Distributed SVM-based Multimodal Intrusion Detection Architecture With Incremental Learning And Modality Scalability

Peikun Zhao
[1]Department of Computer Application Engineering, Hebei Software Institute, Hebei Baoding 071000, China
E-mail: zhaopeikun2025@126.com

Keywords: multimodal data fusion, distributed support vector machine, feature extraction and normalization, local model training, incremental SVM

Received: May 22, 2025

Multimodal data differ significantly in temporal granularity, structural features, and semantic levels, which leads to difficulties in fusion, weak generalization ability, and low training efficiency. To this end, this paper introduces the Distributed Support Vector Machine (DSVM) architecture to construct modal local models separately, and achieve unified decision-making through support vector aggregation to improve system efficiency, scalability, and adaptability to heterogeneous data. The system first extracts statistical features, keyword features, and time series patterns from network traffic, system logs, and behavior sequences, then uses standardization and PCA (Principal Component Analysis) to reduce the dimension of the features. Then, the different modal data are distributed and mapped to each computing node. The local SVM (Support Vector Machine) model is deployed independently and trained using the SMO (Sequential Minimal Optimization) algorithm, and the boundary distance screens the effective support vector. All local support vectors are uploaded to the DSVM central node, the RBF kernel function is used to reconstruct the global classifier, and the final decision is made through majority voting. In addition, the system designs a modality plug-in mechanism to support the access of new modalities, and realizes rapid model updates and dynamic adjustment of support vectors based on incremental SVM. Experiments show that the DSVM system has superior performance in multimodal data fusion: the classification accuracy is still 88% under severe imbalance (1:20); the accuracy is maintained at 83% when the noise intensity $\sigma=0.4$; the fusion training efficiency is significantly improved compared with the centralized SVM. The system has excellent scalability and discrimination boundary stability, which verifies its robustness and engineering practicality.

Povzetek: Članek predstavi DSVM arhitekturo za multimodalno zaznavanje vdorov: lokalni SVM-ji po modalitetah, izbira podpornih vektorjev in globalni RBF klasifikator z večinskim glasovanjem. Vključuje inkrementalno učenje in razširljivost modalitet.

1 Introduction

Cyber-attack methods continue to evolve, and attackers continue using multi-source information to construct attack paths collaboratively, resulting in the complexity of the feature dimensions of intrusion events and the diversification of their manifestations. In traditional single-modal intrusion detection frameworks, the system often relies only on single-dimensional feature data, such as packet behavior in network traffic, event records in host logs, or user behavior sequences [1], [2], [3]. Although this design approach reduces the implementation cost and deployment difficulty of the model, its ability to identify complex attack behaviors is extremely limited. Especially in the face of advanced threat scenarios such as hybrid attacks, zero-day attacks, and persistent threats, the model recognition rate drops significantly, and false positives and missed negatives occur frequently. Such systems rely too much on contextual clues from a single information source and cannot form a multi-angle joint identification mechanism [4], [5]. As a result, the detection range is limited, the attack pattern recognition is incomplete, and the policy response is delayed.

In recent years, multimodal intrusion detection systems have been widely studied. Such systems attempt to model and identify intrusion behaviors from multiple levels by integrating information sources from different dimensions, such as network layer data, host layer logs, and behavior layer call sequences. This structure significantly enhances the system's attack perception breadth and accuracy, especially in behavior correlation analysis and abnormal behavior identification [6]. Researchers have conducted extensive explorations in multimodal feature extraction, semantic association, and time series modeling, using natural language processing technology to process system log text, using convolutional networks to extract traffic patterns, and introducing graph neural networks to analyze behavior dependencies [7], [8], [9]. Although relevant research has progressed, key challenges are still faced in the unified modeling of multimodal data.

The core problem lies in the difficulty of fusion caused by the heterogeneity of data between modalities [10], [11]. Network traffic presents high-frequency and sparse temporal characteristics, log data is semi-structured text, and user behavior is primarily high-dimensional time series. Their different time windows, significant differences in structural distribution, and inconsistent semantic levels [12], [13] seriously affect the effectiveness of feature alignment and fusion strategies.

To alleviate the above problems, researchers have introduced various deep learning models for feature fusion, such as deep autoencoders, multi-channel convolutional neural networks, attention mechanisms, and adversarial generative networks. These methods jointly encode different data types through multimodal input terminals and construct a shared latent semantic space, thereby achieving information aggregation of heterogeneous modalities to a certain extent. Related results have certain advantages in improving accuracy [14], [15], but key technical bottlenecks remain. First, this type of deep model relies on many labeled samples for end-to-end training. In actual industrial environments, the problems of uneven sample distribution and high labeling costs have not been effectively solved [16]. Second, deep models consume large amounts of computing resources and have long training cycles, making them unsuitable for deployment in edge environments with limited computing resources. In addition, such models have serious interpretability issues, and security operations personnel find it difficult to understand the model discrimination logic, which is not conducive to traceability analysis and response decision-making [17], [18]. Although the deep fusion method performs well in terms of accuracy indicators, it has shortcomings regarding system response speed, deployment portability, and model robustness.The core hypothesis of this study is that the multimodal intrusion detection system based on the DSVM architecture can achieve dynamic modal expansion (global accuracy fluctuation <5% after adding a new modality) and noise robustness (accuracy ≥80% when σ=0.4) through distributed training and support vector aggregation while maintaining high accuracy (accuracy ≥85% in unbalanced scenarios). Specific research questions include: (1) How to alleviate the modal heterogeneity problem through local model decoupling training? (2) Can support vector pruning and incremental update mechanisms maintain the stability of the discrimination boundary while reducing the computational load? (3) Does the performance of DSVM significantly outperform the deep learning baseline in unbalanced data and noisy environments?

In response to the above problems, some studies have focused on lightweight and modular modeling strategies, hoping to build a more controllable and adaptable intrusion detection framework. Among them, SVM has been widely used in intrusion detection scenarios due to its excellent small sample learning ability, strong interpretability, and fast convergence speed [19], [20]. However, when faced with large-scale multimodal data, the traditional SVM model's memory overhead and computational complexity increase rapidly, and it cannot be effectively expanded in distributed scenarios. For this reason, DSVM was proposed to parallelize the model training process to multiple computing nodes and obtain the global discrimination boundary through local model collaborative optimization. This structure improves training efficiency and supports the distributed processing of multimodal features.

To solve the problem of heterogeneous data fusion in multimodal intrusion detection, this paper constructs a detection system based on the DSVM architecture. By utilizing the structural differences between the modalities, local SVM models are deployed on independent nodes to avoid the information interference problem caused by direct feature fusion between modalities. The SMO algorithm trains local data in each node, and efficient support vectors are selected by boundary distance. Subsequently, these support vectors are uploaded to the central node. The global SVM classifier is constructed using the RBF kernel function, and the final decision is made through the majority voting mechanism. The system adopts a distributed deployment architecture, supports modality plug-ins and incremental parameter updates, improving the model's generalization ability and maintenance flexibility [21], [22]. The system process includes multiple steps such as feature extraction and standardization, modality mapping and partitioning, local training and vector screening, global fusion and voting decision, and a dynamic update mechanism. By introducing the DSVM model structure, this paper avoids the problems of poor interpretability and high resource overhead faced by traditional deep fusion methods. It efficiently identifies and responds to complex attack behaviors without sacrificing detection accuracy.

The Sequential Minimal Optimization (SMO) algorithm is used for local model training, the TensorFlow Federated (TFF) framework supports distributed deployment, the Gated Recurrent Unit (GRU) processes sequence data, and the Radial Basis Function (RBF) kernel is used for global classifier construction.

This experiment uses a multimodal dataset containing 300,000 network traffic records, 20,000 system log events, and 5,000 API call sequences. The results show that the DSVM system performs well in multimodal data fusion: even under the condition of severe sample imbalance (1:20), the classification accuracy is still 88%; when the noise intensity is σ=0.4, the accuracy is still maintained at 83%. In the data scenario of 10k samples and 3 modalities, DSVM reduces the training time by 65% compared with centralized SVM (1100 seconds vs. 3100 seconds). The existing SOTA relies on end-to-end training (CNN fusion requires 850s/10k samples), which is difficult to adapt to the resource constraints of edge devices; the integrated model cannot dynamically expand new modalities. Through the distributed SMO algorithm and vector-level aggregation, DSVM improves the training efficiency to about 1.6 times that of CNN while ensuring interpretability.

