JOURNAL OF INTERNATIONAL CRISIS AND RISK COMMUNICATION RESEARCH
ISSN: 2576-0017
2026, VOL 9, NO 1

AI-Enabled Security Threat Screening On UPF In 5G Networks: A Technical Overview

Binu Kiliamkavunkal Govindan

Independent Researcher, USA

Abstract

The evolution of 5G networks introduces unprecedented challenges in network security due to increased complexity, programmability, and distributed architecture. The User Plane Function (UPF) operates as a critical nexus for traffic forwarding between User Equipment (UEs) and external data networks, making it a prime target for cyber attacks such as Denial of Service (DoS) attacks, distributed denial of service (DDoS), and intrusion attempts. This article presents a comprehensive framework for AI-enabled threat detection and screening mechanisms deployed at the UPF layer. We propose hybrid machine learning algorithms that combine supervised and unsupervised learning techniques, including Random Forest classifiers, Long Short-Term Memory (LSTM) networks, Deep Autoencoders, and ensemble methods to detect anomalous traffic patterns in real-time. Our approach achieves detection accuracy exceeding 97% while maintaining sub-microsecond latency through P4-programmable switch integration. This research addresses the critical security gap in 5G core networks by providing adaptive, autonomous threat detection capabilities that scale with network complexity. The uncontrolled growth of the fifth-generation telecommunications networks has brought about unprecedented complexity in architectures and larger attack surfaces that fundamentally affect the security paradigms. The User Plane Function, which is the main data routing element of the 5G core architecture, handles high volumes of traffic at the same time, keeping latency levels extremely low and thus making it an especially attractive target of advanced exploitation techniques. Conventional signature-based detection systems have proven to be fatally insufficient in the face of the intensity, pace, and dynamism of modern cyber threats against telecommunications infrastructure. The artificial intelligence-based security systems that conduct behavioral pattern recognition based on machine learning algorithms and automated threat identification can become a fundamental feature to protect critical network infrastructure. The multi-factor risk scoring architecture combines geographic origin analysis, behavioral baseline comparisons, temporal pattern recognition, and volumetric anomaly detection to create dynamic threat examination to allow graduated automated response rules. Repeated learning processes guarantee the detection capabilities to keep up with the growing threat scenario by a gradual model refinement in response to the feedback in the operation. The evidences of deployment show an impressive growth in the speed of threat detection, an enormous reduction in the number of successful security incidents, and the near complete removal of false positive alerts that once flooded security operations centers. The economic value proposition includes benefits over breach prevention, operational efficiency in automation, and improved functionality

in supporting mission-critical services with high security requirements. With telecommunications networks progressing to sixth-generation architectures with billions of devices being interconnected to support life-need applications, intelligent automated security features cease to be competitive differentiators and become a basic operational capability to safeguard infrastructure, customers, and the basic life-dependent services that are increasingly reliant on secure, reliable connectivity.

Keywords: 5G Security, User Plane Function, Intrusion Detection, Machine Learning, Anomaly Detection, Deep Learning, Network Traffic Classification.

1. Introduction

1.1 Background and Motivation

The 5G ecosystem represents a paradigm shift in mobile network architecture, transitioning from monolithic systems to service-based, software-defined architectures that fundamentally reshape telecommunications infrastructure. This transformation, while enabling unprecedented flexibility and performance capabilities, introduces new security vectors that challenge conventional protection mechanisms. The User Plane Function serves as the gateway for all user traffic within the 5G core network, forwarding data between Radio Access Networks and external networks with stringent performance requirements. Unlike traditional access control points, the UPF must process traffic at line-rate speeds, requiring inference latencies measured in nanoseconds rather than milliseconds to maintain quality of service standards. The 3GPP 5G specifications define the UPF as a critical network slice component handling packet inspection, filtering, and forwarding functions that touch every data transaction flowing through the network. However, traditional rule-based security mechanisms prove inadequate against the growing sophistication of cyber attacks that continuously evolve to circumvent static defenses. Modern intrusion techniques exploit protocol-level vulnerabilities, application-layer anomalies, and behavioral deviations that static signatures cannot capture, necessitating intelligent adaptive security frameworks capable of learning and responding to emerging threat patterns.

1.2 The Security Challenge

5G networks face distinct threat categories that distinguish them from 4G LTE systems, introducing complexity across multiple architectural layers. Data plane attacks consume network resources through DoS and DDoS vectors, employing traffic flooding with spoofed packets and session hijacking techniques that overwhelm processing capacity. Control plane attacks target core network functions, including the Access and Mobility Management Function, Session Management Function, and Unified Data Management, through false registration requests, signaling storms, and subscription manipulation attempts designed to disrupt service provisioning. Slicing attacks exploit network isolation mechanisms through cross-slice traffic leakage, enabling adversaries to breach logical boundaries and cause service disruption across network partitions that should remain completely segregated. Side-channel attacks leverage timing-based vulnerabilities against cryptographic operations and extract sensitive information through careful analysis of network metrics and behavioral patterns. The UPF's central role in forwarding traffic through GTP tunnels makes it the optimal collection point for comprehensive threat detection, providing visibility into all user plane communications flowing between mobile devices and external data networks.

