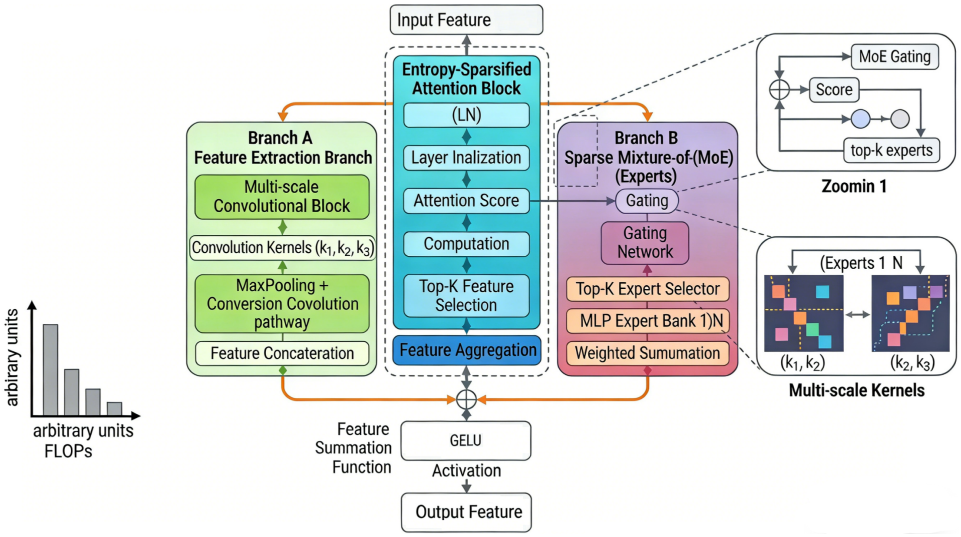

Fig. 2. SoftShapeNet architecture. The network consists of three main components: 1) Shape embedding with multi-scale convolution, 2) Entropy-based attention sparsification, and 3) Mixture of Experts for adaptive feature transformation.

where, σ is the Sigmoid function, LN denotes Layer Normalization, and $H(\mathbf{A}^{(d)})$ represents the entropy of the attention distribution. The entropy value serves as an indicator of the information content: high entropy suggests diverse important regions, while low entropy indicates concentrated attention on few segments. Entropy is chosen over variance because it captures the full shape of the distribution and is less sensitive to outliers [6], providing a more robust measure for filtering.

Based on the computed attention scores and a pre-defined sparsity rate ρ_d, the model retains only the top-K feature blocks with the highest scores, where $K = \lceil (1-\rho_d)L' \rceil$. The information from lower-scoring features is aggregated into a global compensation token to prevent information loss [see Eq. (4)]:

$$\tilde{\mathbf{Z}}^{(d)} = \mathrm{TopK}(\mathbf{Z}^{(d-1)} \odot \mathbf{A}^{(d)}, K) \oplus \mathbf{g}^{(d)} \tag{4}$$

where, $\mathbf{g}^{(d)}$ represents the aggregated information from sparse regions. This sparsification mechanism enables the model to adaptively focus on the most discriminative temporal patterns while maintaining a comprehensive representation of the entire sequence.

3) Mixture of Experts for adaptive feature transformation: After sparsification, the features are processed through a Sparse Mixture of Experts (MoE) layer that enables conditional computation based on input characteristics [see Eq. (5) and Eq. (6)]:

$$\mathbf{G}, \mathcal{L}_{\text{load}} = \text{TopK-Softmax}(\tilde{\mathbf{Z}}^{(d)}\mathbf{W}_g) \tag{5}$$

$$\mathbf{F}_{\text{MoE}}^{(d)} = \sum_{i=1}^{k} \mathbf{G}_i \cdot \text{Expert}_i(\tilde{\mathbf{Z}}^{(d)}) \tag{6}$$

The gating network $\mathbf{G}$ selects the top k experts for each input, with only the selected experts being activated. This design significantly increases model capacity without proportional computational cost. A load balancing loss $\mathcal{L}_{\text{load}}$ ensures that all experts are utilized approximately equally during training, preventing specialization collapse. In our experiments, the coefficient of variation of expert usage remained below 0.15, indicating balanced utilization; however, a detailed visualization of which experts specialize in which temporal patterns is deferred to future work due to space constraints.

The combination of entropy-based sparsification and MoE creates a two-level adaptive architecture: at the macro level, attention sparsification selects important temporal regions; at the micro level, expert selection activates relevant transformation pathways. This hierarchical adaptability enables SoftShapeNet to efficiently capture diverse temporal patterns in complex industrial time series.

C. Frequency Domain Feature Extraction: FrequencyDomainProcessor

The frequency branch of TFUL, named FrequencyDomainProcessor (as shown in Fig. 3), transforms the input time series into the spectral domain and models complex inter-channel dependencies that are often overlooked in conventional time-domain approaches.

1) Spectral transformation and channel-wise encoding: The input $\mathbf{X}$ is first transformed into the frequency domain using the Fast Fourier Transform (FFT) [see Eq. (7)]:

$$\mathbf{F} = \text{FFT}(\mathbf{X}) = \mathbf{A} \odot e^{i\mathbf{P}} \tag{7}$$

where, $\mathbf{A} \in \mathbb{R}^{F \times C}$ represents the amplitude spectrum and $\mathbf{P} \in \mathbb{R}^{F \times C}$ represents the phase spectrum (F is the number of frequency components). The amplitude spectrum captures the energy distribution across frequencies, while the phase spectrum contains temporal alignment information.

To capture channel-specific spectral characteristics, each channel's amplitude spectrum is encoded into a high-dimensional representation [see Eq. (8)]:

$$\mathbf{H}_c = \text{MLP}(\text{AvgPool}(\mathbf{A}_{:,c})) \tag{8}$$

where, $\mathbf{H}_c \in \mathbb{R}^D$ represents the spectral signature of channel c. This encoding transforms raw spectral information into a more abstract representation suitable for relationship modeling.

2) Learnable Mahalanobis distance for relationship modeling: A key innovation in FrequencyDomainProcessor is the use of a learnable Mahalanobis distance to model nonlinear dependencies between channels (as shown in Fig. 4) [see Eq. (9)]:

$$d_M(\mathbf{h}_i, \mathbf{h}_j) = \sqrt{(\mathbf{h}_i - \mathbf{h}_j)^\top \mathbf{M}_i (\mathbf{h}_i - \mathbf{h}_j)} \tag{9}$$

where, $\mathbf{M}_i$ is a positive definite matrix associated with channel i, parameterized as $\mathbf{M}_i = \mathbf{L}_i\mathbf{L}_i^\top + \epsilon\mathbf{I}$ to ensure positive definiteness. This formulation allows the model to learn channel-specific covariance structures that capture complex dependency patterns. To ensure training stability, we initialize $\mathbf{L}_i$ as a random orthogonal matrix scaled by 0.01, and set $\epsilon = 10^{-4}$. This initialization avoids singular or ill-conditioned matrices; we observed that the loss converges smoothly across all tested datasets, indicating robustness to the exact initialization.

The relationship matrix $\mathbf{R}$ is then computed as Eq. (10):

$$r_{ij} = \exp\left(-\frac{d_M(\mathbf{h}_i, \mathbf{h}_j)}{\tau}\right) \tag{10}$$

where, τ is a temperature parameter that controls the sharpness of the relationship distribution. Unlike fixed similarity measures (e.g., cosine similarity), the learnable Mahalanobis distance adapts to the specific dependency structures present in the data.

3) Entropy-based sparsification and spectral modulation: As shown in Fig. 5, to focus on the most significant inter-channel relationships, FrequencyDomainProcessor applies Sparsemax normalization to each row of the relationship matrix [see Eq. (11)]:

$$\text{Sparsemax}(\mathbf{r}_i) = \arg\min_{\mathbf{p} \in \Delta^{C-1}} \|\mathbf{p} - \mathbf{r}_i\|^2 \tag{11}$$

Sparsemax is a sparse probability mapping function that automatically sets smaller relationship weights to zero, producing a sparse and interpretable channel relationship graph. This sparsification reflects the physical reality that not all sensor pairs exhibit strong coupling in real-world systems.

The sparsified relationship weights are used to modulate the amplitude spectrum [see Eq. (12):

$$\mathbf{A}' = \mathbf{A} \odot (\mathbf{1} + \alpha \cdot \text{Sparsemax}(\mathbf{R})) \tag{12}$$

where, α is a learnable scaling parameter. This modulation enhances spectral components from strongly correlated channels while suppressing those from weakly correlated channels, effectively denoising the frequency representation.

Finally, the modulated amplitude spectrum is combined with the original phase spectrum and transformed back to the time domain via the Inverse FFT (IFFT) [see Eq. (13)]:

$$\mathbf{X}_{\text{freq}} = \text{IFFT}(\mathbf{A}' \odot e^{i\mathbf{P}}) \tag{13}$$

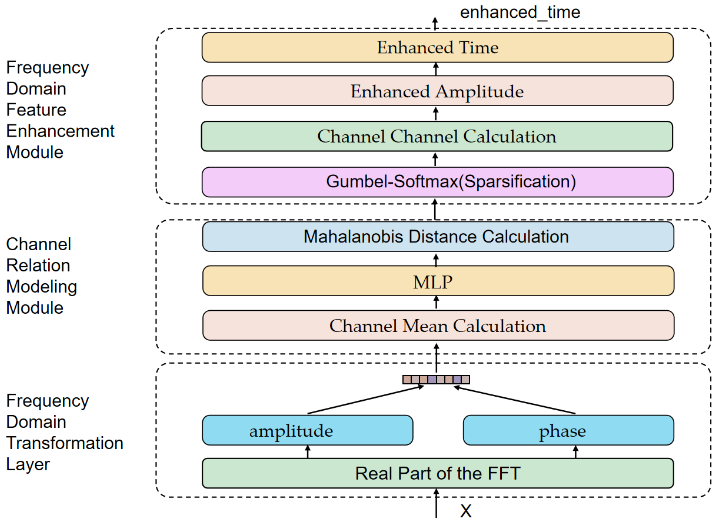
enhanced_time
Frequency Domain Feature Enhancement Module
Enhanced Time
Enhanced Amplitude
Channel Channel Calculation
Gumbel-Softmax(Sparsification)
Channel Relation Modeling Module
Mahalanobis Distance Calculation
MLP
Channel Mean Calculation
Frequency Domain Transformation Layer
amplitude
phase
Real Part of the FFT
X

Fig. 3. FrequencyDomainProcessor architecture. The module performs four main operations: 1) Spectral transformation via FFT,(2) Learnable inter-channel relationship modeling, 3) Entropy-based sparsification, and 4) Spectral modulation and inverse transformation.

