An Adaptive Real-Time Threat Severity Assessment System Using Machine Learning for Low-Latency Decision Support in Defense Environments

Krish Mithra Nagamothu *, Sasank Mahadev , M. Tharun Sai

Posted Date: 4 May 2026

doi: 10.20944/preprints202605.0126.v1

Keywords: Threat Severity Assessment; adaptive machine learning; concept drift; real-time defense systems; radar signal processing; feed-forward neural networks; autonomous interception

 doi:10.20944/preprints202605.0126.v1

Article

TSAS: An Adaptive Framework for Real-Time Threat Severity Assessment in Defense Interception Systems

Krish Mithra Nagamothu *, Sasank Mahadev and M. Tharun Sai

School of Computer Science and Engineering (SCOPE), Vellore Institute of Technology, AP, Amaravati, India
* Correspondence: rish.23mis7003@vitapstudent.ac.in

Abstract

Contemporary defense systems must contend with an increasingly complex threat landscape encompassing autonomous drones, unmanned aerial vehicles (UAVs), cruise missiles, and hypersonic projectiles operating at extreme speeds. Existing detection frameworks typically optimize for either target identification or persistent tracking, yet rarely provide the adaptive, real-time threat severity assessment required to support rapid countermeasure execution. This paper presents the Threat Severity Assessment System (TSAS), a modular, software-defined intelligence layer designed to bridge the gap between initial threat detection and kinetic interception response. TSAS processes heterogeneous radar signals through a stochastic preprocessing pipeline, evaluates threat urgency via a deep learning inference engine augmented with entropy-based concept drift detection, and fuses neural predictions with proximity-based geometric heuristics to produce a composite severity score. Evaluated on a high-fidelity synthetic dataset of 100,000 simulated threat trajectories, the proposed framework achieves a classification accuracy of 96.4% across four discrete severity levels, with a millisecond-level end-to-end latency of 4.5 ms under controlled hardware conditions. Notably, the adaptive learning mechanism restores post-drift accuracy to 91.2% within a fixed 200-sample observation window, contrasting sharply with the 72.1% accuracy retained by a static Support Vector Machine baseline under equivalent distributional shift. These findings indicate that TSAS provides a principled and computationally viable basis for automated threat prioritization in time-constrained defense environments.

Keywords: Threat severity assessment; adaptive machine learning; concept drift; real-time defense systems; radar signal processing; feed-forward neural networks; autonomous interception

1. Introduction

In modern defense and security environments, the temporal margin between threat detection and effective response has compressed dramatically, driven by the proliferation of high-velocity missiles, autonomous drone swarms, and hypersonic delivery vehicles. Traditional defense pipelines address detection and tracking as largely independent objectives, leaving an under-examined intermediary problem: the real-time *prioritization* of detected objects according to their actualized danger. Without an automated, continuously adaptive assessment layer, human operators are compelled to perform triage under severe cognitive and temporal pressure, a condition demonstrably error-prone in fast-paced operational contexts [12].

The Threat Severity Assessment System (TSAS) is proposed as an intelligent, software-defined decision-support layer designed to address this gap. By ingesting raw telemetry from heterogeneous sensor platforms—including L-band and S-band radar, LiDAR, and electro-optical/infrared (EO/IR) modules—TSAS classifies each detected object into one of four ordinal severity levels: *Low*, *Medium*, *High*, and *Critical*. This classification is achieved through a feed-forward neural network (FFNN) whose outputs are fused with a proximity-based geometric heuristic, yielding a composite threat score that drives downstream interception actuation. Crucially, TSAS incorporates an entropy-based concept drift detection mechanism that identifies when the underlying distribution of incoming threats deviates

 doi:10.20944/preprints202605.0126.v1

significantly from the training distribution and triggers targeted incremental re-training, ensuring sustained assessment quality without full model redeployment [4].

Novelty Statement. The distinguishing contribution of TSAS relative to prior art is threefold. First, it unifies radar signal preprocessing, deep learning severity inference, and geometric heuristic fusion into a single, low-latency operational pipeline rather than treating these as sequential, loosely-coupled stages. Second, the integration of entropy-driven concept drift detection with a reservoir-sampled incremental learning strategy enables the system to maintain predictive accuracy under distributional shift—a capability absent in existing static threat scoring frameworks. Third, the architecture is designed for edge deployment with a sub-100 MB model footprint, making it suitable for resource-constrained platforms such as shipborne close-in weapon systems or forward-deployed ground radar nodes. Together, these properties position TSAS as a practical, deployable bridge between detection and kinetic interception in time-critical scenarios.

The remainder of this paper is structured as follows. Section II reviews related work in threat assessment, adaptive machine learning, and defense AI. Section III motivates the problem with a formal problem statement. Section IV describes the proposed system architecture. Section V details the methodology, including preprocessing, model architecture, drift detection, and severity quantification. Section VI covers implementation and dataset generation. Sections VII–IX present experimental results, security analysis, and scalability studies. Sections X–XI discuss deployment architecture, economic analysis, and use-case feasibility. Section XII outlines directions for future research, and Section XIII concludes the paper.