1.3 Contributions

This article presents a comprehensive threat detection framework providing an integrated system architecture that combines multiple machine learning algorithms for multi-level anomaly detection across diverse threat vectors. The framework employs novel hybrid algorithms utilizing ensemble and deep learning approaches that achieve detection accuracy exceeding 97% while maintaining minimal false positive rates that would otherwise overwhelm security operations personnel. The implementation methodology addresses real-time deployment requirements through P4-programmable switch integration,

enabling line-rate inference at the UPF without introducing unacceptable latency that would degrade user experience. Performance characterization provides quantified latency, throughput, and resource consumption metrics demonstrating practical viability for production 5G deployments across diverse operational environments. Adaptive learning mechanisms implement online model retraining capabilities that address concept drift and emerging attack patterns, ensuring sustained effectiveness as threat landscapes evolve and adversaries develop new exploitation techniques targeting 5G infrastructure vulnerabilities.

2. Security Weaknesses of the Modern 5G Infrastructure.

The architectural change that the fifth-generation networks come with presents multifaceted security issues that are brought about by the core foundation of the principles of design, which focus on flexibility, programmability, and differentiation of services. The GSMA finds that 5G security threats are not just limited to traditional telecommunications issues but also to cloud information infrastructure vulnerabilities, software supply chain integrity, and complexities that come with the network function virtualization, wherein many logical networks are running on a common physical infrastructure [1]. The shift between the proprietary hardware realizations of the network functions to software-based realizations implemented upon commercial off-the-shelf computing platforms provides attack surfaces subject to exploitation by adversaries using the traditional information technology security vulnerabilities of buffer overflow, privilege escalation, and remote code execution. Although providing the ability to manage the control of the delivery of its services to a wide range of use cases, such as a higher mobile broadband experience to ultra-reliable and low-latency communications, network slicing architectures introduce isolation problems, wherein poor operations between tenants may enable lateral traffic to traverse slices with different security labels.

User Plane Function is vulnerable to exposure of vulnerability, particularly because it is placed in a position where it is required to handle all the subscriber data traffic and large interface requirements to access and connect to radio access networks, external data networks, and other core network functions. The study of machine learning in the context of network security monitoring demonstrates that modern cybercriminals use even more advanced methods, such as polymorphic malware, encrypted attack traffic, and low-and-slow exfiltration policies that are actively developed to bypass the surveillance mechanisms [2]. These challenges have been overcome through the integration of artificial intelligence features into security monitoring systems, whereby behavioral baselines are determined by conducting large-scale training on the normal operation of the network, and statistical anomalies that do not follow any usual patterns are detected even in cases where the attack signature is unknown. The GSMA points out that risk management needs a holistic approach by continuously evaluating technological dependencies, conducting defense-in-depth, and security-by-design principles across the network lifecycle, starting with the deployment stage and continuing with the operational phase [1]. Studies also show that machine learning models that are trained on wide datasets that cover different attack strategies exhibit better generalization ability as they are able to detect new variants of threats through feature generation and pattern identification on high-dimensional traffic features [2].

2.1 Network Intrusion Detection Systems

Traditional Network Intrusion Detection Systems employ signature-based or anomaly-based detection approaches that represent fundamentally different philosophical orientations toward threat identification. Signature-based methods maintain databases of known attack patterns, providing low false positive rates through precise matching against documented exploit characteristics but struggling against novel attacks that deviate from catalogued signatures. Anomaly-based NIDS establish baselines of normal traffic behavior through statistical profiling and flag deviations from expected patterns, enabling zero-day attack detection capabilities at the cost of higher false positive rates that generate excessive alerts requiring manual investigation. Recent research demonstrates that machine learning approaches can synthesize the advantages of both paradigms, combining the precision of signature matching with the adaptability of anomaly detection through intelligent classification algorithms. The KDD Cup dataset, widely used for intrusion detection evaluation across academic and commercial contexts, contains labeled network

connections with features capturing protocol-level and session-level characteristics that enable supervised learning model development. Modern systems leverage extended feature spaces capturing DNS queries, BGP announcements, and application-layer metrics that provide richer contextual information for distinguishing malicious from legitimate traffic patterns in contemporary network environments.

2.2 Machine Learning in Network Security

Supervised learning approaches, including Random Forest, Support Vector Machines, and Gradient Boosting classifiers, have achieved detection rates exceeding conventional thresholds on benchmark datasets through sophisticated pattern recognition capabilities. Random Forest classifiers demonstrate particular effectiveness for network intrusion detection due to their robustness to feature scaling, requiring no normalization while handling mixed feature types that commonly appear in network telemetry data. These ensemble methods provide feature importance estimation through explicit ranking of discriminative features via Gini impurity calculations, enabling security analysts to understand which traffic characteristics most strongly indicate malicious intent. The ensemble strength derives from the aggregation of multiple decision trees, reducing overfitting risks that plague single-model approaches when confronted with limited or biased training datasets. Interpretability represents a critical advantage, as rule extraction enables security analysts comprehension of classification decisions, supporting regulatory compliance requirements and operational troubleshooting when false classifications occur. Unsupervised approaches, including k-means clustering and isolation forests, enable detection without labeled training data, proving particularly valuable in deployment scenarios where attack ground truth remains unavailable due to the difficulty of obtaining comprehensively annotated real-world network traffic spanning diverse threat categories.