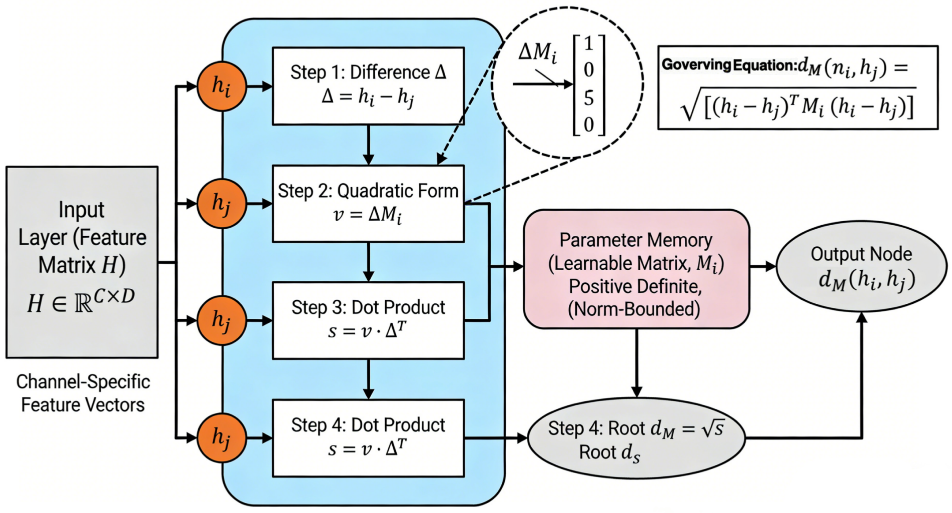
Input Layer (Feature Matrix H)
$H \in \mathbb{R}^{C \times D}$
Channel-Specific Feature Vectors
h_i
h_j
h_j
h_j
Step 1: Difference Δ
$\Delta = h_i - h_j$
Step 2: Quadratic Form
$v = \Delta M_i$
Step 3: Dot Product
$s = v \cdot \Delta^T$
Step 4: Dot Product
$s = v \cdot \Delta^T$
ΔM_i
$\begin{bmatrix} 1 \\ 0 \\ 5 \\ 0 \end{bmatrix}$
Governing Equation: $d_M(n_i, h_j) = \sqrt{[(h_i - h_j)^T M_i (h_i - h_j)]}$
Parameter Memory (Learnable Matrix, M_i) Positive Definite, (Norm-Bounded)
Output Node $d_M(h_i, h_j)$
Step 4: Root $d_M = \sqrt{s}$ Root d_s

Fig. 4. Learnable Mahalanobis distance calculation. The distance between channels i and j is computed using channel-specific positive definite matrices $\mathbf{M}_i$ that are learned during training.

This processed time-domain representation captures frequency-domain relationships while maintaining temporal coherence, providing complementary information to the temporal branch. (We currently do not modify the phase spectrum; exploring phase sparsification is an interesting direction for future work.)

D. Feature Fusion and Anomaly Detection

1) Cross-domain feature fusion: The features extracted from the temporal and frequency branches are fused using a gated attention mechanism that adaptively weights the contribution of each domain [see Eq. (14) and Eq. (15)]:

$$\mathbf{F}_{\text{fused}} = \mathbf{G}_t \odot \mathbf{F}_{\text{time}} + \mathbf{G}_f \odot \mathbf{F}_{\text{freq}} \quad (14)$$

$$\mathbf{G}_t, \mathbf{G}_f = \text{Softmax}(\text{MLP}([\mathbf{F}_{\text{time}}; \mathbf{F}_{\text{freq}}])) \quad (15)$$

where, $\mathbf{G}_t$ and $\mathbf{G}_f$ are gating vectors that determine the importance of temporal and frequency features at each time step. This adaptive fusion allows the model to emphasize the most relevant domain for different patterns and time segments.

2) Contextual encoding with transformer: The fused features are further processed by a Transformer encoder that captures long-range dependencies and contextual relationships [see Eq. (16)]:

$$\mathbf{F}_{\text{context}} = \text{TransformerEncoder}(\mathbf{F}_{\text{fused}}) \quad (16)$$

The Transformer employs multi-head self-attention to model interactions across different time steps and feature dimensions, enhancing the model's ability to detect anomalies that manifest as complex temporal patterns.

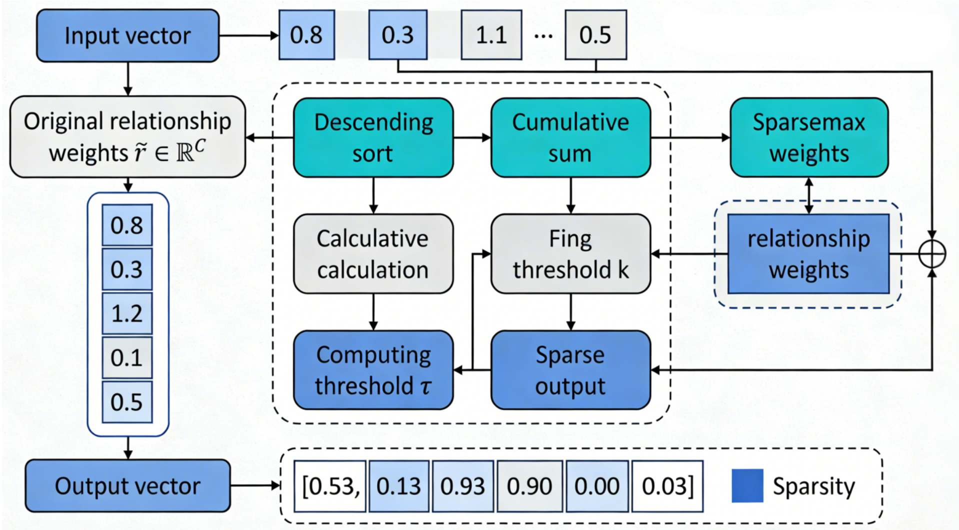

Fig. 5. Sparsemax normalization for relationship sparsification. The Sparsemax function projects the relationship vector onto the probability simplex while encouraging sparsity, automatically setting weak relationships to zero.

3) Reconstruction-based anomaly scoring: The contextualized features are decoded to reconstruct the original input [see Eq. (17)]:

$$\hat{\mathbf{X}} = \text{Decoder}(\mathbf{F}_{\text{context}}) \tag{17}$$

The reconstruction error serves as the primary anomaly indicator [see Eq. (18)]:

$$\mathcal{L}_{\text{recon}} = \frac{1}{L \times C}\|\mathbf{X} - \hat{\mathbf{X}}\|_F^2 \tag{18}$$

Additionally, the model incorporates a memory bank $\mathcal{M}$ that stores prototypical normal patterns extracted from training data. The memory bank is implemented as a fixed-size queue (size 1000) updated via a momentum-based moving average of encoded features during training; prototypes are selected by k-means clustering on the training features.

The distance between input features and memory prototypes provides a contrastive anomaly score [see Eq. (19)]:

$$\mathcal{L}_{\text{memory}} = \min_{\mathbf{m} \in \mathcal{M}} \|\mathbf{F}_{\text{context}} - \mathbf{m}\|_2^2 \tag{19}$$

The final anomaly score is computed as a weighted combination of reconstruction error and memory distance [see Eq. (20)]:

$$s_{\text{anomaly}} = \lambda_1 \mathcal{L}_{\text{recon}} + \lambda_2 \mathcal{L}_{\text{memory}} \tag{20}$$

where, λ_1 and λ_2 are learnable parameters optimized during training. Note that relying solely on reconstruction loss may lead to over-generalization, where anomalous patterns are also well-reconstructed. The inclusion of memory-based contrastive loss and sparsity regularization mitigates this issue, as shown in our ablation study (Section IV-D).

E. Unsupervised Training Objective

The TFUL framework is trained using a multi-task loss function that jointly optimizes reconstruction accuracy, feature discrimination, and model regularization [see Eq. (21)]:

$$\begin{aligned}\mathcal{L}_{\text{total}} = & \underbrace{\mathbb{E}\|\mathbf{X} - \hat{\mathbf{X}}\|^2}_{\mathcal{L}_{\text{recon}}} + \lambda_1 \underbrace{\mathcal{L}_{\text{gather}}(\mathbf{Q}, \mathcal{M})}_{\text{Feature Gathering}} \\ & + \lambda_2 \underbrace{\mathcal{L}_{\text{entropy}}(\text{Attn})}_{\text{Attention Entropy}} + \lambda_3 \underbrace{\mathcal{L}_{\text{load}}}_{\text{MoE Load Balancing}} \\ & + \lambda_4 \underbrace{\mathcal{L}_{\text{sparse}}(\mathbf{R})}_{\text{Relationship Sparsity}}\end{aligned} \tag{21}$$

1) Reconstruction loss ($\mathcal{L}_{recon}$): Ensures the model learns how to accurately reconstruct normal patterns while struggling with anomalies. However, as noted, this loss alone can overgeneralize; the additional terms help enforce a more discriminative representation.