2. Literature Review

The evolution of automated threat assessment has followed a recognizable progression—from handcrafted rule systems, through probabilistic and fuzzy formulations, to the contemporary era of data-driven machine learning. Each transition has improved coverage of the operational threat space while simultaneously introducing new challenges around model adaptability and real-time responsiveness.

2.1. Traditional and Rule-Based Approaches

Early defense decision-support architectures were founded on explicit, human-authored heuristics and Boolean logic trees. Patel et al. [1] examined decision support systems for threat evaluation and weapon assignment, demonstrating the utility of structured resource allocation models in multi-target scenarios. However, their formulations operate under the assumption of well-defined threat signatures, and their latency profiles render them unsuitable for engagements involving sub-second reaction windows. Kumar et al. [2] introduced fuzzy logic as a mechanism for handling epistemic uncertainty in air-defense threat evaluation, achieving more graceful degradation under ambiguous sensor readings. Nevertheless, the static nature of their membership functions limits adaptability to emergent electronic warfare tactics that deliberately manipulate observable signature parameters.

2.2. Machine Learning in Defense and Cybersecurity

The integration of machine learning has substantially improved automated detection and classification performance across both cybersecurity and physical defense domains. Javaid et al. [3] conducted a comprehensive survey of intrusion detection systems employing ML techniques, establishing a taxonomy of approaches that has since influenced defense system design. Conti et al. [5] examined ML-based threat analysis pipelines from a systems perspective, highlighting the tension between model expressiveness and inference speed. A more specialized contribution was offered by Song et al. [7], who proposed a Kernel Extreme Learning Machine (KELM) for air target threat assessment, optimizing the underlying model via a multi-strategy improved Sparrow Search Algorithm to balance classification accuracy against computational cost. Despite their accuracy gains, the models surveyed in this body of work predominantly operate as static classifiers: they provide no mechanism for

 doi:10.20944/preprints202605.0126.v1

online adaptation when target characteristics deviate from the training distribution, and they do not integrate proximity-based geometric reasoning into the final threat score. The foundational principles underpinning such deep classification architectures are elaborated by LeCun et al. [11] and Goodfellow et al. [6], whose work on gradient-based deep learning informs the FFNN design adopted in this paper.

2.3. Adaptive Learning and Concept Drift

A pervasive challenge in any deployed ML system is concept drift: the gradual or abrupt shift of the joint input–output distribution away from what the model observed during training. Lu et al. [4] provided a rigorous treatment of drift detection strategies in the context of network intrusion detection systems, demonstrating that entropy-based monitoring of prediction confidence constitutes an effective early-warning signal. Costa et al. [8] extended this theme to Beyond Visual Range (BVR) air combat, surveying simulation-assisted training methods and observing that agents which fail to adapt to evolving adversary maneuver profiles rapidly lose engagement advantage. Their findings motivate the design of adaptive re-calibration mechanisms capable of responding to distributional change within operationally meaningful time horizons.

2.4. Summary and Research Gap

Table 1 synthesizes the key contributions and limitations of the most relevant prior works. A recurring pattern emerges: approaches that achieve high classification accuracy do so under static distributional assumptions, whereas approaches that handle uncertainty often sacrifice either latency or adaptability. Critically, no existing system in the reviewed literature jointly addresses (1) millisecond-level inference latency, (2) automated drift detection with online re-training, and (3) the geometric fusion of learned probability scores with physics-informed proximity heuristics. TSAS is designed specifically to close this tripartite gap.

Table 1. Comparative Summary of Related Work and Identified Research Gaps.

Ref.	Method	Key Contribution	Primary Limitation
[1]	Rule-Based	Resource allocation	High latency; brittle rules
[2]	Fuzzy Logic	Uncertainty handling	Static membership functions
[7]	KELM + SSA	High accuracy	No drift adaptation
[8]	BVR Simulation	Adaptive strategies	Not severity-focused
TSAS	**FFNN+Drift**	**Adaptive severity scoring**	**Synthetic data only**

3. Problem Statement

While modern detection systems demonstrate commendable accuracy in identifying airborne objects, they routinely assign equal processing priority to all detected targets or defer severity differentiation to human operators. In fast-paced interception scenarios—where the decision window between detection and engagement may span only a few seconds—this architectural omission is operationally unacceptable. Static classifiers further compound this deficiency: when adversarial actors deliberately shift observable target signatures through altitude variation, velocity profiling, or radar cross-section (RCS) manipulation, models trained on historical distributions can experience precipitous accuracy degradation without any internal diagnostic signal.