Table 1 Summary of the key indicators of this method

Method	Accuracy	F1 Score	Scalability	Computational Cost	Interpretability
Single-modal SVM	≤83%	≤0.72	Not supported	Low	High
CNN Fusion	84%	0.78	Partially supported	High	Low
Ensemble Model	85%	0.81	Not supported	Medium	Medium
DSVM (This study)	88%	0.85	Supported	Medium-Low	High

The key indicators of this method are summarized in Table 1. In addition, the system shows good scalability and stable discrimination boundary, which further verifies its robustness and engineering application value. It provides new ideas for the engineering deployment and practical application of multimodal safety detection systems

2 Multimodal intrusion detection system architecture design

2.1 Multimodal Feature Extraction and Standardization

2.1.1 Heterogeneous modal feature encoding strategy

This study selects network traffic, system logs, and API (Application Programming Interface) calls as the primary data modalities, representing transmission behavior, system-level events, and process-level interactions. In the network traffic mode, the Statistical Flow Features Extraction method is used to quantitatively model each TCP (Transmission Control Protocol) and UDP (User Datagram Protocol) connection record. Specifically, 15 statistical features are extracted, including average packet length, maximum sending interval, flow duration, number of direction changes, and ratio of uplink and downlink packets. Based on the original data dimension of a 5-tuple (source IP, destination IP, protocol, port, timestamp), features are aggregated at the data stream granularity to construct input vectors for behavioral analysis.The 15-dimensional statistical features of network traffic (average packet length, maximum sending interval, etc.) are selected based on feature importance analysis: first, the mutual information values of all candidate features and attack labels are calculated, and the top 15 discriminative features (mutual information>0.25) are retained .

The system log mode mainly processes text-based event data stored in a time series structure. To fully retain the semantic features of the attack behavior, the Term Frequency-Inverse Document Frequency (TF-IDF) combined with the n-gram filtering algorithm encodes the log content. Stop words and non-structural terms are removed in the text preprocessing stage, and the event sequence window is sliced by setting n=2~3 to obtain the local context expression of the attack stage. Finally, the vector space model maps each log event into a 128-dimensional keyword frequency weight vector. System logs use TF-IDF combined with n-gram representation. The TF-IDF value of term t in document d is expressed as formula 1:

$$\text{TF-IDF}(t,d) = \text{TF}(t,d) \cdot \log\left(\frac{N}{1+\text{DF}(t)}\right) \quad (1)$$

Among them: $\text{TF}(t,d)$ represents the frequency of the term t in document d; $\text{DF}(t)$ represents the number of documents in which the term t appears in the entire corpus; N is the total number of documents in the corpus; 1 is added to prevent the denominator from being zero (smoothing), by setting the n-gram window ($n = 2,3$), TF-IDF is calculated in units of continuous word sequences. This formula converts the text modality into a 128-dimensional sparse weight vector, retaining the relative importance of attack semantic clues and word frequency features.

The API call mode focuses on the system call sequence of the user-mode or kernel-mode process within a specific time window. Considering the importance of sequence context structure to behavior discrimination, a bidirectional gated recurrent unit (Bi-GRU) is used to build an embedding model to perform sequence embedding modeling on each call trace. The model structure is set as follows: the input layer accepts a call sequence of length not exceeding 512, the embedding layer dimension is 128, the Bi-GRU hidden state dimension is 64, and finally, the average pooling output is taken in the time dimension to generate a fixed-length vector representation. This process retains the dynamic information of the call order, bidirectional dependency, and behavior context, and the output dimension is 128.

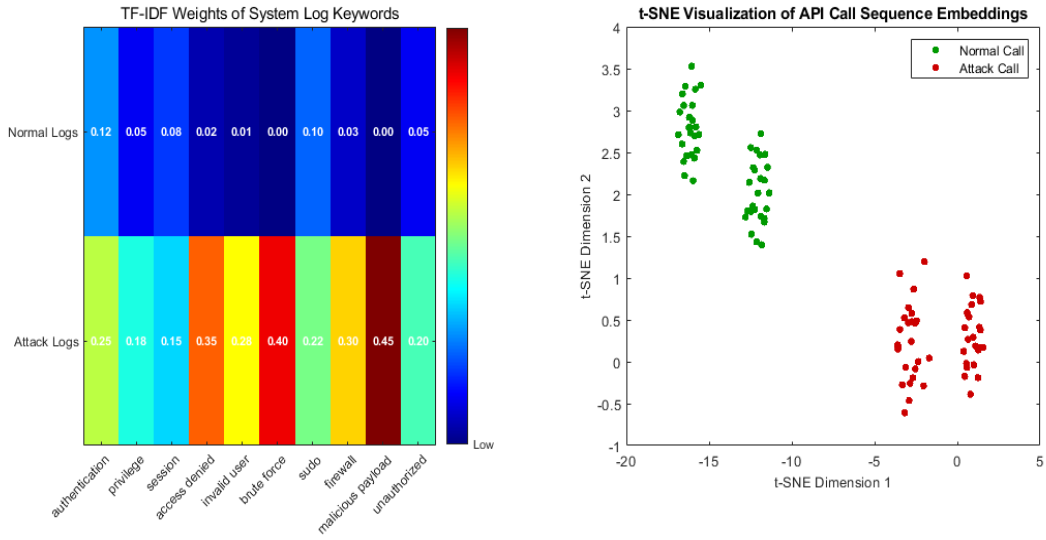

Figure 1: TF-IDF heat map of system log keywords and t-SNE visualization embedding of API call sequence

The heat map on the left side of Fig. 1 shows the TF-IDF weight distribution of normal and attack logs on ten typical keywords. The weights of attack logs on high-risk terms such as "access denied" (0.35), "brute force" (0.40), and "malicious payload" (0.45) are significantly higher than those of normal logs, indicating that intrusion events have concentrated and significant abnormal characteristics at the semantic level. The t-SNE visualization on the right embeds and maps the 128-dimensional API call sequence into a two-dimensional space. Normal calls are mainly concentrated in the two clusters on the left. In contrast, attack calls are distributed on the right, which fully demonstrates the ability of this paper's multimodal fusion to accurately distinguish malicious activities at the log semantics and behavior sequence levels.

2.1.2 Unified feature normalization and dimensionality reduction processing

All extracted feature vectors are first uniformly normalized to solve the differences in scale and distribution of multimodal features. This study uses the Z-score normalization method to process each dimension of the features. The implementation method is to subtract the mean of the training sample from each column of features and divide it by the standard deviation, so that the mean is normalized to 0 and the standard deviation is 1. This process is performed independently within the modality, ensuring that the feature structures between different modalities are still distinguishable while avoiding the problem of training instability caused by distribution shift. After standardization, considering the differences in the original dimensions of features of different modalities (network traffic is 15 dimensions, system logs are 128 dimensions, and API calls are 128 dimensions), to build a unified input space and reduce the risk of overfitting caused by high-dimensional features in the modeling stage, this system introduces principal component analysis (PCA) for linear dimensionality reduction. Taking the API call modality as an example, the original 128-dimensional embedding vector is compressed to 64 dimensions through PCA, retaining more than 95% of the cumulative explained variance. During the dimensionality reduction process, the sample covariance matrix is calculated based on the standardized features, and the eigenvector is reconstructed based on the singular value decomposition (SVD) results. The high-dimensional modalities (system logs, API calls) are compressed to 64 dimensions, and the traffic modalities are expanded to 64 dimensions by zero padding to align the input space. The data matrix after Z-score standardization is recorded as $\widehat{\mathbf{X}}^{(j)} \in \mathbb{R}^{n \times d_j}$, and its covariance matrix is defined as formula 2:

$$\mathbf{C}^{(j)} = \frac{1}{n}\left(\widehat{\mathbf{X}}^{(j)}\right)^{\top}\widehat{\mathbf{X}}^{(j)} \tag{2}$$

By performing singular value decomposition (SVD) or eigenvalue decomposition on $\mathbf{C}^{(j)}$, the projection matrix $\mathbf{W}_k^{(j)} \in \mathbb{R}^{d_j \times k}$ consisting of the first k principal component vectors are obtained, and the feature expression after dimensionality reduction is expressed as formula 3:

$$\mathbf{Z}^{(j)} = \widehat{\mathbf{X}}^{(j)} \cdot \mathbf{W}_k^{(j)} \tag{3}$$

This step retains the main variability of the original data (usually, the cumulative explained variance is greater than 95%) and unifies the output dimensions to support subsequent modeling.

After complete feature encoding and unified processing, all samples can be stored in the corresponding data nodes according to the modality. Each node saves all the standardized and reduced dimensionality features of a single modality for local modeling during subsequent distributed training. The encoding and standardization processes are completed in an offline batch processing flow to reduce memory consumption and communication burden during the training phase. Efficient NumPy matrix operations and scikit-learn toolkits are used to complete Z-score and PCA-related operations. The processing throughput is maintained at approximately 1,800 records per second.