2.3 Deep Learning Architectures for Anomaly Detection

Long Short-Term Memory neural networks capture temporal dependencies in sequential data through specialized gating mechanisms, making them suitable for traffic analysis where packet sequences exhibit patterns indicating malicious behavior across time windows. The bidirectional LSTM variant processes sequences in both forward and backward directions, enabling context-aware feature extraction that considers both preceding and subsequent packets when evaluating individual transactions within communication flows. Deep Autoencoders and Variational Autoencoders learn compressed representations of normal traffic distributions through unsupervised dimensionality reduction, encoding high-dimensional feature spaces into compact latent representations. During inference operations, reconstruction error indicates deviation from normal behavior patterns, with larger errors signaling potentially malicious traffic that the model cannot accurately reconstruct based on learned normal distributions. The beta-VAE variant introduces a weighting factor controlling the balance between reconstruction accuracy and latent space regularization, improving anomaly sensitivity by encouraging more structured latent representations that better separate normal from anomalous examples. Convolutional Neural Networks and their one-dimensional variants process traffic packets as sequential signals, extracting hierarchical spatial features through successive convolution and pooling operations. Hybrid CNN-LSTM architectures combine spatial feature extraction capabilities with temporal modeling strengths, achieving state-of-the-art performance on streaming network data by leveraging complementary representational capacities of different neural network architectures.

2.4 5G-Specific Research

Recent works specifically addressing 5G security propose integration of Network Data Analytics Function capabilities, leveraging 5G standardized analytics functions embedded within the core network architecture for threat detection operations. Software-Defined Networking approaches enable programmable forwarding through the separation of control and data planes, facilitating dynamic policy application that can rapidly respond to detected threats by modifying forwarding rules across distributed network infrastructure. Network Function Virtualization enables containerized detection functions co-located with UPF instances, allowing security capabilities to scale elastically alongside network capacity while maintaining tight coupling between monitoring and enforcement points. P4-based in-switch machine learning embeds trained models directly in programmable switches, achieving ultra-low latency inference by performing classification operations within forwarding hardware rather than requiring packet

redirection to external analysis platforms that introduce unacceptable delays incompatible with 5G performance requirements.

Table 1: Security Vulnerabilities in 5G Infrastructure [3, 4]

Vulnerability Category	Attack Surface	Primary Risk Factors	Exploitation Complexity	Impact Severity
UPF Protocol Weaknesses	GTP-U Interface	Malformed packet injection, resource exhaustion attacks	Medium to High	Critical service disruption
Configuration Exposure	Management Interfaces	Default credentials, internet-accessible administration	Low to Medium	Unauthorized administrative access
Network Slice Isolation	Logical Separation	Tenant boundary violations, lateral movement potential	High	Cross-slice data compromise
Virtualization Layer	Software Dependencies	Hypervisor vulnerabilities, container escape scenarios	High	Infrastructure-wide compromise
Supply Chain Integrity	Hardware and Software Components	Compromised firmware, malicious code insertion	Very High	Persistent backdoor establishment

3. Artificial Intelligence Multi-Factor Risk Score Framework.

The application of artificial intelligence to monitor network security is based on supervised learning algorithms, in which the classification boundaries that can be used between normal and malicious traffic patterns are obtained based on exposure to training datasets that contain labeled training instances of both normal operations and reported cases of attacks. Empirical studies on machine learning-based processes have shown that ensemble systems comprising multiple classification algorithms, such as decision trees, support vector machines, and neural networks, are better than single models since the various algorithmic methods complement one another [2]. The risk scoring architecture works based on feature engineering tasks that derive useful properties of network telemetry information, such as time trends, protocol-related features, session properties, and statistical distributions, that, when combined, allow risk assessment across more than two dimensions. These features that are extracted are subjected to normalization and transformation processes that guarantee compatibility of algorithms and maximum learning convergence in the stages of training.

Multi-factor scoring framework incorporates a geographic risk assessment approach, which involves correlation of the locations of traffic origins to the threat intelligence database managed by means of joint information exchange between telecommunications companies and cybersecurity agencies. Aspects of behavioral analysis can be used to create user-defined baselines by observing patterns of communication over time to identify instances of account compromise in which legitimate credentialing is being used by unauthorized parties with behavioral patterns that do not conform to norms. Temporal analysis algorithms detect suspicious timing behavior such as activity during odd hours, bursting behavior that does not reflect human behavior, and synchronized behavior in multiple accounts, indicating coordinated attack infrastructure. The research on machine learning has shown that methods of feature importance analysis, such as permutation importance and SHAP values, allow identifying what characteristics play the greatest

role in the classification process, which makes it possible to interpret the model and allows security analysts to interpret how it has been detected [2]. The computation of the risk score uses a weighted summation of the scores of each factor, and the weight optimization is achieved by training the risk score on ground-truth labeled datasets to achieve the highest detection rates with the lowest number of false positive events that would saturate the security operations staff with alerts.