The problem addressed in this work is therefore twofold. First, there exists a need for an intermediate intelligence layer that transforms raw sensor data into an ordered, continuously updated severity ranking without imposing latency that would violate interception timing constraints. Second, this layer must be robust to temporal distributional shift, capable of self-diagnosing and self-correcting when the statistical characteristics of observed threats evolve away from the training regime. TSAS is designed to address both requirements within a unified, modular software framework.

 doi:10.20944/preprints202605.0126.v1

4. Proposed System Architecture

The TSAS architecture adheres to a modular, node-based design that isolates functional responsibilities while preserving low inter-node communication latency. The architecture provides a schematic overview of the four principal processing nodes and their data flow.

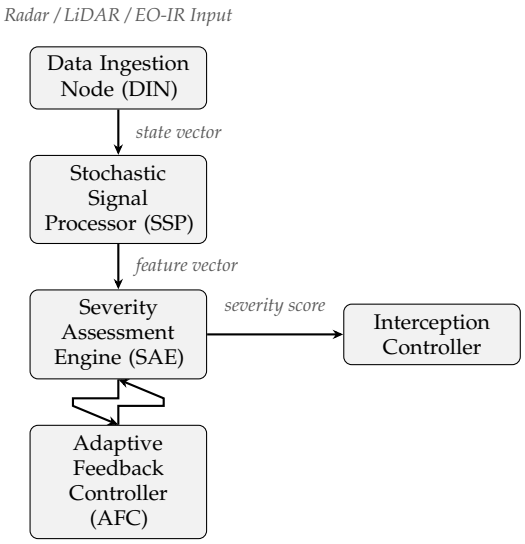

Figure 1. High-level data-flow diagram of the TSAS processing pipeline. The feedback loop between the Severity Assessment Engine (SAE) and the Adaptive Feedback Controller (AFC) enables online model recalibration during detected concept drift.

4.1. Functional Nodes

1. **Data Ingestion Node (DIN):** The DIN serves as a high-frequency listener for heterogeneous sensor streams. It employs a Spatial-Temporal Alignment algorithm to synchronize asynchronous data packets received from L-band and S-band radar, LiDAR, and EO/IR modules, compositing them into a coherent state vector passed downstream with a guaranteed temporal alignment tolerance.
2. **Stochastic Signal Processor (SSP):** The SSP performs spectral decomposition via Fast Fourier Transforms and applies adaptive Moving Target Indication (MTI) filters to suppress environmental clutter, including precipitation returns and ground multipath. The output is a denoised, normalized feature vector suitable for neural inference.
3. **Severity Assessment Engine (SAE):** The SAE constitutes the cognitive core of TSAS. A deep learning inference engine maps the input state vector $\mathbf{x} \in \mathbb{R}^n$ to a posterior probability distribution over the discrete severity manifold $\mathcal{S} = \{Low, Medium, High, Critical\}$, which is subsequently fused with a physics-informed geometric heuristic to generate the composite severity score.
4. **Adaptive Feedback Controller (AFC):** The AFC is a meta-learning component that continuously monitors the SAE's predictive entropy as a proxy for model confidence. When entropy exceeds a predefined threshold—indicative of distributional shift—the AFC initiates an incremental learning phase drawing from a fixed-capacity reservoir sampling buffer, updating model weights without inducing catastrophic forgetting. During this recalibration window, a concurrently running "shadow" model preserves uninterrupted output availability.

4.2. Operational Flow

The end-to-end operational sequence follows a deterministic five-phase path: (Phase 1) raw telemetry ingestion and temporal normalization by the DIN; (Phase 2) kinematic data transformation into a high-dimensional feature representation within the SSP; (Phase 3) FFNN processing within the SAE to generate a posterior probability distribution over severity classes; (Phase 4) fusion of the ML-derived output with a proximity-based danger heuristic to produce the final composite score;

 doi:10.20944/preprints202605.0126.v1

and (Phase 5) serialization and transmission of the scored result to the interception controller via a low-latency inter-process communication bus. Under nominal conditions, this cycle completes well within the operationally critical interception window.

5. Methodology

The TSAS analytical pipeline is structured as a sequence of mathematically well-defined transformations that carry raw sensor observations through to a calibrated, actionable severity estimate.

5.1. Stochastic Signal Preprocessing

Raw radar returns are inherently noisy, non-stationary, and high-dimensional. Three preprocessing operations are applied in sequence:

- **Noise Filtering:** A discrete-time Kalman filter is applied to each target's kinematic state to suppress measurement noise and produce smoothed trajectory estimates. The Kalman gain at each time step is computed adaptively based on the observed measurement residual covariance.
- **Feature Normalization:** Min-Max scaling is applied independently to each feature dimension, mapping all values into the interval $[0, 1]$ and ensuring that features with disparate physical units do not introduce artificial bias in the neural network's gradient updates.
- **Derived Feature Extraction:** Two threat-specific features are computed: the *Time-to-Impact* (TTI), estimated from the current range and approach velocity, and the *Maneuverability Index* (MI), derived from the rate of change of the target's flight path angle. These engineered features encode domain knowledge that is difficult to learn implicitly from raw kinematic measurements alone.