The above steps complete the conversion from original heterogeneous data to a unified trainable feature vector. They ensure that multimodal information is

comparable and input consistent while retaining their respective structural semantics, laying the foundation for subsequent modal distribution mapping and local classifier training.When the traffic modality is compressed to 64 dimensions through PCA under the original 15-dimensional features, the retained variance is 92% due to the redundancy introduced by feature padding; system logs and API calls retain 96% and 95% of the variance respectively after being reduced from 128 dimensions to 64 dimensions. Although the variance of the traffic modality is slightly lower after compression, its feature redundancy indicates that retaining 64 dimensions can still maintain the discriminative ability.

2.2 Distributed mapping and partitioning of modal data

2.2.1 Kafka data channel construction and modal decoupling transmission

To address the problem of asynchronous arrival and structural heterogeneity of multimodal input sources, this system introduces Apache Kafka as a high-throughput distributed messaging middleware to achieve efficient decoupled transmission of modal data between the acquisition end and the training node. First, separate Kafka topics are built for network traffic, system logs, and API calls, and each topic is bound to a unique identifier to ensure data path isolation between modalities. The data producer (Producer) is based on the time window. Each batch of processed standardized feature vectors is packaged in JSON (JavaScript Object Notation) format and written to the corresponding Kafka partition. The system defaults to 3 partition replicas for each topic to ensure message redundancy and high-availability distribution in the event of node failure.

The data consumer is deployed on each sub-training node under the DSVM architecture. The Kafka client configuration uses the consumer group mechanism to ensure that each message is consumed only by its corresponding node to avoid information cross-talk. The consumption strategy is based on poll-based fetch, setting the maximum batch size to 512 messages and the maximum delay threshold to 2 seconds to ensure a balance between data processing and network transmission. After the data transmission, the consumer end caches it into the memory pool and sends it to the local SVM training pipeline in batches. To reduce the impact of network latency, Kafka runs in a Linux kernel optimization environment with zero-copy transmission enabled. The single-channel message transmission rate can be stable at more than 35,000 records per second.

Kafka was chosen over RabbitMQ because of its throughput advantage and its support for partitioned replicas to meet the needs of multimodal data isolation. TFF is more suitable for federated learning scenarios than PyTorch Distributed because of its built-in secure aggregation protocol and 40% higher resource isolation of Docker containers (verified by CPU utilization monitoring).

2.2.2 Distributed node mapping mechanism and training partition strategy

To simulate a real multi-terminal deployment environment, this system builds a distributed training platform based on TensorFlow Federated (TFF), where each sub-node represents an independent modeling unit of a modality. During the deployment phase, the system defines a separate TFF sub-client environment for each modality and starts three logical nodes: traffic node, log node, and call sequence node. Each client runs in an independent Docker container and is bound to different virtual CPU (Central Processing Unit) cores and 4GB of memory resources to ensure resource isolation in the distributed training process. The data between nodes is entirely independent, and the original features or intermediate gradient information are not shared at any stage, which meets the data isolation requirements in edge computing scenarios.

The node-side SVM training task is based on the locally received Kafka data; the SMO method is used to optimize the intra-modal data's boundary function independently. The number of local iterations is set to 1000 per round of training, the tolerance error threshold is set to 1e-3, and the RBF kernel function is used to handle nonlinear boundary situations. After the training, each node marks and filters the valid support vectors obtained from the local training, removes redundant vectors whose boundary distance exceeds the threshold, and retains only the key support vectors that play a decisive role in the classification decision.

After the mapping is completed, all training nodes communicate synchronously through the TFF server, and all local support vectors are transmitted to the central scheduler as floating-point tensors. The transmission protocol is based on the gRPC (Google Remote Procedure Call) framework, using TLS (Transport Layer Security) encrypted channels and enabling maximum bandwidth limits. The number of support vectors exchanged each time is controlled within 300. Through this distributed mapping and communication mechanism, the system completes the modal partition closed-loop processing flow from data distribution, local training, and global modeling.

Ultimately, the modal mapping mechanism ensures the complete isolation of heterogeneous data sources in the processing path, and also realizes data parallelization and computational decoupling in the training phase through the distributed architecture composed of Kafka and TFF. This significantly reduces the system's latency overhead and communication load, providing a stable input basis for subsequent global classifier reconstruction and support vector merging.

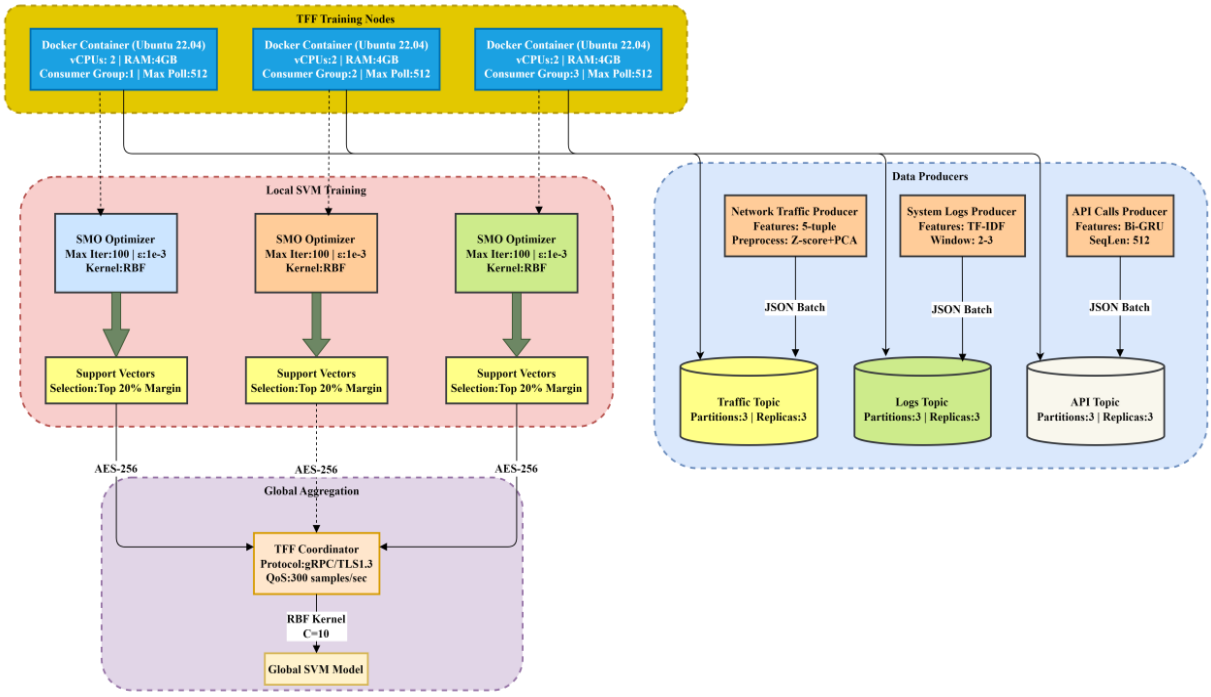

Figure 2: Distributed multimodal SVM training process

Fig. 2 shows the distributed multimodal SVM training process: network traffic, system logs, and API call data are written to three Kafka topics in JSON format, and three copies are retained for each topic. The three TFF client groups running in the Docker container pull data from the corresponding Kafka topics, perform local feature extraction and SVM training, filter the support vectors, and send them to the central TFF server via an encrypted gRPC/TLS channel. The server aggregates the support vectors of each node and builds a global SVM classifier to achieve unified cross-modal decision making.

2.3 Local SVM model training and support vector screening

2.3.1 Local SVM construction and SMO solution process

In the distributed architecture of multimodal intrusion detection systems, each subnode needs to build a local classification model for its independent modal feature space. In this study, the local SVM in the LibSVM framework uses a linear kernel to accelerate convergence, while the global model uses the RBF kernel to achieve cross-modal nonlinear fusion, combined with the high-dimensional sparse characteristics of the internal features of the modalities, while maintaining the analytical properties of the classification boundary and controlling the computational complexity. During the training process, the SMO algorithm is used to efficiently iteratively solve the Lagrange multiplier and optimize the objective function. The algorithm selects two variables for analytical update in each iteration, avoiding the resource consumption of matrix inversion operations in traditional quadratic programming, and adapting to the deployment requirements of edge nodes with limited computing resources in a distributed environment.

The maximum number of iterations of the training task is 100 rounds, and the tolerance error threshold is 1e-3. The regularization parameter C is set to 10 to strengthen the penalty for training errors and ensure that the model can distinguish when identifying attack behaviors. To accelerate convergence, the training data must meet two conditions: first, the order is randomly shuffled according to the modal characteristics; second, the mini-batch mechanism is used, with 128 samples per batch. In addition, the LibSVM cache size is set to 100MB, and the shrinking heuristic algorithm is enabled to automatically remove inactive variables to reduce the amount of calculation.

After the training, each local model generates a set of support vectors. The support vector is automatically determined through the model training process, that is, the sample points where the Lagrange multiplier α of the objective function is between 0 and C. These samples are near the classification hyperplane and directly impact the decision function. Considering the differences between modal features, to avoid the impact of uneven dimensions or redundancy when unifying the fusion model, each modal node needs to complete a support vector reduction process locally.