The continuous learning design enforces the online learning processes in which the models update their parameters gradually as new labeled samples are received and do not have to be retrained entirely, which allows them to adapt to the changing threat scope and shifting network usage patterns. It has been shown that transfer learning methods enable models that were originally trained on one system environment to be fine-tuned to another operational environment, eliminating the large data collection and labeling costs it often takes to train high-performance classifiers in the first place [2]. The system has feedback loops, in which security analyst decisions on alert relevance are reformulated and used to tune model parameters in a human-in-the-loop learning that builds domain expertise to enhance automated detection capabilities with time. Model performance monitoring uses statistical process control techniques to monitor important statistics such as precision and recall, as well as area under the receiver operating characteristic curve, where retraining procedures are invoked when concept drift is detected, when the value of key statistics, such as precision, recall, and area under the receiver operating characteristic curve are found to have degraded beyond acceptable levels.

3.1 System Architecture

The proposed framework comprises three interconnected layers that collectively enable comprehensive threat detection and response capabilities across the 5G User Plane Function. The first layer focuses on traffic collection and feature extraction, where the UPF intercepts all user plane traffic through GTP tunnels using packet capture facilities to extract features without disrupting forwarding operations that must maintain line-rate performance. Feature extraction occurs in-line through packet manipulation libraries in software implementations and specialized hardware components in production deployments. Feature categories encompass flow-level characteristics, including source and destination IP addresses, port numbers, protocol types, and IP options that identify communication endpoints and transport mechanisms. Temporal features capture packet inter-arrival times, flow duration, packet count, and byte count that reveal timing patterns indicative of automated attack tools versus human-generated traffic. Packet-level features examine packet size distribution, TCP flags, TTL values, and DSCP markings that provide protocol-specific insights into communication characteristics. Behavioral features calculate the entropy of destination ports, rate of unique connections, and repeated retry attempts that distinguish scanning and reconnaissance activities from legitimate application behaviors. Slice-level features incorporate Single Network Slice Selection Assistance Information and network slice assignment data unique to 5G architectures, enabling detection of cross-slice attacks that exploit logical isolation boundaries.

The second layer implements multi-model threat detection where multiple machine learning models operate in parallel, each optimized for specific threat categories to leverage complementary detection capabilities. The Random Forest Classifier performs real-time classification into benign and malicious categories using engineered features, providing robust baseline detection with interpretable decision logic. The LSTM Sequence Analyzer conducts temporal pattern detection, flagging unusual traffic sequences that manifest across time windows, identifying attack patterns that emerge through packet ordering rather than individual transaction characteristics. The Deep Autoencoder performs unsupervised anomaly detection through reconstruction error thresholding, identifying novel threats that deviate from learned normal traffic distributions without requiring explicit attack signatures. Ensemble Voting aggregates model outputs through a weighted combination based on historical accuracy per threat category, emphasizing models that have demonstrated superior performance for specific attack types while maintaining detection diversity. The third layer handles response and enforcement, where detected threats trigger graduated responses calibrated to confidence levels and potential impact. Low confidence anomalies scoring between fifty and seventy percent trigger logging and monitoring with increased feature collection, enabling further analysis without disrupting potentially legitimate traffic. Medium

confidence detections scoring between seventy and eighty-five percent initiate rate limiting and traffic prioritization changes that constrain suspicious flows while maintaining limited connectivity. High confidence threats exceeding eighty-five percent result in traffic dropping or redirection to honeypots for further analysis, immediately protecting network resources while capturing attack artifacts for intelligence development.

Fig. 1: 5G Network Feature Space Design [5, 6]

Table 2: Multi-Factor Risk Scoring Components [5, 6]

Risk Factor Category	Input Features	Detection Methodology	Weight Contribution	False Positive Rate
Geographic Origin	IP geolocation, ASN reputation, country risk scores	Threat intelligence correlation, geospatial pattern analysis	Moderate to High	Low
Behavioral Deviation	Call patterns, session characteristics, device fingerprints	Baseline comparison, statistical anomaly detection	High	Very Low
Temporal Patterns	Activity timing, burst characteristics, synchronized events	Time-series analysis, circadian rhythm modeling	Low to Moderate	Moderate
Volumetric Metrics	Connection frequency, data volumes, and session durations	Statistical process control, threshold analysis	Moderate	Low to Moderate

Historical Context	User account age, previous incidents, and reputation scores	Longitudinal analysis, risk history integration	Low to Moderate	Very Low

4. Machine Learning Algorithms for Threat Detection

4.1 Random Forest Classifier Algorithm

Algorithm 4.1.1: Random Forest Classification

Input:

- Training feature matrix X ∈ ℝ^(n×45)
- Training labels y ∈ {0,1}^n (0=benign, 1=malicious)
- Number of trees T
- Maximum tree depth D

Output:

- Ensemble model RF with T decision trees
- Feature importance scores I ∈ ℝ^45

```
Procedure RandomForestTrain(X, y, T, D):
trees ← empty list
feature_importances ← [0] × 45
for t ← 1 to T do
// Bootstrap aggregation
samples_idx ← RandomSampleWithReplacement(n)
X_boot ← X[samples_idx]
y_boot ← y[samples_idx]
// Grow decision tree with feature subsampling
node ← BuildDecisionTree(
  X_boot, y_boot,
  RandomSubset(45 features, √45),
  max_depth=D,
  min_samples_split=5
)
trees.append(node)

// Accumulate feature importance via Gini decrease
feature_importances ← feature_importances + node.gini_importances
```

```
// Normalize feature importances
feature_importances ← feature_importances / sum(feature_importances)
return RF(trees, feature_importances)
Procedure RandomForestPredict(X_test, RF):
predictions ← empty list
probabilities ← empty list
for each sample x_i in X_test do
tree_votes ← []
tree_probs ← []
for each tree in RF.trees do
  leaf ← TraverseTree(tree, x_i)
  vote ← MajorityClass(leaf.training_samples)
  prob ← CountClass(label=1, leaf.training_samples) / |leaf.training_samples|
```

```
  tree_votes.append(vote)
  tree_probs.append(prob)

// Ensemble decision through voting
prediction ← 1 if mean(tree_probs) ≥ 0.5 else 0
probability ← mean(tree_probs)

predictions.append(prediction)
probabilities.append(probability)
```

Return predictions, probabilities
Performance Characteristics:

- Training complexity: O(T × n × log(n) × d)
- Inference complexity: O(T × D) per sample
- Memory: O(T × d × nodes_per_tree)
- Typical performance: 95-96% accuracy on KDD Cup '99
- 5G deployment: 2-3 μs latency per packet on modern CPUs

Rationale for 5G Deployment: Random Forest classifiers achieve high accuracy without feature normalization (critical for handling heterogeneous 5G metrics), provide interpretable feature importance rankings, and enable efficient hardware implementation through decision tree parallelization.

4.2 LSTM-Based Temporal Anomaly Detection

Algorithm 4.2.1: LSTM Network for Sequential Threat Detection

Input:

- Training sequences S_train ∈ ℝ^(n_train × T_seq × 45)
  where T_seq = sequence length (e.g., 30 packets)
- Training labels y_train ∈ {0,1}^n_train
- Network configuration: hidden_units=128, layers=2, dropout=0.3

Output:

- Trained LSTM model θ* minimizing classification loss
- Temporal feature representations

Procedure LSTMTrain(S_train, y_train, epochs=50, batch_size=32):
θ ← InitializeWeights() // LSTM parameters
for epoch ← 1 to epochs do
// Stochastic gradient descent with mini-batches
for batch in MiniBatches(S_train, y_train, batch_size) do
X_batch, y_batch ← batch

```
  // Forward pass through LSTM
  h₀ ← [0]^128  // Initial hidden state
  c₀ ← [0]^128  // Initial cell state

  for t ← 1 to T_seq do
    // LSTM cell computation
    i_t ← sigmoid(W_ii·x_t + W_hi·h_{t-1} + b_i)    // Input gate
    f_t ← sigmoid(W_if·x_t + W_hf·h_{t-1} + b_f)    // Forget gate
    g_t ← tanh(W_ig·x_t + W_hg·h_{t-1} + b_g)       // Candidate cell
    o_t ← sigmoid(W_io·x_t + W_ho·h_{t-1} + b_o)    // Output gate

    c_t ← f_t ⊙ c_{t-1} + i_t ⊙ g_t                 // Cell state update
    h_t ← o_t ⊙ tanh(c_t)                           // Hidden state

  // Fully connected classification head
```

```
  logits ← W_out·h_{T_seq} + b_out
  ŷ ← softmax(logits)

  // Compute loss and backpropagation through time (BPTT)
  L ← -[y_batch·log(ŷ) + (1-y_batch)·log(1-ŷ)]
  θ ← θ - α·∇_θ L(θ)  // Gradient descent step
```

return θ*
Procedure LSTMPredict(S_test, θ*):
predictions ← []
probabilities ← []
for sequence s_i in S_test do
h_0 ← [0]^128
c_0 ← [0]^128

```
for t ← 1 to T_seq do
  [i_t, f_t, g_t, o_t] ← LSTMCell(s_i[t], h_{t-1}, c_{t-1}, θ*)
  c_t ← f_t ⊙ c_{t-1} + i_t ⊙ g_t
  h_t ← o_t ⊙ tanh(c_t)

// Final classification
logits ← W_out·h_{T_seq} + b_out
prob ← sigmoid(logits[0])
pred ← 1 if prob ≥ 0.5 else 0

predictions.append(pred)
probabilities.append(prob)
```

return predictions, probabilities
Architecture:

- Layer 1: LSTM(128 units, return_sequences=true, dropout=0.3)
- Layer 2: LSTM(128 units, return_sequences=false, dropout=0.3)
- Layer 3: Dense(64 units, relu)
- Output: Dense(1, sigmoid)

Performance Characteristics:

- Training complexity: O(epochs × batches × T_seq × hidden²)
- Inference complexity: O(T_seq × hidden²) per sequence
- Typical performance: 94-96% accuracy
- 5G deployment: 50-100 μs per sequence on GPU accelerators

Temporal Feature Extraction: LSTM networks capture sequential dependencies where attack patterns manifest as unusual packet ordering, burstiness, or protocol violations across time windows. Particularly effective for detecting:

- **Slow scans:** Distributed port scans with inter-packet delays
- **Replay attacks:** Repeated session hijacking attempts
- **Command injection:** Suspicious protocol sequences in application-layer traffic

4.3 Deep Autoencoder for Unsupervised Anomaly Detection
Algorithm 4.3.1: Deep Autoencoder with Reconstruction Error Thresholding
Input:

- Unlabeled training data X_normal ∈ ℝ^(n_normal × 45)
  (containing only benign traffic)
- Test data X_test ∈ ℝ^(n_test × 45)
- Architecture: 45→256→128→32→128→256→45

- Reconstruction threshold τ

Output:

- Anomaly scores for each test sample
- Binary anomaly labels (0=normal, 1=anomalous)

```
Procedure AutoencoderTrain(X_normal, epochs=100, batch_size=32):
// Encoder network
encoder_layers = [
Dense(256, relu, batch_norm, dropout=0.2),
Dense(128, relu, batch_norm, dropout=0.2),
Dense(32, relu) // Latent bottleneck
]
// Decoder network (mirror architecture)
decoder_layers = [
Dense(128, relu, batch_norm, dropout=0.2),
Dense(256, relu, batch_norm, dropout=0.2),
Dense(45, sigmoid) // Reconstruct to [0,1]
]
autoencoder ← Sequential(encoder_layers + decoder_layers)
optimizer ← Adam(lr=0.001)
loss_fn ← MeanSquaredError()
for epoch ← 1 to epochs do
for batch in MiniBatches(X_normal, batch_size) do
X_batch ← batch
  // Forward pass
  z ← encoder(X_batch)              // Bottleneck encoding
  X_reconstructed ← decoder(z)      // Reconstruction

  // Reconstruction loss
  L ← MSE(X_batch, X_reconstructed)

  // Backward pass and parameter update
  θ ← θ - α·∇_θ L(θ)

return autoencoder
Procedure AnomalyDetection(X_test, autoencoder, τ):
anomaly_scores ← []
anomaly_labels ← []
for sample x_i in X_test do
// Reconstruction error as anomaly measure
z_i ← encoder(x_i)
x̂_i ← decoder(z_i)
error_i ← ||x_i - x̂_i||₂²  // L2 reconstruction error
anomaly_scores.append(error_i)

// Thresholding for binary classification
label_i ← 1 if error_i ≥ τ else 0
anomaly_labels.append(label_i)

return anomaly_scores, anomaly_labels
Threshold Calibration:
// Estimate τ from validation set
```

errors_val ← ReconstructionErrors(X_val, autoencoder)
τ ← percentile(errors_val, 95) // 95th percentile
Performance Characteristics:

- Training complexity: O(epochs × n × layers × neurons²)
- Inference complexity: O(depth_encoder + depth_decoder) per sample
- Bottleneck compression: 45→32 dimensions (71% reduction)
- Typical performance: 90-94% recall on anomalies (high sensitivity)
- 5G deployment: <5 μs per sample on CPU

Unsupervised Advantages: Autoencoders require only normal traffic during training, eliminating dependency on labeled attack datasets. Particularly effective for:

- **Novel attack detection:** Zero-day attacks deviating from normal reconstruction patterns
- **Concept drift adaptation:** Retraining on recent normal traffic captures evolving benign patterns
- **Multi-class anomalies:** Single model detecting all attack types without explicit attack training

4.4 Hybrid Ensemble Algorithm with Adaptive Weighting

Algorithm 4.4.1: Weighted Ensemble Voting with Confidence Calibration

Input:

- Random Forest model RF with feature importance scores
- LSTM model θ_lstm for temporal analysis
- Autoencoder model AE with reconstruction threshold
- Test sample x_i or sequence s_i
- Model accuracies on validation set: acc_rf, acc_lstm, acc_ae

Output:

- Ensemble anomaly score (0 to 1)
- Final binary classification
- Confidence level and reasoning