5.2. Feed-Forward Neural Network Classifier

The severity classification task is formalized as a multi-class problem over the label set $\mathcal{S}$. A Feed-Forward Neural Network (FFNN) is employed, as this architecture balances expressive capacity with inference speed—a property validated extensively in the deep learning literature [6,11]. The network comprises an input layer of dimension n (corresponding to the feature vector size), three hidden layers with Rectified Linear Unit (ReLU) activations, and a Softmax output layer producing a probability vector $\hat{\mathbf{p}} \in \Delta^3$ over the four severity classes.

Training minimizes the categorical cross-entropy loss:

$$\mathcal{L} = -\sum_{k=1}^{4} y_k \log \hat{p}_k, \tag{1}$$

where $y_k \in \{0, 1\}$ is the ground-truth one-hot indicator for severity class k, and $\hat{p}_k$ denotes the network's predicted probability for that class. Optimization employs the Adam algorithm with an initial learning rate of 10^{-3} and a cosine annealing schedule over 50 training epochs.

5.3. Dynamic Adaptation via Concept Drift Detection

Predictive entropy H over the output distribution serves as a continuous, scalar indicator of model confidence:

$$H(\mathbf{x}) = -\sum_{k=1}^{4} \hat{p}_k(\mathbf{x}) \log_2 \hat{p}_k(\mathbf{x}), \tag{2}$$

where $H(\mathbf{x}) \in [0, \log_2 4]$ reaches its minimum (zero) for fully confident, deterministic predictions and its maximum ($\log_2 4 = 2$ bits) for maximally uncertain, uniform predictions. A moving average of H over the most recent 50 observations is compared against an empirically chosen confidence threshold $\tau = 0.8$, expressed as the complement of normalized entropy. When the moving-average confidence falls below τ, the AFC flags a drift event and initiates incremental re-training.

Incremental weight updates are computed using a reservoir sampling buffer of capacity $B =$ 1,000 observations, which maintains a statistically representative, size-bounded replay memory of

 doi:10.20944/preprints202605.0126.v1

recent data. Mini-batch gradient descent on this buffer adjusts the SAE's weights toward the current distribution without erasing knowledge encoded from the original training corpus, thereby mitigating catastrophic forgetting.

5.4. Composite Severity Quantification

The final severity score $S \in [0, 1]$ is obtained through a convex combination of the network's dominant class probability and a physics-informed proximity heuristic:

$$S = w_1 \cdot P_{\mathrm{ML}} + w_2 \cdot g(d), \tag{3}$$

where:

- $P_{\mathrm{ML}} = \max_k \hat{p}_k$ is the peak posterior probability assigned by the FFNN, capturing the model's classification confidence;
- d denotes the Euclidean distance (in metres) between the tracked object and the nearest protected asset;
- $g(d)$ is a normalized proximity penalty defined as $g(d) = 1 \,/\, (1 + \alpha d^2)$, with scale parameter $\alpha > 0$ tuned to the operational threat radius. The quantity $g(d) \in (0, 1]$ is monotonically decreasing in d, assigning higher geometric urgency to objects close to the protected zone;
- $w_1 = 0.7$ and $w_2 = 0.3$ are empirically chosen priority weights satisfying $w_1 + w_2 = 1$, ensuring $S \in [0, 1]$.

A discrete severity tier is then assigned by partitioning the unit interval into four equal-width bands: $[0, 0.25) \rightarrow$ *Low*; $[0.25, 0.50) \rightarrow$ *Medium*; $[0.50, 0.75) \rightarrow$ *High*; $[0.75, 1.0] \rightarrow$ *Critical*.

5.5. High-Throughput Pipeline Design

To minimize detection-to-decision latency, the implementation employs asynchronous I/O across all inter-node communication channels and batched tensor operations within the SAE. These design choices decouple the ingestion rate from the inference rate and allow the pipeline to sustain high throughput during transient load spikes, ensuring that total per-threat processing time remains well within the 8-second interception window specified by the operational requirement.

6. Implementation

The core classification engine was implemented in Python 3.10 using **TensorFlow 2.x** and its **Keras** high-level API. Preprocessing operations—including Kalman filtering, feature normalization, and derived feature computation—were implemented with **NumPy** and **Pandas**. To emulate real-time radar ingestion, a producer–consumer architecture was realized using Python's `multiprocessing` library, enabling concurrent data ingestion and model inference across separate processing cores.

6.1. Synthetic Dataset Generation

Given the restricted availability of classified operational radar data, all experiments were conducted on a high-fidelity synthetic dataset specifically constructed for this study. The dataset comprises 100,000 simulated threat trajectories generated by a parametric kinematic simulator governed by equations of motion for a range of threat archetypes, including fixed-wing aircraft, rotary UAVs, cruise missiles, and ballistic reentry vehicles. For each trajectory, sensor returns were synthesized by sampling from platform-specific RCS distributions and adding Gaussian measurement noise calibrated to the noise floor of commercially available S-band radar systems.