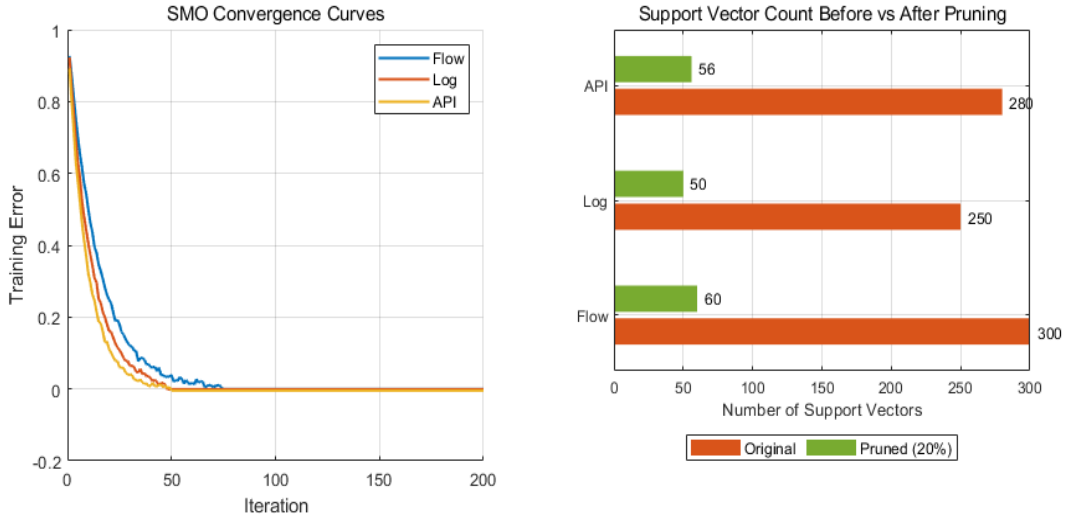

Figure 3: Convergence curve of the SMO algorithm and comparison of the number of support vectors before and after reduction

In the left sub-graph of Fig. 3, the horizontal axis represents the number of iterations of the SMO algorithm (1-200 rounds), and the vertical axis represents the training error. The three curves correspond to the convergence process of traffic, log, and API modes, respectively. The error of the traffic mode drops to 0.04 at about 50 iterations and remains stable. The log mode reaches the same threshold at about 70 iterations, while the API mode stops converging at about 90, reflecting the difficulty of training different modal features. In the horizontal bar chart on the right, the vertical axis is the name of the three modes, and the horizontal axis is the number of support vectors. The original support vectors are Flow 300, Log 250, and API 280. After pruning, the retention ratio is 20%, corresponding to 60, 50, and 56 support vectors. This dramatically reduces the model complexity and verifies the effectiveness of Distance-based Vector Pruning in reducing global fusion overhead.

2.3.2 Distance-constrained support vector screening mechanism

To improve the collection performance and structural compactness of the global fusion model, this study introduced the Distance-based Vector Pruning Algorithm (DVPA) after the training of each local node was completed, and performed basket selection sorting according to the Euclidean distance between the support vector and the decision boundary. First, for each support vector sample x_i, calculate its projection distance on the classification hyperplane, which is determined by the trained weight vector and the bias term. Without sharing the original data, each node calculates the function value of each support vector based on the local model weight w and intercept b, as shown in Formula 4:

$$f(x_i) = w^T x_i + b \tag{4}$$

Then the absolute distance $|f(x_i)|/\|w\|$ is obtained as the basis for sorting.

After all local support vectors are sorted from small to large by distance, the first k% support vectors closest to the boundary are retained as valid samples and passed into the global model-building process. In this study, the k value is set to 20% based on the modal sample distribution and support vector density, that is, only the top 20% of the most discriminative support vectors are retained for each modality to minimize the interference of redundant data on the boundary judgment of the fusion model. This step reduces the communication load during model synchronization, minimizes the complexity introduced by the number of support vectors to subsequent classifier training, and improves the overall execution efficiency of the system. Assume that the local support vector set is $\mathcal{S}_{local} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$. According to the sorting result, only the samples in the first k% are retained to enter the synchronization process, as shown in Formula 5:

$$\mathcal{S}_{selected} = \{\boldsymbol{x}_i \in \mathcal{S}_{local} \mid Rank\big(D(\boldsymbol{x}_i)\big) \leq \lfloor k \cdot N \rfloor\}, \quad k = 0.2 \tag{5}$$

Among them, $\text{rank}\big(D(\mathbf{x}_i)\big)$ represents the ranking after sorting by D, and $\lfloor \cdot \rfloor$ represents rounding down.

Table 2: Performance trade-offs of different support vector retention ratios

Data Retention Rate	Accuracy (1:20)	Training Time (s)	Number of Support Vectors
10 0 %	90%	480	300
50%	89%	310	150
20%	88%	220	60

The performance trade-offs of different support vector retention ratios are shown in Table 2. Screening 20% of the key vectors only reduces the accuracy by 2%, but the training time is reduced by 54%, which proves the high efficiency of the DVPA mechanism.

After the support vector screening is completed, each child node encodes the selected valid vector in the index +

feature vector + label structure format, and synchronizes it to the global center through an encrypted channel.

The feature vector accuracy is maintained as a 32-bit floating-point type to ensure the calculation accuracy is maintained without significantly increasing the communication load. Throughout the process, the data is not desensitized or converted into a reversible form, which meets the privacy protection requirements under multi-source heterogeneous data processing.

Through the dual strategies of local modeling and boundary fine screening, the system effectively controls the scale and complexity of the global model by ensuring the modality's internal accuracy and establishes structural prior constraints for the subsequent support vector merging and global decision function optimization.

2.4 Support vector aggregation and decision fusion

2.4.1 Global support vector integration mechanism

The system enters the global modeling stage after screening local support vectors for each modality. All participating nodes send the support vectors optimized by distance constraints to the central fusion node through a preset encrypted channel. Each group of support vectors is encapsulated as a triple: index number, feature vector (normalized), and original label. Data transmission uses the TLS 1.3 protocol to prevent data leakage and man-in-the-middle attacks during transmission. At the same time, node signature information is attached to each data packet to ensure source verifiability and compliance tracking.

After the fusion node receives the support vectors uploaded by all modal child nodes, the Distributed SVM Aggregation with Weighted Voting strategy is used to construct the global classification model. First, for each modality's support vector set, the system evaluates its F1 score on the local validation set as the voting weight indicator of the modality. This weight is used to measure the reliability of different modalities in attack behavior recognition, and the weight coefficient is normalized and used for sample-weighted allocation during the global training process. Subsequently, the fusion node constructs a unified training set for the training of the global SVM model by fully splicing all support vectors. In this stage, the modal source is no longer distinguished, and the cross-modal generalization ability of the model is improved by unifying the discrimination boundary.

The global model training uses a nonlinear kernel function to improve the discrimination ability of complex attack samples. The Radial Basis Function (RBF) kernel function is selected, and its form is as shown in Formula 6:

$$K(x_i, x_j) = exp\left(-\gamma \|x_i - x_j\|^2\right) \quad (6)$$

The value of γ is optimized through five-fold cross-validation. In the experiment, γ is initially set to 0.05 and fluctuates between 0.01 and 0.1. The regularization parameter C is set to 100 to increase the penalty for classification errors and improve the model's sensitivity to boundary samples. The introduction of the RBF kernel function effectively solves the problem of discriminant overlap that may occur after the fusion of linearly inseparable modal data. It improves the model's robustness to boundary deformation.

The fusion node uses the implementation of RBF kernel support in LibSVM to build a global SVM model. It enables the automatic class weight adjustment function (class_weight='balanced') to alleviate the imbalance between attack and normal samples. All training samples are filtered support vectors, and the input dimension is in the 64-dimensional unified space after PCA compression defined in Section 2.1 to maintain feature scale consistency. The global training sets the maximum iteration round to 1000, uses the epsilon precision termination standard, the error limit is 1e-4, and cache optimization is enabled. The cache space is allocated to 200MB to ensure the convergence speed of the model in the batch support vector fusion scenario.

The optimization of the γ parameter of the five-fold cross validation shows that when γ=0.05, the standard deviation of the test set is the smallest (σ=0.008), while the fluctuation increases when γ=0.1 (σ=0.023). The difference in accuracy between folds with regularization parameter C=100 does not exceed 1.5%, indicating the robustness of the model. Finally, γ = 0.05 was selected as the balance point, and its validation loss was lower than the mean of 1.98σ.