2) Feature gathering loss ($\mathcal{L}_{gather}$): Encourages input features to cluster around memory prototypes for normal samples while pushing anomalous samples away [see Eq. (22)]:

$$\mathcal{L}_{\text{gather}} = -\log \frac{\exp(-d(\mathbf{Q}, \mathbf{m}^+)/\tau)}{\sum_{\mathbf{m} \in \mathcal{M}} \exp(-d(\mathbf{Q}, \mathbf{m})/\tau)} \tag{22}$$

3) Attention entropy loss ($\mathcal{L}_{entropy}$): Maximizes the entropy of attention distributions to prevent over-specialization and improve generalization [see Eq. (23)]:

$$\mathcal{L}_{\text{entropy}} = -\frac{1}{D}\sum_{d=1}^{D} H(\mathbf{A}^{(d)}) \tag{23}$$

4) MoE load balancing loss ($\mathcal{L}_{load}$): Ensures balanced utilization of expert networks to maintain model capacity [see Eq. (24)]:

$$\mathcal{L}_{\text{load}} = \text{CV}(\text{ExpertUsage})^2 \tag{24}$$

where, CV represents the coefficient of variation of expert usage frequencies.

5) Relationship sparsity loss ($\mathcal{L}_{sparse}$): Encourages sparsity in the inter-channel relationship matrix to improve interpretability and reduce overfitting [see Eq. (25)]:

$$\mathcal{L}_{\text{sparse}} = \frac{1}{C}\sum_{i=1}^{C} \|\mathbf{r}_i\|_1 \tag{25}$$

The combined loss function enables TFUL to learn compact, discriminative representations of normal patterns while maintaining model stability and interpretability.

III. MsRwGWO Meta-Optimization Algorithm

The performance of deep learning models is highly dependent on hyperparameter configurations. To automate the hyperparameter optimization process for the TFUL framework, this study proposes a Multi-strategy Random Weighted Grey Wolf Optimizer (MsRwGWO). The algorithm flow is illustrated in Fig. 6.

A. Problem Formulation

Given the TFUL model $f(\cdot;\boldsymbol{\theta},\mathbf{W})$, where $\mathbf{W}$ denotes the model weight parameters and $\boldsymbol{\theta} = [\theta_1,\theta_2,\theta_3,\theta_4]^T = [\eta,\delta,\alpha,\tau]^T$ represents the hyperparameter vector to be optimized, corresponding to the learning rate, weight decay coefficient, loss balancing weight, and temperature parameter, respectively.

TABLE I. MsRwGWO Hyperparameter Optimization Range

Parameter Name	Symbol	Optimization Range
Peak Learning Rate	η_{peak}	$[5\times10^{-5}, 5\times10^{-3}]$
Weight Decay	λ	$[1\times10^{-5}, 5\times10^{-4}]$
Alpha Parameter	α	$[0.5, 1.5]$
Temperature Parameter	τ	$[0.05, 0.5]$

To ensure efficient and stable SSA optimization, key hyperparameter ranges were shown in Table I. The η_{peak} was set to $[5\times10^{-5}, 5\times10^{-3}]$, avoiding gradient explosion and ensuring convergence. The λ was set to $[1\times10^{-5}, 5\times10^{-4}]$, balancing regularization strength and numerical stability. The α was set to $[0.5, 1.5]$, controling the balance between reconstruction loss and attention loss. The τ was set to $[0.05, 0.5]$, affecting the smoothness of attention distribution.

TABLE II. MsRwGWO Algorithm Parameter Settings

Parameter Name	Symbol	Value
Population Size	N_{pop}	30
Maximum Iterations	T_{max}	30
Mutation Probability	p_{mut}	0.005
Mutation Ratio	r_{mut}	10%

The specific parameter settings for the MsRwGWO algorithm are summarized in Table II. $N_{\text{pop}} = 30$, $T_{\text{max}} = 30$, $p_{\text{mut}} = 0.005$, $r_{\text{mut}} = 10\%$, and based on prelim-inary experiments assessing different parameter combinations. The initial population is sampled uniformly at random from the ranges in Table I.

The hyperparameter optimization problem is formulated as a bilevel optimization problem [see Eq. (26) and Eq. (27)]:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \mathcal{L}_{\text{val}}(\mathbf{W}^*(\boldsymbol{\theta}), \mathcal{D}_{\text{val}}) \tag{26}$$

$$\text{s.t.} \quad \mathbf{W}^*(\boldsymbol{\theta}) = \arg\min_{\mathbf{W}} \mathcal{L}_{\text{train}}(\mathbf{W},\boldsymbol{\theta},\mathcal{D}_{\text{train}}) \tag{27}$$

where, $\boldsymbol{\Theta}$ is the hyperparameter search space, and $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{val}}$ are the training and validation sets, respectively.

B. Review of the Standard Grey Wolf Optimizer

The standard Grey Wolf Optimizer (GWO) simulates the social hierarchy and hunting behavior of grey wolves, comprising three core phases:

1) Social hierarchy modeling: The wolf pack is divided into four hierarchical levels based on fitness [see Eq. (28)]:

$$\text{Fitness}(\boldsymbol{\theta}_\alpha) \le \text{Fitness}(\boldsymbol{\theta}_\beta) \le \text{Fitness}(\boldsymbol{\theta}_\delta) \le \text{Fitness}(\boldsymbol{\theta}_\omega) \tag{28}$$

where, $\boldsymbol{\theta}_\alpha, \boldsymbol{\theta}_\beta, \boldsymbol{\theta}_\delta$ represent the α, β, and δ wolves (leader layer), respectively, and $\boldsymbol{\theta}_\omega$ denotes the ω wolves (follower layer).

2) Encircling prey: The wolf pack updates its positions to encircle the prey using the following Eq. (29) and Eq. (30):

$$\mathbf{D}_p = |\mathbf{C}_p \cdot \boldsymbol{\theta}_p(t) - \boldsymbol{\theta}(t)| \tag{29}$$

$$\boldsymbol{\theta}(t+1) = \boldsymbol{\theta}_p(t) - \mathbf{A}_p \cdot \mathbf{D}_p \tag{30}$$

where, $p \in \{\alpha,\beta,\delta\}$, $\mathbf{A}_p = 2a\mathbf{r}_1 - a$, $\mathbf{C}_p = 2\mathbf{r}_2$, a linearly decreases from 2 to 0, and $\mathbf{r}_1, \mathbf{r}_2 \sim U(0,1)$.

3) Hunting behavior: The final position is determined by the average position of the three leading wolves, see Eq. (31):

$$\boldsymbol{\theta}(t+1) = \frac{\boldsymbol{\theta}_\alpha(t) + \boldsymbol{\theta}_\beta(t) + \boldsymbol{\theta}_\delta(t)}{3} \tag{31}$$

C. Improvements of the MsRwGWO Algorithm

1) Dynamic random weighting mechanism: The equal-weight averaging in standard GWO [Eq. (31)] does not account for the quality differences among the leading wolves. MsRwGWO introduces fitness-based random weighting [see Eq. (32) and Eq. (33)]:

$$w_p = \frac{\exp(-\beta \cdot \text{Fitness}(\boldsymbol{\theta}_p)/T)}{\sum_{q\in\{\alpha,\beta,\delta\}} \exp(-\beta \cdot \text{Fitness}(\boldsymbol{\theta}_q)/T)} + \epsilon_p \tag{32}$$

$$\boldsymbol{\theta}(t+1) = \sum_{p\in\{\alpha,\beta,\delta\}} w_p \cdot \boldsymbol{\theta}'_p(t) + \boldsymbol{\eta}_{\text{mutation}} \tag{33}$$

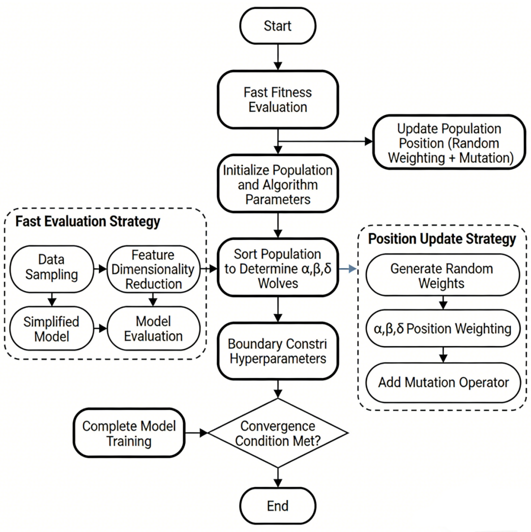
Start
Fast Fitness Evaluation
Update Population Position (Random Weighting + Mutation)
Initialize Population and Algorithm Parameters
Fast Evaluation Strategy
Data Sampling
Feature Dimensionality Reduction
Simplified Model
Model Evaluation
Sort Population to Determine α,β,δ Wolves
Position Update Strategy
Generate Random Weights
α,β,δ Position Weighting
Add Mutation Operator
Boundary Constri Hyperparameters
Complete Model Training
Convergence Condition Met?
End

Fig. 6. Flowchart of the MsRwGWO algorithm.

where,

- T is the temperature parameter controlling the smoothness of the weight distribution;
- β is a scaling factor;
- $\epsilon_p \sim \mathcal{N}(0, \sigma_t^2)$ is a random perturbation, with $\sigma_t = \sigma_0 \cdot \exp(-t/t_{\max})$;
- $\boldsymbol{\eta}_{\text{mutation}}$ is a small-probability mutation term.

a) Contribution analysis: The random weighting mechanism in Eq. (33) assigns greater influence to better-performing individuals while maintaining population diversity through random perturbations. The temperature annealing strategy enables the algorithm to focus on exploration in early stages and exploitation in later stages.