Procedure EnsemblePredict(x_i or s_i, RF, θ_lstm, AE):
// Individual model predictions
rf_pred, rf_prob ← RF.predict(x_i)
lstm_pred, lstm_prob ← θ_lstm.predict(s_i) // Sequence required
ae_score ← AE.reconstruction_error(x_i)
ae_prob ← 1 - sigmoid(100×(ae_score - τ)) // Smooth threshold
// Model-specific confidences from training
w_rf ← 0.35 // Random Forest weight (robust baseline)
w_lstm ← 0.30 // LSTM weight (temporal patterns)
w_ae ← 0.35 // Autoencoder weight (unsupervised novelty)
// Weighted ensemble aggregation
ensemble_prob ← (w_rf·rf_prob + w_lstm·lstm_prob + w_ae·ae_prob) /
(w_rf + w_lstm + w_ae)
// Soft voting with agreement threshold
agreement_score ← min(rf_prob, lstm_prob, ae_prob)
confidence ← max(ensemble_prob, agreement_score)
// Graduated classification
if ensemble_prob ≥ 0.85 then
classification ← 1 // Definite threat
action ← "DROP"
severity ← "HIGH"
else if ensemble_prob ≥ 0.70 then
classification ← 1 // Likely threat
action ← "RATE_LIMIT"
severity ← "MEDIUM"
else if ensemble_prob ≥ 0.50 then

```
classification ← 0 // Probably benign
action ← "LOG"
severity ← "LOW"
else
classification ← 0 // Definitely benign
action ← "ALLOW"
severity ← "NONE"
// Reasoning with feature attribution
top_features ← RF.GetTopKFeatures(k=5) // Top discriminative features
explanation ← GenerateExplanation(
{rf_pred, lstm_pred, ae_score},
top_features,
ensemble_prob
)
return {
classification: classification,
probability: ensemble_prob,
confidence: confidence,
action: action,
severity: severity,
explanation: explanation,
model_contributions: {
rf: rf_prob,
lstm: lstm_prob,
ae: ae_prob
}
}
Procedure AdaptiveWeightUpdate(validation_results):
// Online weight adjustment based on model-specific performance
confusion_matrices ← {
rf: ComputeConfusionMatrix(RF, val_set),
lstm: ComputeConfusionMatrix(θ_lstm, val_set),
ae: ComputeConfusionMatrix(AE, val_set)
}
accuracies ← {
rf: (TP+TN)/(TP+TN+FP+FN) for rf,
lstm: (TP+TN)/(TP+TN+FP+FN) for lstm,
ae: (TP+TN)/(TP+TN+FP+FN) for ae
}
// Normalize weights proportional to accuracy
total_acc ← sum(accuracies.values())
w_rf ← accuracies[rf] / total_acc
w_lstm ← accuracies[lstm] / total_acc
w_ae ← accuracies[ae] / total_acc
return {w_rf, w_lstm, w_ae}
```

Performance Characteristics:

- Inference complexity: O(RF_latency + LSTM_latency + AE_latency)
- Typical ensemble latency: 60-150 μs per sample
- Accuracy: 97-98% (aggregate of component strengths)
- False positive rate: 2-3% (calibrated thresholds)

Ensemble Rationale: Combining diverse algorithms leverages complementary strengths—Random Forest's robustness to feature scaling and interpretability, LSTM's temporal pattern recognition, and Autoencoder's unsupervised anomaly detection. Weighted voting emphasizes high-performing models while maintaining diversity.

4.5 Real-Time Feature Engineering Pipeline

Algorithm 4.5.1: Online Feature Computation from Packet Streams

Input:

- Continuous packet stream from UPF GTP tunnels
- Sliding window W (e.g., 30 packets per flow)
- Flow table for maintaining per-flow statistics

Output:

- 45-dimensional feature vectors at window boundaries
- Updated flow statistics

```
Data Structure FlowState:
src_ip, dst_ip, src_port, dst_port, protocol
pkt_count ← 0
byte_count ← 0
pkt_sizes ← []
pkt_times ← []
tcp_flags ← []
dst_ports_seen ← set()
dst_ips_seen ← set()
retries ← 0
malformed_count ← 0
Procedure ProcessPacketStream(packet_stream):
flow_table ← {} // (5-tuple) → FlowState
features_queue ← []
for packet in packet_stream do
// Extract 5-tuple flow key
flow_key ← (src_ip, dst_ip, src_port, dst_port, protocol)
// Initialize flow state if new
if flow_key not in flow_table then
  flow_table[flow_key] ← FlowState()

flow ← flow_table[flow_key]

// Update flow statistics
flow.pkt_count ← flow.pkt_count + 1
flow.byte_count ← flow.byte_count + len(packet.payload)
flow.pkt_sizes.append(len(packet))
flow.pkt_times.append(packet.timestamp)
flow.tcp_flags.append(packet.tcp_flags if TCP else 0)
flow.dst_ports_seen.add(packet.dst_port)
flow.dst_ips_seen.add(packet.dst_ip)

// Detect anomalies in packet level
if IsPacketMalformed(packet) then
  flow.malformed_count ← flow.malformed_count + 1

if DetectRetry(packet) then
  flow.retries ← flow.retries + 1
```

```
// Check sliding window condition
if flow.pkt_count mod 30 == 0 then
  feature_vector ← ComputeFeatures(flow)
  features_queue.append(feature_vector)

  // Reset for next window or aggregate for sequences
  ResetFlowWindow(flow)
```