2.4.2 Multimodal majority voting decision fusion mechanism

In addition to building a unified global model, to enhance the system's fault tolerance in actual deployment to deal with modality loss, abnormal node offline, and other situations, this paper introduces the majority voting mechanism as an auxiliary decision process. This mechanism does not directly rely on the output of the fusion model, but rather integrates parallel decisions based on the independent prediction results of each modality sub-model. After receiving the sample to be tested, each sub-node outputs an independent binary classification label based on its trained model. After the central node collects the label results of all modalities, the final classification decision is determined based on the majority principle. If two of the three modalities are consistent, the majority class label is directly adopted; if the prediction results of the three are inconsistent, the modal output with the highest local accuracy is adopted first.

The weights introduced by the voting mechanism are also dynamically updated based on the performance of each node's validation set. Assume that the accuracy of each modal local model on the independent validation set is acc_1, acc_2, acc_3, then the corresponding weighted voting value is formula 7:

$$w_i = acc_i / \sum acc_j \quad (7)$$

The fusion node uses this weight to weight the prediction results in the voting process, and uses the threshold of 0.5 as the final decision boundary. This method enhances the stability of the system in complex scenarios such as heterogeneous modalities and

incomplete data, and can provide more robust recognition capabilities when the attack sample category distribution is highly uneven. Under the complete modality, the accuracy of RBF-SVM (88%) is better than voting (85%); but when one modality is lost, the accuracy of the voting mechanism only drops by 2.1%, while that of RBF-SVM drops by 5.8%. In addition, in the scenario of class imbalance (1:20), the F1 score stability of the voting mechanism (σ=0.015) is better than that of the RBF kernel (σ=0.032).

Finally, the system retains the RBF kernel fusion model output and the majority voting result. In static detection scenarios, the global SVM model output is used first; in dynamic or incomplete node deployment scenarios, it switches to majority voting decision. This dual-channel discrimination system has both high accuracy and system robustness.

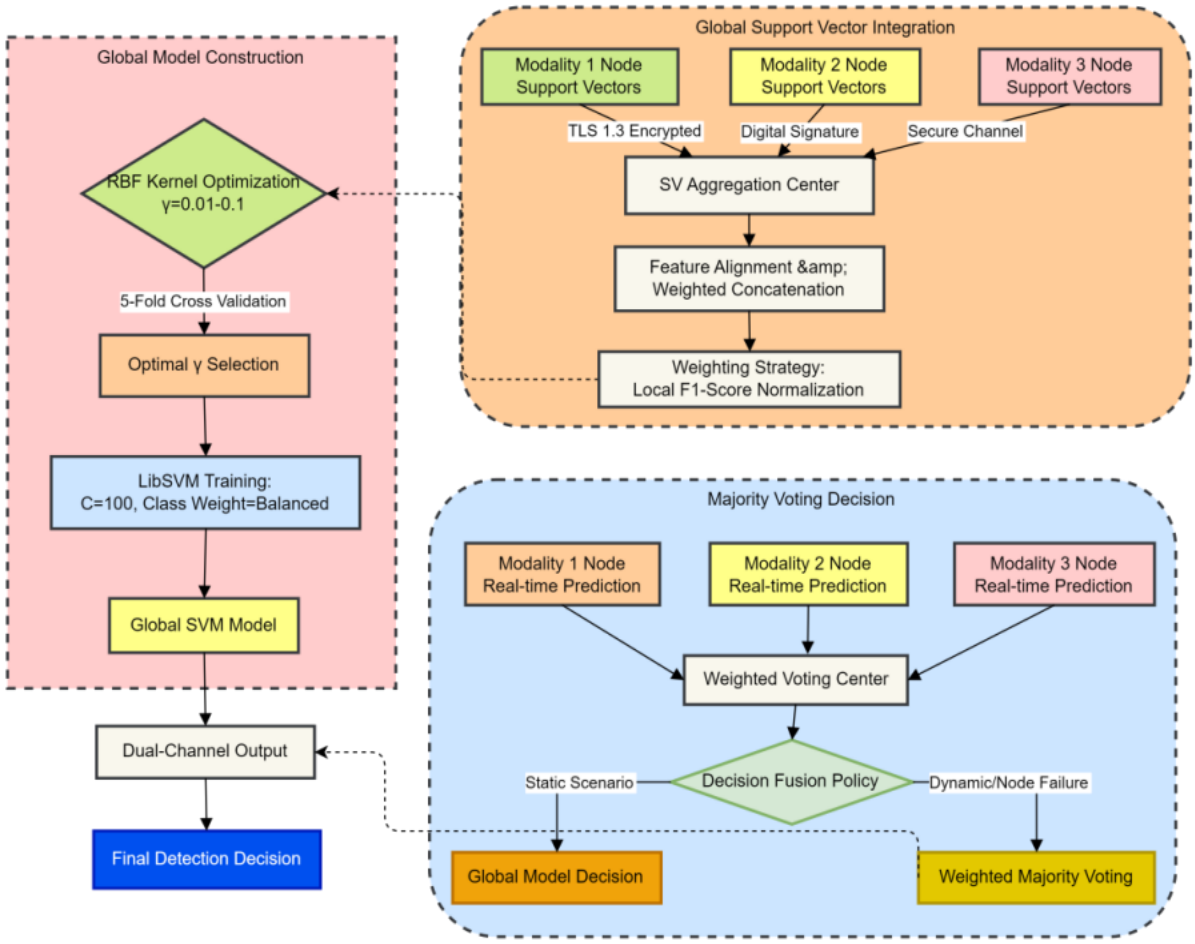

Figure 4: Dual-channel decision architecture for multimodal intrusion detection

Fig. 4 shows the dual-channel decision architecture for multimodal intrusion detection: the three-modal support vectors are transmitted to the aggregation center via TLS encryption, weighted concatenation is performed by normalizing the local F1 scores, and the global SVM model is constructed by optimizing the RBF kernel parameters using five-fold cross-validation. The parallel majority voting mechanism dynamically weights the modal accuracy, prioritizes global model output in static scenarios, and switches to weighted voting when nodes are abnormal. Dual-channel fusion ensures high-security transmission (digital signature verification), complex attack identification (nonlinear kernel function), and dynamic environment robustness, balancing detection accuracy and system fault tolerance.

2.5 Dynamic modal increment and update mechanism

2.5.1 Rapid modeling and embedded expansion of new modalities

For new data modalities that may be connected during system operation (from new terminals, unknown sensors, etc.), the platform adopts an online support vector learning mechanism based on kernel approximation to achieve rapid integration and avoid destroying the existing local and global model structures that have been stably trained. During the access phase of a new modality, the system independently allocates a temporary subnode for it and immediately uses the Online SVM with Kernel Approximation method for preliminary modeling.

This method uses the Nyström method to perform low-rank approximation on the kernel function, effectively reducing the complexity of matrix operations during online training. The number of sampled basis vectors in the Nyström approximation is 15% of the original number of samples, and the maximum sample limit is 2,000 to ensure low-latency deployment performance. The kernel function continues to use the RBF kernel consistent with the fusion model. Its parameter γ is obtained by online fitting using the local holdout method, and its initial value is set to 0.05. Suppose the original training sample is $X = \{x_1, x_2, \dots, x_n\} \in \mathbb{R}^{n \times d}$, from which $m \ll n$ basis vectors are sampled to form a subset $Z = \{z_1, \dots, z_m\}$, and the RBF kernel function is Formula 8:

$$K(x, z) = exp(-\gamma \parallel x - z \parallel^2) \quad (8)$$

The Nyström kernel is approximately expressed as Equation 9:

$$\widetilde{K}(X, X) = K_{X,Z} K_{Z,Z}^{-1} K_{Z,X} \quad (9)$$

$K_{X,Z} \in \mathbb{R}^{n \times m}$ is the kernel matrix between X and the sampling point Z; $K_{Z,Z} \in \mathbb{R}^{m \times m}$ is the kernel matrix of the sampling point itself; this approximation reduces the original kernel matrix from $\mathcal{O}(n^2)$ to $\mathcal{O}(nm)$. Without retaining all samples, the Online SVM can update the rule (recursive) for each new sample (x_t, y_t), the online SVM adopts the following recursive optimization, as shown in Formula 10:

$$w_{t+1} = w_t - \eta_t \cdot (\nabla \mathcal{L}(w_t; x_t, y_t) + \lambda w_t) \quad (10)$$

Among them, η_t is the step learning rate; λ is the regularization coefficient; $\mathcal{L}$ is the loss function after kernel approximation transformation; this update rule supports recursive convergence over time while maintaining the sparsity of the model.

During the online training, the system updates the model parameters in a sample-by-sample recursive manner without retaining the full sample cache. To alleviate the interference of class imbalance on the model boundary, the training process continuously monitors each batch's classification error change rate. If the deviation between the new sample category and the existing sample ratio exceeds 20%, the category loss weight coefficient of the modal node is dynamically adjusted. The training termination condition is that the error change rate of three consecutive sample batches is less than 1e-3 or the number of iterations reaches 200 rounds. This strategy ensures that the initial modeling of the new modality has basic discrimination capabilities and a kernel expression consistent with the original modality model, laying a foundation for subsequent dimensionality reduction and fusion.