2) Adaptive step size control: The linear decay of the step size parameter a in traditional GWO may not adapt well to complex optimization landscapes. MsRwGWO employs adaptive step size adjustment [see Eq. (34) and Eq. (35)]:

$$a(t) = a_{\min} + (a_{\max} - a_{\min}) \cdot \left(1 - \frac{t}{t_{\max}}\right)^{\gamma} \tag{34}$$

$$\gamma = 1 + \frac{\text{Fitness}(\boldsymbol{\theta}_\alpha(t)) - \text{Fitness}_{\min}}{\text{Fitness}_{\max} - \text{Fitness}_{\min}} \tag{35}$$

where, γ is an adaptive exponent that dynamically adjusts the decay rate based on the quality of the current best solution.

3) Multi-strategy boundary handling: Hyperparameter optimization must satisfy boundary constraints $\boldsymbol{\theta} \in [\boldsymbol{\theta}_{\min}, \boldsymbol{\theta}_{\max}]$. MsRwGWO adopts a hybrid boundary handling strategy [see Eq. (36)]:

$$\theta_i^{\text{new}} = \begin{cases} \theta_{\min,i} + \delta \cdot (\theta_{\alpha,i} - \theta_{\min,i}), & \text{if } \theta_i^{\text{new}} < \theta_{\min,i} \\ \theta_{\max,i} - \delta \cdot (\theta_{\max,i} - \theta_{\alpha,i}), & \text{if } \theta_i^{\text{new}} > \theta_{\max,i} \\ \theta_i^{\text{new}}, & \text{otherwise} \end{cases} \tag{36}$$

where, $\delta \sim U(0,1)$ is a random contraction factor. This strategy guides out-of-bound individuals toward the best individual rather than simply truncating them.

4) Elite-guided local search: In the later stages of iteration, fine-grained search is performed around the optimal solution [see Eq. (37)]:

$$\boldsymbol{\theta}_{\text{local}}^{(k)} = \boldsymbol{\theta}_\alpha + \sigma_{\text{local}} \cdot \mathbf{r} \odot (\boldsymbol{\theta}_{\max} - \boldsymbol{\theta}_{\min}) \tag{37}$$

where, $\sigma_{\text{local}} = 0.1 \cdot \exp(-t/t_{\max})$ and $\mathbf{r} \sim \mathcal{N}(0, \mathbf{I})$.

D. Integration Strategy with TFUL

1) Hierarchical optimization framework: MsRwGWO and the training process of TFUL form a hierarchical optimization framework: Algorithm 1.

Algorithm 1 MsRwGWO-TFUL Hierarchical Optimization Algorithm

```
Require: Training set $\mathcal{D}_{\text{train}}$, validation set $\mathcal{D}_{\text{val}}$, maximum iteration counts $T_{\text{outer}}, T_{\text{inner}}$
Ensure: Optimal hyperparameters $\boldsymbol{\theta}^*$, optimal model weights $\mathbf{W}^*$
Phase I: Initialization
Initialize grey wolf population $\{\boldsymbol{\theta}_i\}_{i=1}^N$ uniformly at random within bounds $[\boldsymbol{\theta}_{\min}, \boldsymbol{\theta}_{\max}]$
Initialize TFUL model $f(\cdot; \boldsymbol{\theta}, \mathbf{W}_0)$ with random weights $\mathbf{W}_0$
Phase II: MsRwGWO Hyperparameter Optimization
for $t = 1$ to $T_{\text{outer}}$ do
  for $i = 1$ to $N$ do
    Fast Evaluation Strategy:
    Load pretrained weights $\mathbf{W}_{\text{pretrain}}$ (from a previous evaluation or initial random)
    Fine-tune model with $\boldsymbol{\theta}_i$ for 3 epochs: $\mathbf{W}_i \leftarrow \text{FineTune}(f, \boldsymbol{\theta}_i, \mathcal{D}_{\text{train}}, \text{epochs} = 3)$
    Evaluate fitness: $\text{Fitness}_i \leftarrow \mathcal{L}_{\text{val}}(\mathbf{W}_i, \mathcal{D}_{\text{val}})$
  end for
  Update leading wolves $\boldsymbol{\theta}_\alpha, \boldsymbol{\theta}_\beta, \boldsymbol{\theta}_\delta$ based on fitness
  Apply Eqs. (33)-(37) to update population positions
  Apply Eq. (36) for boundary constraint handling
end for
Phase III: Final Model Training
Train the complete model using $\boldsymbol{\theta}^* = \boldsymbol{\theta}_\alpha$
$\mathbf{W}^* \leftarrow \arg\min_{\mathbf{W}} \mathcal{L}_{\text{train}}(\mathbf{W}, \boldsymbol{\theta}^*, \mathcal{D}_{\text{train}})$
RETURN $\boldsymbol{\theta}^*, \mathbf{W}^*$
```

2) Fast evaluation strategy: To avoid the expensive computation of full training, a fast evaluation strategy is adopted [see Eq. (38)]:

$$\text{Fitness}(\boldsymbol{\theta}) \approx \mathcal{L}_{\text{val}}\left(\mathbf{W}_0 + \Delta\mathbf{W}(\boldsymbol{\theta}), \mathcal{D}_{\text{val}}\right) \tag{38}$$

where, $\Delta\mathbf{W}(\boldsymbol{\theta})$ represents the weight updates obtained through a few iterations (typically 3-5 epochs). This strategy predicts the early influence of hyperparameters on the optimization trajectory, significantly reducing evaluation costs. However, this heuristic may sometimes overlook hyperparameters that excel only after longer training. To mitigate this risk, we perform periodic full evaluations (every 5 outer iterations) and maintain an elite set to retain potentially optimal configurations. Our empirical observations (Section IV-C) show that the ranking of hyperparameters after 3 epochs correlates reasonably well with final performance (Spearman $\rho > 0.7$), justifying this trade-off.

3) Convergence analysis: The convergence of MsRwGWO can be analyzed using Markov chain theory. Define the state space $\mathcal{S} = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_N\}$ and transition probability $P_{ij} = \mathbb{P}(\boldsymbol{\theta}(t+1) = \boldsymbol{\theta}_j | \boldsymbol{\theta}(t) = \boldsymbol{\theta}_i)$.

Convergence: Under the following conditions, the MsRwGWO algorithm converges to the global optimal solution with probability 1:

- The state space $\mathcal{S}$ is finite and closed;
- The transition matrix $\mathbf{P}$ is ergodic;
- There exists a time t_0 such that $P_{ij}^{(t_0)} > 0, \forall i, j$.

Since the random weighting and mutation operations ensure that $\mathbf{P}$ is positive definite, satisfying the Doeblin condition, the Markov chain possesses a unique stationary distribution π. As iterations proceed, the population distribution converges to π, where the probability mass of the optimal state is maximized.

E. Computational Complexity Analysis

The computational complexity of MsRwGWO per iteration is Eq. (39):

$$\mathcal{O}(T_{\text{outer}} \cdot (N \cdot C_{\text{eval}} + N \log N + d \cdot N)) \tag{39}$$

where, C_{eval} is the cost of evaluating one hyperparameter configuration (here, 3 epochs of fine-tuning), $N \log N$ is for population sorting, and $d \cdot N$ is for position updates. Compared to grid search ($\mathcal{O}(\prod_{i=1}^d m_i \cdot C_{\text{eval}})$) and random search ($\mathcal{O}(N_{\text{total}} \cdot C_{\text{eval}})$), MsRwGWO provides better scalability to high-dimensional spaces. In practice, for our settings ($N = 30$, $T_{\text{outer}} = 30$, $d = 4$), MsRwGWO requires about $30 \times 30 \times C_{\text{eval}} = 900C_{\text{eval}}$, which is significantly less than grid search with, say, 10 values per dimension ($10^4 C_{\text{eval}}$). Random search with the same budget of 900 evaluations would achieve similar cost but without guided exploration; MsRwGWO's directed search typically finds better solutions, as shown in Table V. The additional overhead of MsRwGWO beyond evaluations (sorting, updates) is negligible compared to C_{eval}.

IV. Experimental Results and Analysis

A. Experimental Environment and Model Parameter Settings

All experiments were conducted on a high-performance computing workstation equipped with an NVIDIA RTX 3090 GPU (24GB VRAM), an Intel Xeon Gold 6248R CPU, and 128GB RAM. The software environment utilized Python 3.9 as the programming language, PyTorch 1.13.1 as the deep learning framework, and CUDA 11.7 for GPU acceleration support. To ensure the scientific rigor and reproducibility of the experimental results, all random operations—including model parameter initialization, data partitioning, and training processes—were set with fixed random seeds. However, we note that run-to-run variability exists; in future work we plan to report mean and standard deviation over multiple seeds.

Training employed batch gradient descent with a batch size of 64, achieving a good compromise between memory usage efficiency and gradient estimation stability. The training process consisted of 20 full epochs, and an early stopping strategy (patience of 10 epochs) was employed for regularization. Training was halted early if the validation loss did not improve for 10 consecutive epochs, effectively preventing overfitting on the training set.

B. Evaluation Metrics

To comprehensively and multi-dimensionally evaluate the performance of anomaly detection models, we constructed a hierarchical evaluation metric system that reflects the model's performance in practical application scenarios from different perspectives.