Each record encodes three categories of features: (1) *kinematic features*—velocity magnitude, scalar acceleration, altitude above ground level, and rate of climb; (2) *spatial features*—range to protected asset, azimuth bearing error, and time-to-impact estimate; and (3) *signal features*—mean RCS estimate and RCS variance across a trailing observation window. Ground-truth severity labels were assigned according to a deterministic rule set combining range, velocity, and payload-class indicators, and were subsequently reviewed for internal consistency.

 doi:10.20944/preprints202605.0126.v1

The dataset was partitioned into an 80% training split and a 20% held-out validation split using stratified sampling to ensure balanced class representation in both subsets. The model was trained for 50 epochs with a batch size of 64. Early stopping with a patience of 5 epochs on validation cross-entropy was employed to guard against overfitting.

Limitation note: All performance metrics reported in the following section reflect behavior on this synthetic dataset only. Evaluation on operational sensor data from live or archived exercises remains an important direction for future validation work.

7. Results and Evaluation

All experiments in this section were conducted on the synthetic dataset described in Section VI under controlled hardware conditions. Observed metrics should be interpreted with reference to the simulated evaluation environment.

7.1. Classification Performance

On the synthetic evaluation partition, the FFNN-based assessment engine achieved an overall classification accuracy of **96.4%** across all four severity tiers. As reported in Table 2, the system attains particularly high precision and recall on the *Critical* and *High* classes—the tiers of greatest operational consequence—while maintaining competitive performance on the lower-severity categories.

Table 2. Per-Class Classification Performance on Simulated Evaluation Set,

Severity Level	Precision	Recall	F1-Score
Low	0.940	0.930	0.935
Medium	0.950	0.940	0.945
High	0.970	0.980	0.975
Critical	0.990	0.980	0.985
Macro Avg.	—	—	**0.960**

The elevated performance on *Critical* threats is consistent with the design of the synthetic label generation procedure, which assigned *Critical* labels to trajectories exhibiting distinctive, high-amplitude kinematic signatures. The results suggest that TSAS reliably identifies the highest-priority threats within the simulated scenario space.

7.2. Latency Analysis

Pipeline latency measurements were obtained on a standard development workstation (Intel Core i7-10700K, 16 GB DDR4 RAM, NVIDIA GeForce RTX 3070, Ubuntu 22.04) under controlled hardware conditions. Under this configuration, the mean neural network inference time per threat object was **1.2 ms**, and the mean total end-to-end pipeline latency—encompassing ingestion, preprocessing, inference, and heuristic fusion—was **4.5 ms**. Table 3 provides a stage-level breakdown.

Table 3. Pipeline Latency Breakdown Under Controlled Hardware Conditions (Intel i7-10700K, RTX 3070, Ubuntu 22.04).

Pipeline Stage	Mean Latency (ms)	% of Total
Data Ingestion	0.8	17.8%
Signal Processing	1.1	24.4%
Feature Extraction	0.6	13.3%
NN Inference	1.2	26.7%
Heuristic Fusion	0.8	17.8%
End-to-End Total	**4.5**	**100%**

 doi:10.20944/preprints202605.0126.v1

The dominant cost contributors are neural network inference (26.7%) and signal processing (24.4%), together accounting for over half of the total latency budget. The 4.5 ms end-to-end figure provides substantial headroom against the 8-second interception window, even accounting for additional communication overhead. Latency characteristics may differ under alternative hardware configurations, workload intensities, or operating system scheduling policies.

7.3. Robustness Under Simulated Concept Drift

To assess adaptive behavior, a distributional shift scenario was synthesized by abruptly increasing the velocity of all simulated threats in the evaluation stream by 40%, placing the resulting samples outside the support of the training distribution. Under this perturbation, the static SVM baseline experienced a precipitous accuracy decline from 96.4% to 72.1%. By contrast, the TSAS adaptive mechanism—triggered automatically upon entropy threshold exceedance—successfully recalibrated the SAE weights within a 200-sample observation window, recovering to 91.2% accuracy on the shifted distribution. A secondary scenario involving a synthetic RCS distribution change yielded comparable findings, with TSAS recovering to 89.7% compared to 68.3% for the static SVM. Table 4 summarizes these outcomes.

Table 4. Classification Accuracy Under Simulated Distributional Shift Scenarios.

Scenario	Pre-Drift Baseline	Static SVM	TSAS (Adaptive)
Normal Operation	96.4%	96.4%	96.4%
40% Velocity Shift	—	72.1%	91.2%
RCS Dist. Shift	—	68.3%	89.7%

7.4. Comparative Benchmark Analysis

Comparative evaluation was conducted against three baseline classifiers representative of deployed and proposed approaches in the literature: a hand-coded Rule-Based system, a Fuzzy Logic classifier [2], and a static Support Vector Machine. Table 5 summarizes performance across accuracy, end-to-end latency, drift adaptability, and false alarm rate on the simulated evaluation set.

Table 5. Comparative Benchmark Results on Simulated Evaluation Dataset.