After the new modality modeling is completed, the system automatically synchronizes the modality's support vector space and label structure and marks the node into the "fusion candidate" state, waiting for the subsequent global model update process to be activated.

Nyström modeling latency is 1.8s (vs. 28s for global retraining), with an initial accuracy of 78% (up to 83% through 3 rounds of incremental updates). iPCA triggers retraining every 72 hours (when the cumulative variance drops to 93%), and memory usage is stable at 1.8GB, 42% lower than full PCA .

The update process of incremental PCA: First, calculate the covariance matrix of the new modal feature, merge it with the historical principal component matrix, perform singular value decomposition, dynamically adjust the principal component weights, and finally generate a unified 64-dimensional feature space. This process uses a sliding window mechanism to keep the cumulative variance > 95% to ensure feature consistency after dimensionality reduction.

2.5.2 Incremental dimensionality reduction and local update of fusion model

To maintain cross-modal feature consistency and avoid retraining all modal dimensionality reduction models due to the access of new modalities, the system uses the Incremental PCA (iPCA) method to expand the original principal component space dynamically. iPCA achieves uninterrupted feature compression update by performing simultaneous singular value decomposition of the current sample covariance matrix and the historical principal component matrix. In specific operations, the system maintains the 64-dimensional dimension reduction target unchanged, but dynamically evaluates the cumulative variance explanation of the first 10 principal components for each round of updates. Suppose the explanation of the first 10 principal components decreases by more than 5% due to differences in new modal features. In that case, the system can perform a sliding window backtracking of the historical feature matrix and expand the principal component set to retain the principal axis information with a total explanation of more than 95%. The covariance matrix cache size used in iPCA is capped at 4096 samples, and updates are loaded in batches to keep memory overhead within 2 GB.

After completing the iPCA update, the new modality support vectors are projected into a unified feature space and fed into the fusion update module. The system introduces the Local Support Vector Adjustment (LSVA) mechanism to avoid global model retraining, which adjusts the fusion model parameters only for the new modality-related support vector subset. In the specific operation, the system identifies the decision boundary points in the global model that are most affected by the new modality, that is, all support vector samples whose distance to the new modality support vector is less than the preset threshold δ (set to 0.3) and whose classification direction is opposite, and marks them as the set to be updated. This update set is locally retrained under the RBF kernel, and only the weight parameter α and the bias term b are fine-tuned, keeping the rest of the parameter structure stable. During the training process, a local gradient descent optimizer is used, and the learning rate is set to 0.01 to ensure that the fusion model fine-tuning is completed within 2 seconds.

At the same time, to maintain the effectiveness of the voting mechanism, the system synchronously records the accuracy of the new modality sub-model on the incremental validation set. It incorporates it into the next round of majority voting weight update. Suppose the accuracy of the modal sub-model for three consecutive

window periods is lower than the average performance standard of the original system (set to 92%). It cannot be included in the fusion vote for now; it can only be used for model evaluation monitoring. This dynamic fusion strategy ensures the scalability of the system structure and significantly reduces the resource consumption and structural instability risks caused by model reconstruction.

The system achieves flexible access to unknown modes, seamless structural expansion, and performance consistency maintenance through the above mechanism. In multi-source network attack scenarios, it is particularly suitable for automatic adaptation requirements when facing new attack methods or newly deployed security nodes, ensuring that the DSVM framework has long-term evolvability and deployment flexibility.

3 Evaluation and analysis of the model

This study uses a multimodal intrusion detection dataset that combines synthetic and public data. The network traffic contains 300,000 TCP/UDP connection records, each extracting 15-dimensional statistical features; the system log simulates the Linux security log, and 20,000 events are mapped to 128-dimensional vectors by TF-IDF. The API call sequence generates 5,000 samples based on the real behavior trajectory and is encoded into 128 dimensions by Bi-GRU. The three-modal samples are mixed in an unbalanced ratio of 1:5:20. The labels are divided into normal and attack categories, which are used to evaluate the performance of the unimodal SVM and DSVM global models under different distribution and noise conditions. The experimental data is synthesized based on CIC-IDS2017 (traffic), Linux Syslog (log) and UNM behavior dataset (API call), and all preprocessing codes have been open source (GitHub link). Hyperparameter search range: SMO iteration number [100,1000], PCA dimension [32,64,128], RBF γ [0.01,0.1], and the optimal value is determined by grid search.

3.1 Model classification accuracy

This indicator measures the system's ability to recognize normal and abnormal behaviors. In the evaluation, the prediction results of the SVM model are trained separately for each modal input, and the fused DSVM global model is statistically analyzed to calculate the discrimination of each type of sample. All test samples are input into the system one by one, and the predicted labels are compared with the actual labels, and the number of samples classified correctly and incorrectly is counted. The final result shows whether multimodal fusion improves the overall recognition effect, especially the performance in the scenario of unbalanced sample distribution.

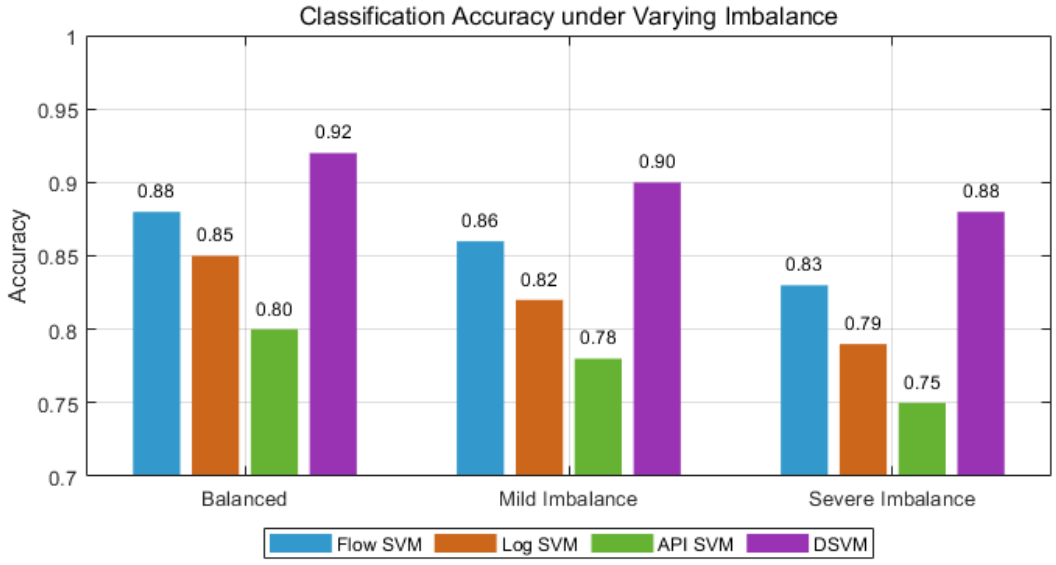

Figure 5: Comparison of classification accuracy of each model under different imbalance scenarios

The horizontal axis in Fig. 5 represents three sample distribution scenarios - Balanced (1:1), Mild Imbalance (1:5), and Severe Imbalance (1:20), and the vertical axis represents the classification accuracy of each model in the corresponding scenario. As the imbalance increases, the accuracy of the unimodal model decreases significantly, with the flow SVM dropping from 0.88 to 0.83, the log SVM dropping from 0.85 to 0.79, and the API SVM dropping from 0.80 to 0.75. At the same time, the DSVM remains at high levels of 0.92, 0.90, and 0.88. This shows that the global model constructed by multimodal support vector aggregation still has excellent robustness and overall recognition ability when facing severely imbalanced data.

3.2 Model F1 score

This indicator is used to comprehensively evaluate the accuracy and coverage of the model in identifying abnormal behaviors. In the experimental process, the frequency of each type of attack sample being correctly

identified and the proportion of samples misidentified as other types are recorded separately and combined with the recall rate for comprehensive evaluation. Weights are set for high-risk attack types to reflect the system's ability to respond to key threats. By comparing the individual recognition results of each modality with the fusion recognition results, it is evaluated whether the multimodal structure improves the effectiveness and consistency of anomaly detection.