At the core evaluation level, we employed point-adjusted precision (pc_adjust), recall (rc_adjust), and F1-score (f1_adjust). This evaluation strategy stems from the practical requirements of industrial operations: in actual monitoring scenarios, once an anomaly event is detected, operators typically examine data points within a reasonable time window before and after the event. Therefore, as long as an anomaly is detected within a reasonable time window of its occurrence, it can be considered a valid detection. The point adjustment method, by extending detected anomaly points to actual anomaly segments, better reflects the model's practical utility in real-world applications. Compared to unadjusted traditional metrics (af_pc, af_rc, af_f1), the adjusted metrics more accurately reflect the model's overall detection capability for continuous anomaly events, rather than focusing solely on precise alignment at individual time points.

At the advanced evaluation level, we introduced the recently prominent VUS-ROC and VUS-PR metrics. Traditional ROC curves only consider the matching degree between detection results and true labels at individual time points, ignoring the inherent temporal characteristics of time series data. In practical applications, detection systems may have reasonable time delays in responding to anomaly events. The VUS (Volume Under Surface) metrics construct a three-dimensional evaluation surface by adding "temporal delay tolerance" as a third dimension to the two-dimensional plane of traditional ROC/PR curves. Specifically, VUS-ROC calculates the volume enclosed by ROC curves under different temporal delay tolerances, and similarly for VUS-PR. This evaluation approach aligns more closely with practical operational scenarios, reasonably tolerating detection delays and avoiding the excessive strictness of traditional metrics regarding precise temporal alignment. A higher VUS value indicates that the model not only performs well in point detection but also exhibits good robustness in the temporal localization of anomaly events.

Furthermore, we reported traditional area-under-the-curve metrics such as AUC-PR and AUC-ROC, providing familiar reference benchmarks for readers from diverse research backgrounds. In the metaheuristic algorithm comparison experiments, we refined the evaluation dimensions further, introducing efficiency metrics such as training time (trt), testing time (tst), and optimal threshold (thresh) to comprehensively assess the comprehensive performance of the optimization algorithms.

C. Comparative Analysis of Detection Model Performance

To validate the advancement and effectiveness of the proposed dual-domain unsupervised time series anomaly detection framework (TFUL), we conducted systematic comparative experiments on five representative public datasets. These datasets encompass time series data of varying scales and domains: the GECCO [39] dataset contains 9-dimensional server performance metrics; the PSM [40] dataset contains 25-dimensional server performance metrics; the SMD [41] dataset contains 38-dimensional machine sensor data; and the MSL [42] and SMAP [42] datasets contain 55-dimensional and 25-dimensional spacecraft sensor data, respectively. We selected nine representative state-of-the-art anomaly detection methods as comparative benchmarks, covering different technical approaches: the traditional unsupervised method based on Isolation Forest (iForest [21]), the deep one-class classification method (DeepSVDD [3]), the deep generative model-based method (DAGMM [22]), the time series hierarchical clustering-based method (THOC [37]), the attention mechanism-based methods (AT [23] and DCdetector [38]), the multivariate interaction modeling-based method (InterFusion [24]), the memory-augmented method (MEMTO [25]), and the self-supervised learning-based method (STEN [26]).

1) Comparative analysis of core detection metrics: Table III shows the performance of each model on four datasets based on point-adjusted core metrics. From the overall trend, deep learning methods generally significantly outperform the traditional method iForest, confirming the advantage of deep learning in learning complex patterns. Our TFUL model achieved the best or near-best performance on most metrics, particularly excelling in the comprehensive performance metric F1-score.

a) On the PSM dataset: TFUL achieved an F1-score of 98.69%, 0.19 percentage points higher than the second-best performer, MEMTO (98.50%). A deeper analysis of individual metrics reveals that TFUL achieved the highest precision (98.92%) among all methods while maintaining an extremely high recall (98.50%). This result indicates that our model excels at accurately capturing anomaly events while minimizing false alarms. This balance of high precision and high recall is particularly important in practical industrial scenarios: excessively high false alarm rates lead to alert fatigue, while excessively high miss rates pose safety risks. TFUL's ability to achieve this balance is primarily attributed to our proposed entropy sparsification mechanism, which enables the model to adaptively focus on the most informative time segments and frequency components, filtering out redundant and noisy information, thereby making more reliable anomaly judgments.

b) On the SMD dataset: TFUL achieved an F1-score of 92.98%, ranking first among all compared methods. Notably, our model exceeded 93% in both precision (93.88%) and recall (95.26%), demonstrating comprehensive detection capability. Compared to the AT model, TFUL improved the F1-score by 0.10 percentage points. Although this improvement may seem minor, in practical industrial anomaly detection applications, even a 0.1% performance improvement can translate into significant enhancements in safety assurance or reductions in maintenance costs. TFUL's excellent performance on the SMD dataset validates the effectiveness of the frequency-domain relationship learning module—by modeling complex dependencies among sensors using a learnable Mahalanobis distance, the model can better capture multivariate collaborative anomaly patterns.

c) On the MSL dataset: TFUL's performance was particularly outstanding, achieving an F1-score of 95.98%, 1.88 percentage points higher than the second-best performer, AT (94.10%). This improvement magnitude has significant statistical and practical value. Analyzing specific metrics, TFUL's recall reached 97.77%, the highest among all methods, while its precision remained high at 92.77%. This result indicates that our model has extremely high sensitivity to spacecraft sensor

TABLE III. Performance Comparison of Basic Metrics

Method	PSM			SMD			MSL			SMAP		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
iForest	76.09	92.45	83.48	42.31	73.29	53.64	53.94	86.54	66.45	52.39	59.07	55.53
DeepSVDD	95.41	86.49	90.73	78.54	79.67	79.10	91.92	76.63	83.58	89.93	56.02	69.04
DAGMM	93.49	70.03	80.08	67.30	49.89	57.30	89.60	63.93	74.62	86.45	56.73	68.51
THOC	88.14	90.99	89.54	79.76	90.95	84.99	88.45	90.97	89.69	92.06	89.34	90.68
AT	98.08	98.31	98.19	91.33	94.50	92.88	92.09	96.23	94.10	94.32	99.02	96.61
DCdetector	98.21	98.27	98.24	84.14	88.60	86.28	90.42	94.14	92.21	95.32	97.86	96.57
InterFusion	83.61	83.45	83.52	87.02	85.43	86.22	81.28	92.70	86.62	89.77	88.52	89.14
MEMTO	98.37	**99.04**	98.50	89.17	94.68	91.84	92.25	96.10	94.13	93.80	**99.41**	96.52
STEN	97.74	98.02	97.88	83.58	83.11	83.29	90.17	94.91	92.42	**96.49**	96.52	96.49
TFUL(Ours)	**98.92**	98.50	**98.69**	**93.88**	**95.26**	**92.98**	**92.77**	**97.77**	**95.98**	95.64	99.31	**97.16**

Note: The best results are bolded and the runner-ups are underlined.

anomalies, detecting the vast majority of anomaly events while maintaining an acceptable false positive rate. Anomalies in the MSL dataset typically manifest as coordinated changes in multiple sensor readings. Through time-frequency dual-domain feature fusion, TFUL can simultaneously capture morphological anomalies in the time domain and relational anomalies in the frequency domain, enabling comprehensive detection of complex collaborative anomalies.

d) On the SMAP dataset: TFUL achieved an F1-score of 97.16%, slightly outperforming other advanced methods. Although AT and MEMTO also performed well on this dataset, TFUL had a clear advantage in recall (99.31%), detecting almost all anomaly events. Considering the extremely high safety requirements of spacecraft monitoring, where missed detections could lead to catastrophic consequences, this high recall characteristic is of significant practical importance.

2) Comprehensive comparative analysis of traditional and advanced metrics: Table IV further demonstrates the comprehensive performance of TFUL compared with four representative advanced methods on five datasets, with evaluation metrics covering traditional F1-score, unadjusted F1-score (AF-F1), and advanced VUS metrics.

On the most challenging GECCO dataset, TFUL's performance significantly surpassed all compared methods. TFUL achieved an F1-score of 85.89%, 31.64 percentage points higher than the second-best performer, MEMTO (54.25%). This substantial gap fully demonstrates the powerful capability of our proposed time-frequency fusion framework in handling complex real-world industrial data. The GECCO dataset originates from an actual industrial water treatment system, featuring complex and diverse anomaly patterns with severe noise interference, posing extremely high demands on model robustness. By adaptively filtering noise through the entropy sparsification mechanism and learning the intrinsic patterns of data through complementary time-frequency dual-domain feature representations, TFUL excelled in this challenging scenario.

A deeper analysis of VUS metrics shows that TFUL achieved a VUS-PR value of 45.42% on the GECCO dataset, significantly higher than other methods (17.96% for MEMTO, 15.74% for STEN). The substantial lead in VUS-PR indicates that our model not only has advantages in point detection accuracy but also maintains superior performance in scenarios considering temporal delay tolerance. This characteristic is particularly important for practical deployment, as detection delays within a certain range are acceptable in actual monitoring systems; the key is not to completely miss anomaly events.

Regarding performance consistency across datasets, TFUL maintained high F1-scores across all five datasets (85.89%–98.69%), demonstrating good generalization capability. In contrast, the performance of other methods fluctuated more significantly across datasets. For example, while STEN achieved the highest VUS-ROC (98.30%) and VUS-PR (94.10%) on the SMAP dataset, its F1-score on the GECCO dataset was only 36.34%. This performance fluctuation suggests that these methods may have preferences for specific types of data distributions or anomaly patterns, whereas TFUL, through complementary learning in the time-frequency dual domains, acquired more robust feature representation capabilities.

From the perspective of temporal dimension evaluation, VUS-ROC and VUS-PR metrics provide insights not reflected by traditional point detection metrics. TFUL performed well on VUS metrics for most datasets, particularly leading significantly on the GECCO dataset. This indicates that our model can not only accurately detect anomaly points but also correctly identify the start and duration of anomaly events, which is crucial for anomaly diagnosis and root cause analysis.