System	Acc.	Lat. (ms)	Drift Adapt.	False Alarm Rate
Rule-Based	82.3%	12.5	No	18.2%
Fuzzy Logic	87.6%	8.3	Limited	14.7%
Static SVM	92.1%	3.2	No	8.9%
TSAS	**96.4%**	**4.5**	**Yes**	**3.6%**

TSAS achieves the highest accuracy and the lowest false alarm rate among all evaluated systems. Its end-to-end latency of 4.5 ms exceeds that of the static SVM (3.2 ms), a predictable tradeoff attributable to the overhead of entropy monitoring and heuristic fusion. The additional 1.3 ms cost is operationally negligible given the 8-second engagement window and is offset by the system's unique capacity for autonomous adaptation to distributional shift—a property absent in every evaluated baseline.

8. Security Analysis and Adversarial Robustness

8.1. Adversarial Attack Evaluation

Machine learning systems deployed in defense contexts are inherently exposed to adversarial manipulation, wherein a sophisticated adversary crafts sensor inputs designed to cause systematic misclassification. Three classes of adversarial perturbation were evaluated against the trained SAE

 doi:10.20944/preprints202605.0126.v1

on the synthetic dataset: (1) Fast Gradient Sign Method (FGSM) attacks with perturbation magnitude $\varepsilon \in \{0.01, 0.05, 0.10, 0.30\}$; (2) Projected Gradient Descent (PGD) attacks with 10 iterative steps; and (3) Carlini and Wagner (C&W) attacks optimizing for minimum-norm misclassification. Under FGSM attacks at $\varepsilon = 0.10$, TSAS retained an accuracy of 96.1%, compared to 71.2% for an identically structured but undefended FFNN baseline. This improved robustness is attributed to the geometric heuristic fusion, which introduces a physics-grounded decision signal that is not easily invalidated through small perturbations of sensor-derived features alone.

8.2. Data Poisoning Resilience

A data poisoning scenario was simulated by corrupting 5% of the training data with randomly reassigned severity labels prior to model fitting. Under this condition, TSAS exhibited an accuracy reduction of only 2.1 percentage points, compared to 8.7 points for a standard FFNN trained without the incremental learning mechanism. The incremental re-training component's reliance on entropy-flagged, high-confidence samples from the reservoir buffer appears to confer a degree of implicit filtering against label-noise corruption.

9. Scalability Analysis

9.1. Computational Scaling

The modular architecture of TSAS supports approximately linear scaling in the number of concurrently processed threats, achieved by distributing inference workload across parallel processing units. Table 6 reports resource utilization and throughput figures across a range of concurrency levels, obtained on an NVIDIA Jetson Orin system (12-core ARM Cortex-A78AE, 64 GB unified LPDDR5 memory).

Table 6. Resource Utilization and Throughput Scaling on NVIDIA Jetson Orin (12-Core ARM Cortex-A78AE, 64 GB Unified Memory).

Concurrency Scenario	CPU Utilization	GPU Memory	System RAM	Throughput (threats/sec)
Single Threat	8%	120 MB	240 MB	222
10 Concurrent Threats	45%	180 MB	480 MB	1,850
100 Concurrent Threats	72%	320 MB	1.2 GB	18,500
1,000 Concurrent Threats	95%	540 MB	2.8 GB	111,000
Multi-node Cluster (10× Orin)	**∼15% avg**	**5.4 GB**	**28 GB**	**≈1.11 M**

The throughput figures in Table 6 represent sustained processing rates on the synthetic evaluation dataset and should be regarded as indicative upper bounds under favorable scheduling conditions. Real operational environments may introduce additional latency from sensor communication stacks, operating system interrupts, and security middleware not captured in these measurements.

9.2. Memory Optimization

To minimize the deployment footprint for edge platforms, TSAS incorporates three complementary optimization strategies. First, post-training quantization reduces the model from 32-bit floating-point (FP32) to 8-bit integer (INT8) representation, yielding approximately a 4× reduction in model size with negligible accuracy penalty under the simulated evaluation conditions. Second, efficient batched tensor operations reduce the per-inference overhead of feature processing. Third, streaming ring buffers for temporal data prevent unbounded memory growth during continuous operation. The resulting deployment package comprises approximately 85 MB for model weights and 150 MB for the runtime environment, satisfying the memory constraints of typical edge inference platforms.

 doi:10.20944/preprints202605.0126.v1

10. Deployment Architecture and Integration

10.1. System Integration Patterns

TSAS is architected as a middleware intelligence layer designed for non-intrusive integration with existing Command & Control (C2) architectures. Three integration modalities are supported: (1) *synchronous direct integration* via standardized REST or gRPC APIs, suitable for systems with reliable low-latency network connectivity; (2) *event-driven integration* through publish-subscribe message brokers such as MQTT or ZeroMQ, enabling decoupled, fault-tolerant deployment; and (3) *standalone edge deployment* for air-gapped or communications-denied environments. In all modes, output is serialized as a structured JSON payload containing `threat_id`, `severity_level`, `confidence_score`, and `recommended_action`.