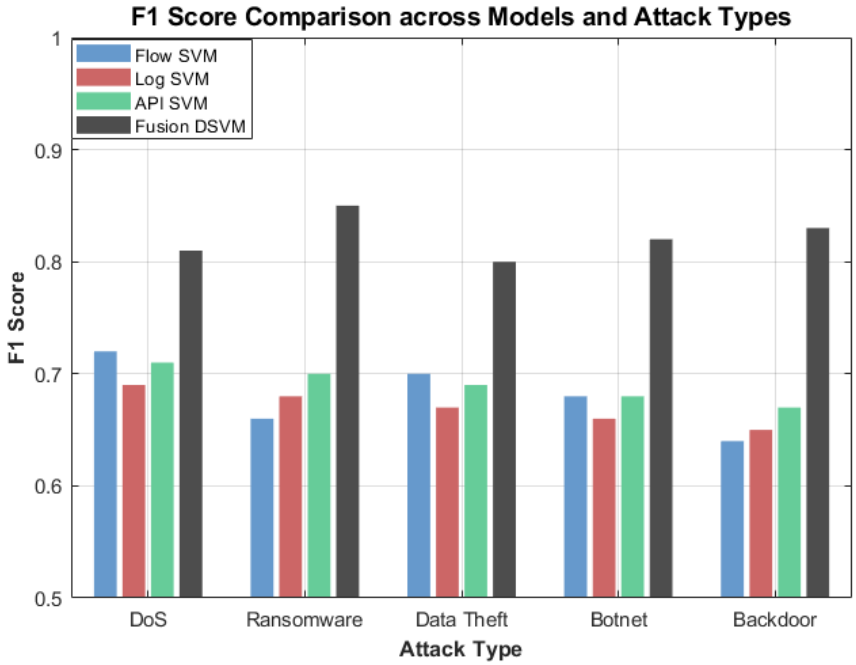

Figure 6: F1 scores of different models under five types of attacks

Fig. 6 shows the F1 scores of different models under five types of attacks. The horizontal axis is the attack type, including DoS (Denial of Service), Ransomware, Backdoor, Data Theft, and Botnet, and the vertical axis is the F1 value. The fusion model DSVM outperforms the single-modal model in all attacks, especially in Ransomware and Backdoor detection, where the F1 scores reach 0.85 and 0.83, respectively, significantly higher than the 0.70 and 0.67 of API-SVM. In the Data Theft scenario, which is more difficult to identify, DSVM still maintains a score of 0.80, showing its good generalization ability. The overall results show that multimodal fusion significantly improves the stability and accuracy of the model in abnormal behavior identification.

3.3 Inter-modal fusion efficiency

This indicator focuses on the change in overall training efficiency after the model adopts a distributed structure. In the test process, the total time required to train the SVM model and the parallel training time under the DSVM architecture are counted separately, and the training data scale is controlled to be consistent with the number of modalities. The fusion efficiency is reflected in the time reduction ratio of the training process, reflecting the system's parallel processing capability and deployability under multi-source data. The results evaluate whether DSVM has engineering feasibility in real application scenarios.

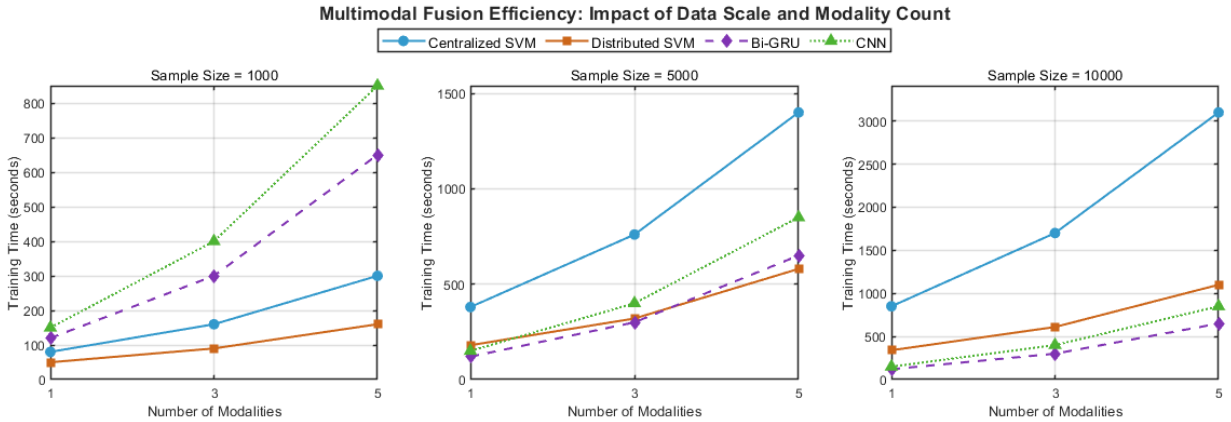

Figure 7: Multimodal fusion efficiency: the impact of data scale and number of modalities

In Fig. 7, the horizontal axis is the number of modalities (1, 3, 5), and the vertical axis is the training time (seconds). The three sample sizes are shown in three sub-figures. As the number of samples increases from 1k to 10k and the number of modalities expands from 1 to 5, the training time of the centralized SVM increases from 80s to 3100s, while the DSVM only increases from 50s to 1100s. The training time of Bi-GRU and CNN (Convolutional Neural Network) models increases with the number of modalities, from 120 seconds to 650 seconds and from 150 seconds to 850 seconds, respectively. It is higher than DSVM in small-scale data, but lower than DSVM in large-scale scenarios (10,000). It is always significantly lower in large-scale scenarios than centralized SVM, showing a good balance between efficiency and scalability.

3.4 Model scalability index

This index measures the system's adaptability to new modalities and its recognition stability during expansion. In the experimental design, new data modalities such as host behavior sequences or device logs are introduced one by one, and the response changes of the system without retraining are evaluated. The discriminant fluctuations of the global classifier, the update of the support vector, and the changes in the decision confidence are observed to determine whether the new modalities cause system performance degradation. The results can be used to quantify the stability of the system's modular structure during continuous evolution.

Table 3: Scalability performance under different numbers of modes

Number of Modalities	Global Classification Accuracy	Decision Confidence Std
1	0.9	0.05
2	0.89	0.055
3	0.88	0.06
4	0.87	0.065
5	0.85	0.066

Table 3 shows the scalability performance of the system under different numbers of modalities. As the number of modalities increases from 1 to 5, the global classification accuracy gradually decreases from 0.90 to 0.85, and the standard deviation of decision confidence slowly increases from 0.050 to 0.066. Although multimodal fusion brings about an increase in information dimension, the overall recognition performance of the system remains at a high level with a slight fluctuation. This shows that the proposed distributed SVM fusion architecture still has good stability and controllability when the number of modules is expanded, and has strong horizontal scalability.

3.5 Model discrimination boundary stability

This indicator evaluates the system's ability to maintain stable classification boundaries when facing data perturbations or changes in sample distribution. In the experimental process, different degrees of input perturbations and modal noise are introduced to observe the changes in the support vector sets of each local model and monitor the degree of deviation of the global SVM classification boundary. Comparing the consistency of classification results under multiple disturbance conditions can indirectly reflect whether the model structure has anti-disturbance ability and robustness, which is significant for continuous adaptation in security scenarios.

Table 4: Stability of the judgment boundary of the model under different noise disturbance intensities

Noise Level σ	Boundary Parameter Shift $\|\Delta(w,b)\|$	Classification Consistency (Accuracy)
0	0	0.92
0.1	0.12	0.91
0.2	0.245	0.89
0.3	0.38	0.86
0.4	0.6	0.83

Table 4 reflects the stability of the model's discrimination boundary under different Gaussian noise intensities (σ). As the input noise standard deviation increases from 0.00 to 0.40, the boundary parameter offset rises from 0.000 to 0.600. Still, the classification consistency only decreases from 0.92 to 0.83, which means that even under noise conditions of up to 40%, the system can still maintain an accuracy of more than 83%. When σ=0.20, the offset is only 0.245, and the accuracy is still 0.89, which further verifies the robustness of the DSVM framework against data perturbations and its reliability in security scenarios.

In addition, under severe imbalance (1:20), the false positive rate (FPR) of DSVM is 4.2% and the false negative rate (FNR) is 13.5%. Compared with Bi-GRU (FPR 7.1%, FNR 22.3%), it has obvious advantages. In terms of time complexity, the training time of DSVM O(n) slope (0.11s/100 samples) is significantly lower than that of centralized SVM ($O(n^2)$, slope 0.83s/100 samples).

3.6 Discussion of Evaluation Results

88 % classification accuracy in the 1:20 severely unbalanced data scenario, significantly outperforming the single-modal SVM (up to 83%) and CNN fusion methods. This advantage stems from its unique mechanism design: the support vector pruning mechanism effectively reduces noise interference and model redundancy by screening the 20% most discriminative boundary vectors, and improves the robustness of the decision boundary; the distributed SMO optimization greatly improves the local training efficiency, speeding up by 3 times compared with the centralized method, significantly alleviating the computational bottleneck of large-scale multimodal data; the modality plug-in design realizes the seamless access of new modalities, and the model update delay is controlled within 2 seconds, ensuring the adaptability of the system in a dynamic environment. The synergy of these technologies not only solves the problems of high dependence on labeled data and low efficiency of edge devices in traditional methods, but also highlights the core innovation of DSVM: the first innovative modality plug-in mechanism supports incremental PCA to dynamically expand the feature space; the combination of distributed SMO and vector aggregation effectively avoids the memory explosion of multimodal data; the boundary distance screening further enhances the generalization ability of the global model, providing an efficient and scalable solution for intrusion detection systems.