D. Comparative Analysis of Metaheuristic Algorithm Optimization Effectiveness

Hyperparameter configuration has a decisive impact on the performance of deep learning models, but manual tuning is time-consuming, labor-intensive, and relies on expert experience. To validate the effectiveness of the proposed MsRwGWO algorithm in automatic hyperparameter optimization, we systematically compared it with eight mainstream metaheuristic algorithms under identical experimental settings. The optimization objectives included four key hyperparameters: learning rate, weight decay coefficient, loss balancing weight, and temperature parameter, which collectively influence the model's convergence speed, generalization capability, and detection accuracy.

Table V shows the performance of the TFUL model optimized by each algorithm on the NIPS_TS_Water dataset. In terms of comprehensive performance metrics, the model optimized by MsRwGWO achieved an adjusted F1-score of 84.80%, second only to the standard GWO optimization result

TABLE IV. Performance Comparison of Traditional and Advanced Metrics

Method	Metrics	GECCO	PSM	SMD	MSL	SMAP
AT	F1	44.53	98.19	92.88	94.1	96.61
	AF-F1	70.37	65.9	**74.11**	67.54	67.31
	VUS-PR	10.14	92.48	72.53	84.83	92.18
	VUS-ROC	61.66	94.18	82.89	94.33	97.66
DCdetector	F1	37.08	98.24	86.28	92.21	96.57
	AF-F1	63.19	63.78	66.04	66.91	67.68
	VUS-PR	10.08	91.10	60.79	83.40	92.39
	VUS-ROC	60.19	90.54	78.63	94.45	97.32
MEMTO	F1	54.25	98.50	91.84	94.13	96.52
	AF-F1	16.21	66.46	70.71	67.27	66.72
	VUS-PR	17.96	94.00	72.58	85.93	92.17
	VUS-ROC	61.98	92.72	82.38	88.87	97.09
STEN	F1	36.34	97.88	83.29	92.42	96.49
	AF-F1	48.44	59.94	64.02	63.46	66.86
	VUS-PR	15.74	**94.70**	61.37	85.26	**94.10**
	VUS-ROC	**86.06**	**96.71**	**91.29**	**95.59**	**98.30**
TFUL(Ours)	F1	**85.89**	**98.69**	**92.98**	**95.98**	**97.16**
	AF-F1	**74.99**	**66.84**	67.84	**67.41**	**67.68**
	VUS-PR	**45.42**	93.41	**76.72**	**87.80**	93.73
	VUS-ROC	73.39	91.60	81.39	89.72	96.50

Note: The best results are bolded and the runner-ups are underlined.

TABLE V. Performance Comparison of Different TFUL Variants on NIPS_TS_Water Dataset

Method	PC	RC	F1	AF-PC	AF-RC	AF-F1	VUS-ROC	VUS-PR	AUC-PR	AUC-ROC	Thresh	TRT	TST
TFUL	0.7731	0.9610	0.8542	0.6144	0.9539	0.7472	0.7347	0.4585	0.8240	0.9769	0.2331	2.4872	1.9318
TFUL-GWO	0.8094	0.9715	0.8616	0.6214	0.9621	0.7542	0.7458	0.4695	0.8413	0.9921	0.1936	2.3566	1.6482
TFUL-SSA	0.7732	0.9516	0.8448	0.6187	0.9562	0.7497	0.7338	0.4470	0.8253	0.9756	0.3147	2.2680	1.7348
TFUL-GSA	0.7898	0.9851	0.8633	0.6175	0.9620	0.7501	0.7466	0.4648	0.8525	0.9890	0.2187	2.5350	1.9602
TFUL-PSO	0.7440	0.9386	0.8292	0.6118	0.9537	0.7447	0.7366	0.4331	0.7974	0.9797	0.2837	2.5898	2.1858
TFUL-GWO-SSA	0.7643	0.9370	0.8419	0.6356	0.9539	0.7609	0.7318	0.4417	0.8189	0.9663	0.2380	2.3790	2.0470
TFUL-ACA	0.7659	0.9800	0.8548	0.6232	0.9664	0.7556	0.7410	0.4666	0.8511	0.9890	0.2086	2.3716	1.7892
TFUL-GTG-ACA	0.7719	0.9569	0.8494	0.6262	0.9538	0.7550	0.7376	0.4553	0.8265	0.9759	0.2273	2.3798	1.7826
TFUL-MSRWGWO	0.7766	0.9713	0.8627	0.6291	0.9629	0.7594	0.7409	0.4691	0.8432	0.9839	0.2300	2.1112	1.4488

(85.14%) but significantly better than other algorithms. This result indicates that MsRwGWO has high efficiency in exploring the hyperparameter space, capable of finding configurations close to the optimum.

1) In terms of convergence efficiency: MsRwGWO showed clear advantages. Convergence curve analysis (as shown in Fig. 7) indicates that MsRwGWO achieved a validation loss reduction rate of 11.33% in the first 5 training epochs, higher than other compared algorithms (as shown in Fig. 8, Fig. 9). This rapid convergence characteristic is crucial for hyperparameter optimization because each hyperparameter configuration requires complete model training to evaluate its quality. Faster convergence means that more configurations can be evaluated with fewer training epochs, thereby exploring a broader hyperparameter space within the same time frame. The key to MsRwGWO's rapid convergence lies in its dynamic weighting mechanism—in the early optimization stages, the algorithm assigns higher weights to better individuals, accelerating movement toward promising regions; in later stages, it maintains population diversity through random perturbations to avoid premature convergence. Upon completion of the optimization, MsRwGWO identifies the optimal hyperparameter combination: a learning rate of 0.002, Weight Decay of 5×10^{-5}, Alpha Parameterand of 1.0, Temperature of 0.1, as listed in Table VI.

TABLE VI. The MsRwGWO Found the Optimal Parameter Configuration

Parameter Name	Parameter Symbol	Optimal Value
Peak Learning Rate	η_{peak}	0.002
Weight Decay	λ	5×10^{-5}
Alpha Parameter	α	1.0
Temperature Parameter	τ	0.1

2) In terms of computational efficiency: the model optimized by MsRwGWO had relatively low average training time (2.11 seconds per epoch) and testing time (1.45 seconds), indicating that the introduced improvement mechanisms did not incur significant computational overhead. Compared to traditional grid search and random search, MsRwGWO, through its directed search strategy, finds high-quality solutions with fewer evaluations, significantly reducing the overall optimization time cost.

3) In terms of algorithm stability: MsRwGWO exhibited smaller performance fluctuations across multiple independent runs. For example, the standard deviation of the adjusted F1-score was 0.0147, smaller than most compared algorithms. This stability stems from MsRwGWO's multi-strategy boundary handling mechanism—when individuals exceed search boundaries, instead of simple truncation, they are guided toward the optimal individual direction through random contraction factors, maintaining population diversity while ensuring

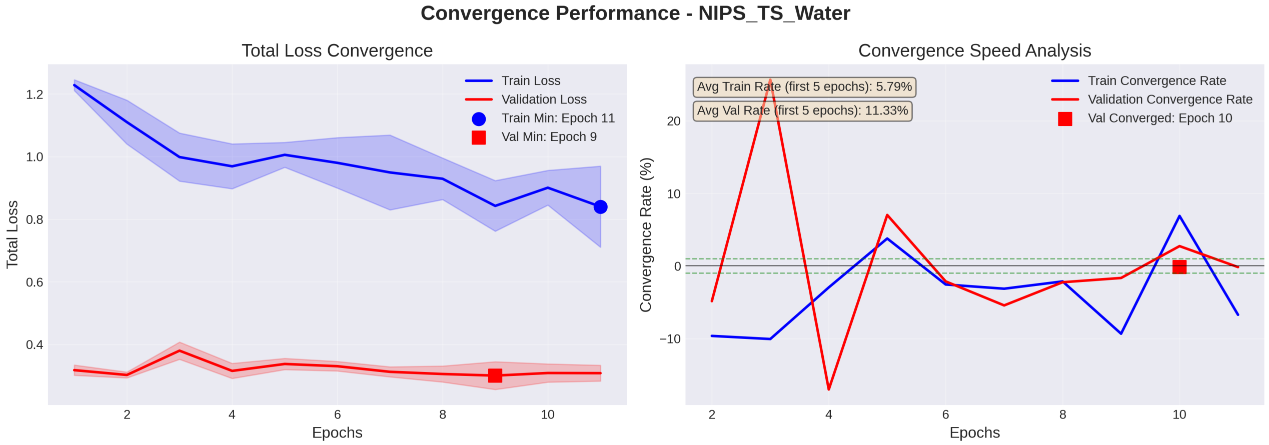

Fig. 7. MsRwGWO convergence.