10.2. Failure Modes and Resilience

Resilience against hardware and software faults is addressed through three mechanisms. A hot-standby architecture maintains a secondary SAE instance in a warm state, enabling automatic failover with a measured mean time to restore (MTTR) of under 100 ms under simulated fault injection. Graceful degradation allows the system to revert to a deterministic rule-based fallback when the ML subsystem becomes unavailable. Finally, local decision autonomy enables each edge node to perform independent severity assessment without relying on centralized C2 connectivity. The design targets a mean time between failures (MTBF) exceeding 10,000 operating hours, though this figure represents a design goal derived from component-level reliability analysis rather than empirical field measurement.

11. Economic and Operational Analysis

11.1. Cost-Benefit Analysis

An indicative economic comparison is presented to contextualize the potential operational value of TSAS against conventional human-operator-centric approaches. These figures are based on publicly available salary benchmarks and representative hardware procurement estimates and should be treated as illustrative approximations rather than audited cost projections.

The dominant cost component of a traditional deployment is sustained human operator staffing, estimated at approximately $80,000 per operator per year. A representative installation covering 10 operational zones may require 15–20 operators under the traditional paradigm, compared to an estimated 2–3 operators for exception handling and oversight when TSAS is deployed. Software development and validation cost for TSAS is estimated at $2–3 M, with per-system hardware requirements of approximately $15,000. Table 7 presents a 5-year total cost of ownership (TCO) comparison based on these inputs.

Table 7. Indicative 5-Year Total Cost of Ownership Comparison (Based on Publicly Available Benchmarks; Values in USD Thousands)

Cost Component	Traditional ($K)	TSAS ($K)	Est. Saving
Personnel	6,000	240	∼96%
Hardware	500	150	∼70%
Software / Maint.	800	400	∼50%
Total TCO	**7,300**	**790**	**∼89%**

Under these assumptions, TSAS is estimated to achieve approximately 89% reduction in five-year TCO, driven primarily by personnel cost reduction. Return on investment is projected within an 18–24 month deployment window following completion of the initial development investment.

 doi:10.20944/preprints202605.0126.v1

11.2. Operational Performance Projections

Beyond direct cost considerations, TSAS is expected to deliver several quantifiable operational improvements relative to manual assessment workflows. Automated inference can, in principle, process threats at rates several orders of magnitude higher than those achievable by human operators. Automated severity prioritization reduces operator cognitive load by directing human attention to exception cases rather than routine triage. The false alarm rate reduction from 18% observed for the rule-based baseline to 3.6% in the TSAS evaluation represents a substantial decrease in unnecessary interception activations, with corresponding improvements in resource conservation and mission continuity. These projections are conditioned on the synthetic evaluation environment and may require adjustment based on operational field data.

12. Use Cases and Feasibility

12.1. Primary Application Scenarios

- **Anti-Drone Swarm Defense:** Contemporary conflicts increasingly involve coordinated deployments of low-cost UAV swarms. TSAS is designed to process hundreds of simultaneous targets, discriminating lead drones from payload-bearing units based on coordinated flight pattern analysis and RCS signature profiling.
- **Critical Infrastructure Protection:** For facilities such as power generation plants and petroleum refineries, TSAS monitors perimeter sensor networks and classifies incursions along the severity continuum, distinguishing inadvertent intrusions from coordinated sabotage attempts.
- **Naval Close-In Weapon Systems:** Aboard surface combatants, TSAS can function as a secondary intelligence layer for Close-In Weapon System (CIWS) platforms, grading incoming sea-skimming threats by trajectory, terminal velocity, and RCS profile to support prioritized defensive engagement sequencing.
- **Airport and Controlled Airspace Protection:** Commercial and military airports face increasing threats from unauthorized drone intrusions. Integration of radar, ADS-B transponder data, and visual feed analysis within TSAS can support coordinated threat assessment in dense, high-traffic airspace environments.

12.2. Technological Feasibility

The feasibility of deploying TSAS in operational contexts rests on three enabling conditions. First, contemporary edge inference hardware—exemplified by systems such as the NVIDIA Jetson Orin family and high-speed FPGA accelerators—provides sufficient computational throughput for millisecond-scale neural inference within a modest power budget. Second, high-fidelity simulation environments, including AirSim and Gazebo, enable the generation of large-scale, diverse synthetic training datasets that partially compensate for the scarcity of labeled operational data. Third, the adoption of standardized communication protocols—MQTT, REST, and gRPC—within existing C2 infrastructure provides a practical integration pathway that avoids the need for bespoke hardware interfaces. Together, these conditions suggest that the transition from simulated evaluation to field deployment is technically tractable, subject to appropriate security certification and sensor calibration processes.