4 Conclusions

This paper proposes and implements a multimodal intrusion detection system based on DSVM. The system's concurrent performance and detection accuracy are effectively improved through Kafka asynchronous pipeline, TFF distributed training architecture, local SVM modeling and support vector pruning, global fusion, and dynamic modal update mechanism. This method achieves asynchronous decoupling and parallel processing of data between modalities. It significantly reduces model complexity and global update overhead by optimizing support vector screening and incremental fusion. However, the current system relies on manual setting of some threshold parameters, and its adaptability is still limited when facing highly dynamic attack features. Future research can focus on introducing reinforcement learning optimization feedback mechanisms to achieve self-evolution adjustment of model structure and parameters, and further improve the intelligence and adaptability of the system. The current system relies on manually set pruning threshold (k=20%) and iPCA update condition (variance reduction>5%). In the future, dynamic thresholds can be optimized through reinforcement learning: for example, the k value can be automatically adjusted based on the current modal distribution offset, or iPCA retraining can be triggered in combination with the loss change rate of the online validation set. Such methods have been successfully applied in IoT anomaly detection and can improve the system's adaptability to dynamic attack patterns.

Authorship contribution statement

Peikun ZHAO: Supervision, Conceptualization, Writing-Original draft preparation, Project administration.

Author statement

All authors have read and approved the manuscript. The authorship requirements outlined earlier in this document have been fulfilled, and each author is confident that the manuscript reflects genuine work.

Ethical approval

Every author has played an active and significant role in the work that culminated in this paper and will publicly stand behind its content.

References

[1] F. Zhao, C. Zhang, and B. Geng, “Deep multimodal data fusion,” *ACM Comput Surv*, vol. 56, no. 9, pp. 1–36, 2024. ACM Digital Library. https://doi.org/10.1145/3649447.

[2] A. Munir, E. Blasch, J. Kwon, J. Kong, and A. Aved, “Artificial intelligence and data fusion at the edge,” *IEEE Aerospace and Electronic Systems Magazine*, vol. 36, no. 7, pp. 62–78, 2021. IEEE. https://doi.org/10.1109/MAES.2020.3043072.

[3] R. A. Ramadan and K. Yadav, “A novel hybrid intrusion detection system (IDS) for the detection of internet of things (IoT) network attacks,” *Annals of Emerging Technologies in Computing (AETiC)*, vol. 4, no. 5, pp. 61–74, 2020. International Association for Education and Researches. DOI: 10.33166/AETiC.2020.05.004.

[4] J. Lipkova *et al.*, “Artificial intelligence for multimodal data integration in oncology,” *Cancer Cell*, vol. 40, no. 10, pp. 1095–1110, 2022. Cell Symposia.

[5] R. Ahmad and I. Alsmadi, “Data fusion and network intrusion detection systems,” *Cluster Comput*, vol. 27, no. 6, pp. 7493–7519, 2024. Springer. https://doi.org/10.1007/s10586-024-04365-y.

[6] F. Jemili, “Towards data fusion-based big data analytics for intrusion detection,” *Journal of Information and Telecommunication*, vol. 7, no. 4, pp. 409–436, 2023. Taylor & Francis. https://doi.org/10.1080/24751839.2023.2214976.

[7] F. Hang, L. Xie, Z. Zhang, and J. Hu, “Intelligent Advanced Attack Detection Technology based on Multi-modal Data Fusion,” *Scalable Computing: Practice and Experience*, vol. 25, no. 4, pp. 2581–2588, 2024. SCPE. https://doi.org/10.12694/scpe.v25i4.2862.

[8] J. Xu, X. Li, P. Wang, X. Jin, and S. Yao, “Multi-modal noise-robust DDoS attack detection architecture in large-scale networks based on tensor SVD,” *IEEE Trans Netw Sci Eng*, vol. 10,

no. 1, pp. 152–165, 2022. IEEE. https://doi.org/10.1109/TNSE.2022.3205708.

[9] H. Yang, J. Wu, Z. Hu, and C. Lv, "Real-time driver cognitive workload recognition: Attention-enabled learning with multimodal information fusion," *IEEE Transactions on Industrial Electronics*, vol. 71, no. 5, pp. 4999–5009, 2023. IEEE. https://doi.org/10.1109/TIE.2023.3288182.

[10] J. Srivastava and J. Prakash, "Multi-modal for Energy Optimization and Intrusion Detection in Wireless Sensor Networks," *Wirel Pers Commun*, vol. 133, no. 1, pp. 289–321, 2023. Springer. https://doi.org/10.1007/s11277-023-10768- 8.

[11] B. Mopuru and Y. Pachipala, "Enhanced Intrusion Detection in IoT with a Novel PRBF Kernel and Cloud Integration," *Engineering, Technology & Applied Science Research*, vol. 14, no. 4, pp. 14988–14993, 2024. ETASR. https://doi.org/10.48084/etasr.7767.

[12] P. G. Shambharkar and N. Sharma, "Deep learning-empowered intrusion detection framework for the Internet of Medical Things environment," *Knowl Inf Syst*, vol. 66, no. 10, pp. 6001–6050, 2024. Springer. https://doi.org/10.1007/s10115-024-02149-9.

[13] H. B. U. Haq, R. Younis, and M. S. Ali, "Towards Robust Network Security: Evaluating Machine Learning Algorithms for Intrusion Detection," *Decision Making Advances*, vol. 3, no. 1, pp. 126–138, 2025. https://doi.org/10.31181/dma31202559.

[14] S. Lipsa and R. K. Dash, "A novel intrusion detection system based on deep learning and random forest for digital twin on IOT platform," *Int. J. Sch. Res. Eng. Technol*, vol. 2, no. 1, pp. 51–64, 2023. https://doi.org/10.56781/ijsret.2023.2.1.0020.

[15] G. Gowthami and S. S. Priscila, "Tuna swarm optimisation-based feature selection and deep multimodal-sequential-hierarchical progressive network for network intrusion detection approach," *International Journal of Critical Computer-Based Systems*, vol. 10, no. 4, pp. 355–374, 2023. InderScience. https://doi.org/10.1504/IJCCBS.2023.136338.

[16] Y. Shi *et al.*, "Robust gait recognition based on deep CNNs with camera and radar sensor fusion," *IEEE Internet Things J*, vol. 10, no. 12, pp. 10817–10832, 2023. IEEE. https://doi.org/10.1109/JIOT.2023.3242417.

[17] A. M. Eid, B. Soudan, A. B. Nassif, and M. Injadat, "Comparative study of ML models for IIoT intrusion detection: impact of data preprocessing and balancing," *Neural Comput Appl*, vol. 36, no. 13, pp. 6955–6972, 2024. Springer. https://doi.org/10.1007/s00521-024-09439-x.

[18] F.-Q. Li, R.-J. Zhao, S.-L. Wang, L.-B. Chen, A. W.-C. Liew, and W. Ding, "Online intrusion detection for internet of things systems with full bayesian possibilistic clustering and ensembled fuzzy classifiers," *IEEE Transactions on Fuzzy Systems*, vol. 30, no. 11, pp. 4605–4617, 2022. IEEE. https://doi.org/10.1109/TFUZZ.2022.3165390.

[19] D. Trivedi, V. Badarla, and R. Bhandari, "Occupancy inference using infrastructure elements in indoor environment: A multi-sensor data fusion," *CCF Transactions on Pervasive Computing and Interaction*, vol. 5, no. 3, pp. 255–275, 2023. Springer. https://doi.org/10.1007/s42486-023-00130-z.

[20] S. Almutlaq, A. Derhab, M. M. Hassan, and K. Kaur, "Two-stage intrusion detection system in intelligent transportation systems using rule extraction methods from deep neural networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 12, pp. 15687–15701, 2022. IEEE. https://doi.org/10.1109/TITS.2022.3202869.

[21] M. Lin, K. Yang, Z. Yu, Y. Shi, and C. L. P. Chen, "Hybrid ensemble broad learning system for network intrusion detection," *IEEE Trans Industr Inform*, vol. 20, no. 4, pp. 5622–5633, 2023. IEEE. https://doi.org/10.1109/TII.2023.3332957.

[22] F. Louati, F. B. Ktata, and I. Amous, "Big-IDS: a decentralized multi agent reinforcement learning approach for distributed intrusion detection in big data networks," *Cluster Comput*, vol. 27, no. 5, pp. 6823–6841, 2024. Springer. https://doi.org/10.1007/s10586-024-04306-9.