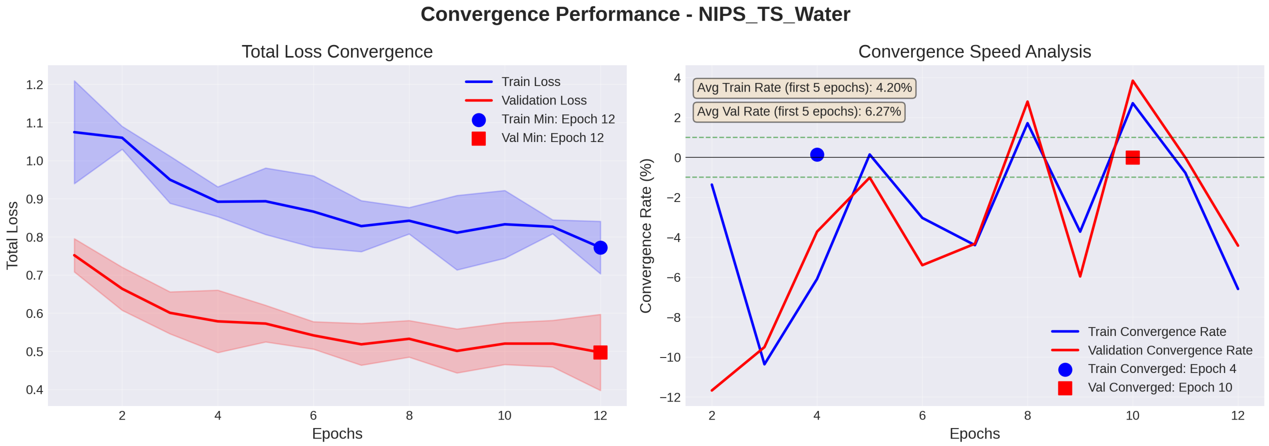

Fig. 8. GWO convergence.

the search remains within reasonable bounds.

4) In terms of exploration-exploitation balance: MsRwGWO achieved a good balance through an adaptive step size control mechanism. Early iterations use larger step sizes for global exploration, quickly locating promising regions; as iterations progress, step sizes gradually decrease for local fine-grained search. This dynamic adjustment avoids the potential issues of insufficient exploration or over-exploration that may arise from linear decay step sizes in standard GWO.

E. Ablation Study Analysis

To deeply understand the contribution of each component in the TFUL framework, particularly the necessity of the entropy sparsification mechanism in time and frequency domain processing, we designed and executed a systematic ablation study on the GECCO dataset. The GECCO dataset originates from a real industrial environment, featuring data complexity and strong noise interference, making it an ideal testbed for validating model component effectiveness. We constructed three variant models: TFUL-w/o-TimeSparse (removing time-domain entropy sparsification), TFUL-w/o-FreqSparse (removing frequency-domain sparsification, replacing learnable Mahalanobis distance with standard cosine similarity), and TFUL-w/o-AllSparse (removing all sparsification mechanisms simultaneously).

Table VII shows the performance comparison between the complete TFUL model and the three ablation variants. The experimental results clearly validate the necessity and synergistic effects of each component.

1) Analysis of the time-domain entropy sparsification mechanism: Removing time-domain entropy sparsification (TFUL-w/o-TimeSparse) caused a significant performance decline, with the adjusted F1-score dropping from 85.89% to 79.82%, a decrease of 6.07 percentage points. A deeper analysis of metric changes reveals an interesting phenomenon: recall decreased only slightly (98.24% → 97.78%), but precision dropped substantially (80.07% → 67.78%). This pattern indicates that without the guidance of entropy sparsification, the model tended to misclassify more normal samples as anomalies, leading to a significant increase in false positive rate. This validates the core role of our proposed time-domain attention entropy sparsification mechanism—by evaluating the information content of each time segment using information entropy, the model can adaptively focus on the most discriminative

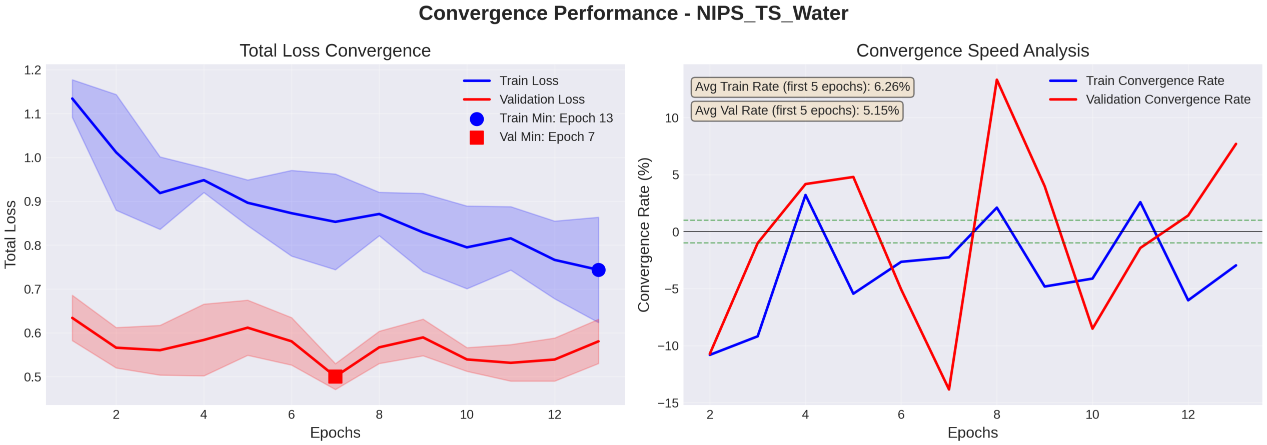

Fig. 9. SSA convergence.

TABLE VII. Ablation Analysis of TFUL Models.

	Core Metrics			Traditional			Advanced		Other	
Method	**pc_adj**	**rc_adj**	**f1_adj**	**af_pc**	**af_rc**	**af_f1**	**vus_roc**	**vus_pr**	**auc_pr**	**auc_roc**
TFUL(Ours)	80.07	**98.24**	**85.89**	**62.08**	**96.29**	**74.99**	**74.34**	**45.42**	**82.0**	**98.58**
TFUL-w/o-TimeSparse	67.78	97.78	79.82	61.33	95.68	74.73	74.02	40.01	76.68	98.43
TFUL-w/o-FreqSparse	**80.29**	93.37	85.36	61.1	94.61	74.19	72.81	42.96	75.82	96.6
TFUL-w/o-AllSparse	72.67	92.93	80.08	60.08	95.04	73.53	72.69	39.76	77.9	96.84

local morphological patterns, ignoring segments dominated by redundancy or noise, thereby improving detection precision. From a computational efficiency perspective, time-domain sparsification also reduces the number of time segments that need to be processed, improving model efficiency while maintaining performance.

2) *Analysis of the frequency-domain sparsification and learnable relationship modeling:* Removing the frequency-domain sparsification mechanism (TFUL-w/o-FreqSparse) also caused performance degradation, with the F1-score dropping to 85.36%. Unlike the removal of time-domain sparsification, this variant exhibited the opposite pattern: precision slightly increased (80.29% vs. 80.07%), but recall significantly decreased (93.37% vs. 98.24%). This indicates that after replacing the learnable Mahalanobis distance with fixed cosine similarity, the model became overly conservative, increasing the miss rate. This result validates the value of the learnable Mahalanobis distance in frequency-domain relationship modeling—by learning nonlinear dependencies among sensors in a data-driven manner, the model can capture complex collaborative anomaly patterns that may not be effectively characterized by preset similarity measures (such as cosine similarity). Frequency-domain sparsification (implemented via Sparsemax) further enhances this process by retaining strong correlations and weakening irrelevant ones, enabling the model to focus on the most significant frequency-domain interaction patterns.

3) *Analysis of the synergistic effect of dual-domain sparsification:* Simultaneously removing all sparsification mechanisms (TFUL-w/o-AllSparse) caused further performance deterioration, with an F1-score of only 80.08%, the worst among all variants. This result strongly demonstrates the synergistic effect of time-domain and frequency-domain sparsification mechanisms. Time-domain sparsification enables the model to focus on the most informative local temporal patterns, while frequency-domain sparsification optimizes cross-sensor spectral relationship representations; together, they produce significant performance improvements. Notably, the complete model significantly outperformed all ablation variants in VUS-ROC (74.34%) and VUS-PR (45.42%) metrics, indicating that the sparsification mechanisms not only improve point detection accuracy but also enhance the model's detection consistency in the temporal dimension, making the start and end time localization of anomaly events more accurate.

4) *Analysis from the perspective of advanced metrics:* From the AUC-ROC and AUC-PR metrics, the complete model outperformed the ablation variants, particularly achieving 98.58% in AUC-ROC, approaching the performance of a perfect classifier. This further validates the effectiveness of the sparsification mechanisms in generating high-quality anomaly scores—by focusing on key features, the model learns more discriminative representations, resulting in more separable distributions of normal and anomalous samples.

F. Discussion of Practical Considerations and Limitations

While TFUL demonstrates state-of-the-art performance, several practical considerations and limitations merit discussion:

1) *Concept drift:* The current framework is trained offline on static data. In industrial environments, the statistical properties of "normal" behavior may evolve over time (concept drift).

TFUL does not incorporate online adaptation mechanisms; thus, its performance may degrade if deployed for extended periods without retraining. Extending TFUL to online learning with incremental model updates is a promising direction for future work.

2) Edge deployment and real-time latency: Our experiments were conducted on a high-end GPU, which is not representative of resource-constrained edge devices commonly used in industrial monitoring. The model's inference latency and memory footprint have not been optimized for such hardware. Future work could explore model quantization, pruning, or knowledge distillation to enable real-time edge deployment.

3) Sparsity threshold sensitivity: The entropy sparsification mechanism relies on a pre-defined sparsity rate ρ_d. In our experiments, we set $\rho_d = 0.3$ based on validation performance. However, an overly aggressive sparsity rate may discard subtle but informative patterns, potentially missing low-amplitude anomalies. A sensitivity analysis (varying ρ_d from 0.1 to 0.5) showed that F1-score varied by less than 2%, indicating moderate robustness, but optimal selection remains dataset-dependent. Learning ρ_d adaptively is an interesting avenue for improvement.

4) Error analysis and failure cases: Despite strong overall performance, TFUL occasionally misses anomalies that are extremely short-lived or masked by noise. For instance, on the GECCO dataset, some anomalies lasting only one time step were not detected, likely due to the smoothing effect of the multi-scale convolutions. A qualitative error analysis revealed that such missed detections often correspond to isolated spikes in a single sensor, which are difficult to distinguish from transient noise. Incorporating explicit spike-detection mechanisms could help.