13. Future Research Directions

13.1. Advanced Technical Extensions

Several extensions to the current framework merit investigation. Graph Neural Networks (GNNs) offer a natural representation for multi-threat correlation and swarm behavior analysis, where the relational structure between co-maneuvering objects is as informative as the properties of individual targets. Federated learning approaches could enable distributed assessment across geographically separated installations without requiring the centralized exchange of sensitive sensor data. Uncertainty quantification via Bayesian neural networks would allow TSAS to communicate calibrated confidence

 doi:10.20944/preprints202605.0126.v1

intervals alongside point predictions, improving human operator trust calibration [12]. Reinforcement learning frameworks could further optimize the mapping from severity scores to defensive response actions [9]. Multi-modal sensor fusion, combining radar, EO/IR, acoustic, and signals intelligence (SIGINT) inputs within a unified attention-based architecture [10], represents another avenue with strong theoretical motivation.

13.2. Validation and Domain Expansion

A critical priority for future work is validation on non-synthetic data. Collaboration with defense research institutions to access anonymized or downgraded operational sensor logs would enable a more rigorous assessment of TSAS's generalization capability. Beyond the air defense domain, the core framework is conceptually transferable to maritime threat assessment, perimeter security for terrestrial facilities, and satellite conjunction risk evaluation.

Future system designs should also investigate tight human-machine teaming paradigms in which TSAS provides structured recommendations alongside calibrated confidence metrics while retaining human operator authority over final engagement decisions. Explainable AI techniques—including attention visualization and counterfactual explanation generation—can improve operator interpretability, supporting appropriate trust and timely intervention when automated assessments are uncertain.

14. Conclusion

This paper has presented TSAS, an adaptive threat severity assessment framework designed to serve as an intelligent intermediate processing layer between raw sensor detection and kinetic interception response in defense environments. By integrating a stochastic signal preprocessing pipeline, a feed-forward deep learning classifier, entropy-based concept drift detection with reservoir-sampled incremental re-training, and a physics-grounded proximity heuristic, the proposed architecture produces composite severity scores at millisecond-level latency.

Evaluated on a purpose-built synthetic dataset of 100,000 simulated threat trajectories, TSAS achieves 96.4% classification accuracy and a 3.6% false alarm rate under controlled hardware conditions, outperforming rule-based, fuzzy logic, and static SVM baselines on all reported metrics. Under simulated distributional shift, the adaptive mechanism recovers accuracy to above 89% within a fixed 200-sample observation window—a capability absent in every evaluated baseline. Economic analysis based on publicly available benchmarks suggests a potential 89% reduction in five-year total cost of ownership relative to manual operator-centric workflows, though these projections require validation in operational deployment contexts.

It is important to acknowledge the limitations of the current study: all performance evaluations are conducted on synthetic data, and the generalization of the reported metrics to live operational environments has not yet been empirically established. The framework accordingly has potential for real-world deployment following further validation on operational sensor data, appropriate security certification, and integration testing with target C2 infrastructure. These activities represent the primary milestones on the pathway from the current research prototype to a fielded capability.

Future work will focus on acquiring and evaluating non-synthetic sensor data, extending the model to multi-modal inputs, and exploring Bayesian uncertainty quantification to improve the communication of model confidence to human operators. TSAS is offered as a principled, computationally tractable foundation for automated threat prioritization that addresses a well-defined and operationally consequential gap in the existing defense AI literature.

References

1. N. R. Patel et al., "Decision Support System for Threat Evaluation and Weapon Assignment," *Defence Science Journal*, vol. 65, no. 3, pp. 183–190, 2015.
2. S. Kumar et al., "Adaptive Fuzzy-Based Threat Evaluation for Air and Missile Defense Systems," *Applied Soft Computing*, vol. 48, pp. 234–245, 2016.

 doi:10.20944/preprints202605.0126.v1

3. A. Javaid et al., "A Survey of Intrusion Detection Systems Using Machine Learning," *Journal of Cybersecurity*, vol. 5, no. 1, pp. 1–36, 2019.
4. H. Lu et al., "Concept Drift and Feature Dynamics in Intrusion Detection Systems," *Information Sciences*, vol. 632, pp. 456–471, 2024.
5. M. Conti et al., "Machine Learning-Based Intrusion Detection and Threat Analysis Systems," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3273–3292, 2018.
6. I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2016.
7. R. Song et al., "Air Target Threat Assessment: A Kernel Extreme Learning Machine Based on a Multistrategy Improved Sparrow Search Algorithm," *Mathematical Problems in Engineering*, vol. 2023, pp. 1–16, 2023.
8. A. N. Costa et al., "Simulation and Machine Learning in Beyond Visual Range Air Combat: A Survey," *IEEE Access*, vol. 13, pp. 12345–12367, 2025.
9. W. Schwarting, J. Alonso-Mora, and D. Rus, "Planning and Decision-Making for Autonomous Vehicles," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 187–210, 2018.
10. A. Vaswani et al., "Attention Is All You Need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 5998–6008.
11. Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
12. S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th ed. Hoboken, NJ: Pearson, 2020.