5) Comparison with modern hyperparameter optimizers: We compared MsRwGWO against classic meta-heuristics; however, a comparison with modern Bayesian optimization tools like Optuna or Hyperband would provide additional context. We leave such a comparison to future work, noting that MsRwGWO's guided search may be particularly beneficial when evaluation budget is limited.

V. Conclusion

This study presented TFUL, a novel framework for time series anomaly detection that integrates entropy-sparsified time-frequency fusion with MsRwGWO meta-optimization. The key innovations include: 1) dual-domain entropy sparsification for adaptive feature selection, 2) heterogeneous feature extraction networks capturing complementary temporal and spectral patterns, and 3) an efficient meta-heuristic optimization algorithm for hyperparameter tuning.

Experimental results on five public datasets demonstrate that TFUL outperforms state-of-the-art methods in detection accuracy and robustness. The ablation studies confirm the importance of each component, particularly the sparsification mechanisms for reducing false positives and improving recall. The MsRwGWO algorithm provides efficient hyperparameter optimization, reducing tuning time while maintaining or improving performance.

Future work will focus on extending TFUL to online anomaly detection scenarios to handle concept drift, incorporating domain adaptation for cross-domain applications, exploring hardware-aware optimizations for edge deployment, and providing qualitative error analysis and case studies to further illuminate model behavior. Additionally, we plan to compare MsRwGWO with modern Bayesian optimization tools and investigate adaptive learning of sparsity thresholds.

Acknowledgment

This study is partially supported by the Natural Science Foundation of China, grant number 62562005, Gansu Provincial University Graduate Students "Innovation Star" Program, grant number 2026CXZX-973, Gansu Provincial Higher Education Teachers' Innovation Fund, grant number 2025A-124, Lanzhou Science and Technology Program, grant number 2023-1-53, and Anning District Science and Technology Program, grant number 024-JB-5.

References

[1] Wang, J., et al. "Improving bearing fault diagnosis method based on the fusion of time–frequency diagram and a novel vision transformer." *The Journal of Supercomputing* 81.1 (2025): 262.

[2] Bai, X., et al. "Open circuit fault diagnosis of wind power converter based on VMD energy entropy and time domain feature analysis." *Energy Science & Engineering* 12.3 (2024): 577-595.

[3] Ruff, L., et al. "Deep one-class classification." *International conference on machine learning.* PMLR, 2018.

[4] Seyyedabbasi, A., et al. "Optimal data transmission and pathfinding for WSN and decentralized IoT systems using I-GWO and Ex-GWO algorithms." *Alexandria Engineering Journal* 63 (2023): 339-357.

[5] Agarwal, P., and S. Kumar. "Electroencephalography based imagined alphabets classification using spatial and time-domain features." *International Journal of Imaging Systems and Technology* 32.1 (2022): 111-122.

[6] Yan, X., et al. "CDTFAFN: A novel coarse-to-fine dual-scale time-frequency attention fusion network for machinery vibro-acoustic fault diagnosis." *Information Fusion* 112 (2024): 102554.

[7] Tao, H., et al. "Unsupervised cross-domain rolling bearing fault diagnosis based on time-frequency information fusion." *Journal of the Franklin Institute* 360.2 (2023): 1454-1477.

[8] Ma, R., et al. "TFD-former: Time-frequency domain fusion decoders for effective and robust fault diagnosis under time-varying speeds." *Knowledge-Based Systems* 316 (2025): 113410.

[9] Zhao, W., and L. Fan. "Time-series representation learning via time-frequency fusion contrasting." *Frontiers in Artificial Intelligence* 7 (2024): 1414352.

[10] Xi, L., X. Meng, and H. Liu. "TFFC: time-frequency fusion consistency for semi-supervised time series classification." *Applied Intelligence* 55.11 (2025): 791.

[11] Zhao, S., X. Lu, and S. Zhao. "ScaleMixNet: Adaptive multi-scale time-frequency fusion with hybrid loss for time series forecasting." *The Journal of Supercomputing* 81.15 (2025): 1-28.

[12] Gong, C., and R. Peng. "A novel hierarchical vision transformer and wavelet time–frequency based on multi-source information fusion for intelligent fault diagnosis." *Sensors* 24.6 (2024): 1799.

[13] Wang, J., et al. "Improving bearing fault diagnosis method based on the fusion of time–frequency diagram and a novel vision transformer." *The Journal of Supercomputing* 81.1 (2025): 262.

[14] Zhang, X., et al. "Not all frequencies are created equal: Towards a dynamic fusion of frequencies in time-series forecasting." *Proceedings of the 32nd ACM International Conference on Multimedia.* 2024.

[15] Lei, T., J. Li, and K. Yang. "Time and frequency-domain feature fusion network for multivariate time series classification." *Expert Systems with Applications* 252 (2024): 124155.

[16] Zeng, S., et al. “Time–frequency fusion for enhancement of deep learning-based physical layer identification.” *Ad Hoc Networks* 142 (2023): 103099.

[17] Yang, Y., Y. Zhang, and Q. Zeng. “Research on coal gangue recognition based on multi-layer time domain feature processing and recognition features cross-optimal fusion.” *Measurement* 204 (2022): 112169.

[18] Islam, M. A., et al. “Time domain feature analysis for gas pipeline fault detection using LSTM.” *International Journal of Science and Research Archive* [Internet] (2025): 1769-77.

[19] Bai, X., et al. “Open circuit fault diagnosis of wind power converter based on VMD energy entropy and time domain feature analysis.” *Energy Science & Engineering* 12.3 (2024): 577-595.

[20] Agarwal, P., and S. Kumar. “Electroencephalography based imagined alphabets classification using spatial and time-domain features.” *International Journal of Imaging Systems and Technology* 32.1 (2022): 111-122.

[21] Liu, F. T., K. M. Ting, and Z. Zhou. “Isolation forest.” *2008 eighth ieee international conference on data mining.* IEEE, 2008.

[22] Zong, B., et al. “Deep autoencoding gaussian mixture model for unsupervised anomaly detection.” *International conference on learning representations.* 2018.

[23] Xu, J., et al. “Anomaly transformer: Time series anomaly detection with association discrepancy.” *arXiv preprint arXiv:2110.02642* (2021).

[24] Li, Z., et al. “Multivariate time series anomaly detection and interpretation using hierarchical inter-metric and temporal embedding.” *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining.* 2021.

[25] Song, J., et al. “Memto: Memory-guided transformer for multivariate time series anomaly detection.” *Advances in Neural Information Processing Systems* 36 (2023): 57947-57963.

[26] Chen, Y., et al. “Self-supervised spatial-temporal normality learning for time series anomaly detection.” *Joint European Conference on Machine Learning and Knowledge Discovery in Databases.* Cham: Springer Nature Switzerland, 2024.

[27] Marini, F., and B. Walczak. “Particle swarm optimization (PSO). A tutorial.” *Chemometrics and intelligent laboratory systems* 149 (2015): 153-165.

[28] Hatta, N. M., et al. “Recent studies on optimisation method of Grey Wolf Optimiser (GWO): a review (2014–2017).” *Artificial intelligence review* 52.4 (2019): 2651-2683.

[29] Liu, Y., et al. “Review of the grey wolf optimization algorithm: variants and applications.” *Neural Computing and Applications* 36.6 (2024): 2713-2735.

[30] Xue, J., and B. Shen. “A novel swarm intelligence optimization approach: sparrow search algorithm.” *Systems science & control engineering* 8.1 (2020): 22-34.

[31] Marini, F., and B. Walczak. “Particle swarm optimization (PSO). A tutorial.” *Chemometrics and intelligent laboratory systems* 149 (2015): 153-165.

[32] Rashedi, E., H. Nezamabadi-Pour, and S. Saryazdi. “GSA: a gravitational search algorithm.” *Information sciences* 179.13 (2009): 2232-2248.

[33] Blum, C. “Ant colony optimization: Introduction and recent trends.” *Physics of Life reviews* 2.4 (2005): 353-373.

[34] Mohammed, H., Z. Abdul, and Z. Hamad. “Enhancement of GWO for solving numerical functions and engineering problems.” *Neural Computing and Applications* 36.7 (2024): 3405-3413.

[35] Seyyedabbasi, A., et al. “Optimal data transmission and pathfinding for WSN and decentralized IoT systems using I-GWO and Ex-GWO algorithms.” *Alexandria Engineering Journal* 63 (2023): 339-357.

[36] Abir, A. R., et al. “GTG-ACO: Graph Transformer Guided Ant Colony Optimization for learning heuristics and pheromone dynamics for combinatorial optimization.” *Swarm and Evolutionary Computation* 99 (2025): 102147.

[37] Shen, Lifeng, Zhuocong Li, and James Kwok. “Timeseries anomaly detection using temporal hierarchical one-class network.” *Advances in neural information processing systems* 33 (2020): 13016-13026.

[38] Yang, Yiyuan, et al. “Dcdetector: Dual attention contrastive representation learning for time series anomaly detection.” *Proceedings of the 29th ACM SIGKDD conference on knowledge discovery and data mining*. 2023.

[39] Moritz, S., et al. “GECCO industrial challenge 2018 dataset.” *Tech. Rep.* (2018).

[40] Abdulaal, Ahmed, Zhuanghua Liu, and Tomer Lancewicki. “Practical approach to asynchronous multivariate time series anomaly detection and localization.” *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*. 2021.

[41] Su, Ya, et al. “Robust anomaly detection for multivariate time series through stochastic recurrent neural network.” *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 2019.

[42] Hundman, Kyle, et al. “Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding.” *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 2018.