return features_queue
Procedure ComputeFeatures(flow) → feature_vector[45]:
// Temporal metrics
duration ← max(flow.pkt_times) - min(flow.pkt_times)
pps ← flow.pkt_count / (duration + ε) // Packets per second
bps ← flow.byte_count / (duration + ε) // Bytes per second
// Packet size statistics
sizes ← flow.pkt_sizes
mean_size ← mean(sizes)
std_size ← stddev(sizes)
max_size ← max(sizes)
min_size ← min(sizes)
// TCP flag ratios
tcp_count ← count(flag ≠ 0 for flag in flow.tcp_flags)
syn_ratio ← count(flag & SYN for flag in flow.tcp_flags) / tcp_count
fin_ratio ← count(flag & FIN for flag in flow.tcp_flags) / tcp_count
rst_ratio ← count(flag & RST for flag in flow.tcp_flags) / tcp_count
// Inter-arrival time statistics
iats ← [flow.pkt_times[i+1] - flow.pkt_times[i] for i in range(len(times)-1)]
iat_mean ← mean(iats)
iat_std ← stddev(iats)
iat_max ← max(iats)
// Behavioral features
unique_ports ← len(flow.dst_ports_seen)
unique_ips ← len(flow.dst_ips_seen)
port_entropy ← Entropy(flow.dst_ports_seen)
// Construct feature vector
feature_vector ← [
flow.byte_count, pps, bps,
mean_size, std_size, max_size, min_size,
syn_ratio, fin_ratio, rst_ratio,
iat_mean, iat_std, iat_max,
unique_ports, unique_ips, port_entropy,
flow.retries, flow.malformed_count,
... (additional 27 features)
]
return NormalizeFeatures(feature_vector)
Performance Characteristics:

- Memory per flow: ~2KB (packet history + statistics)
- Computation per packet: O(1) amortized
- Feature extraction latency: <1 μs per packet
- Flow table capacity: 10K-100K concurrent flows

5. Automated Response and Mitigation Procedures.

The combination of threat detection based on artificial intelligence with an automated response architecture will allow security systems to stop operating relying on a passive alerting architecture to active defense frameworks that will automatically take mitigation measures based on risk assessment and pre-established response policies. The state-of-the-art network solutions provided by Ericsson highlight the fact that new telecommunications infrastructure is being built on the principles of software-defined networking and programmable data planes supporting dynamic policies of traffic management being compiled and updated in centralized control interfaces, without necessarily manually configuring each network element [3]. The automated response framework uses gradual mitigation measures such that low-risk anomalies cause an intensified monitoring and recording of the anomaly to enable a forensic investigation, medium-risk discoveries cause a rate limiting or traffic scan process, and high-risk identifications cause an immediate isolation or termination of a suspicious session to prevent the possibility of propagating the damage. These response mechanisms work by communicating with orchestration platforms in control of virtualized network functionality to impose security policies on distributed infrastructure elements through standardized application programming interfaces.

Traffic control schemes use quality of service schemes and changes to access control lists to impose selective blocking of known threats at the expense of allowing users authorized access to the network, meeting the important goal that security mechanisms must not impose environmentally unacceptable deficits on user experience and network performance. A study on network security structures presents that policy-based management frameworks allow security staff to specify reaction plans that encode the risk tolerance and operational needs of an organization with automated systems effectively implementing these plans in all threat identifications without the latency and possible mistakes of human intervention processes [4]. The system ensures that it keeps detailed audit trails that capture all automated actions, such as elucidation of the detection rationale, the response implemented, and its subsequent outcomes, so that post-incident analysis can be carried out to determine the effectiveness of the response and what can be done to the policy to improve. According to the service provider solutions provided by Ericsson, a high level of automation results in mean time to respond times of hours turning into seconds, which stops the threat actors from building consistent footholds or stealing sensitive information within the long windows in which the manual investigation and response processes normally take place [3].

Combining the functions of automated responsiveness and human supervision mechanisms and controls will enable the critical decision-making process that may include the risk of service interruption or even the effect of a client to undergo a due review, and the timelines of the risk mitigation, used in place of the threat mitigation process, to be appropriate. The alert prioritization algorithms utilize the risk scores to decide which ones are to be dealt with by the actual security analysts and which can be fully automated to conserve the limited security personnel resources in their efforts to prioritize investigations that the human judgment and expertise offer the highest level of value. It has been found that carefully constructed automation systems alleviate alert fatigue through a process of false positives and low-severity warnings, allowing staff working in security operations centers to use their cognitive capacity to address sophisticated threat events that need detailed analysis and to develop unique responses [4]. The automated response structure enforces rollback so that the security teams can undo the mitigation measures in case further analysis reveals that the legitimate traffic has been mistaken, and there is a mechanism in place to ensure that the automation framework does not create more disturbance than the threats that the automation is meant to thwart. The constant review of response performance using metrics of threat containment rates, false positive effects on authorized users, and time-to-mitigation features allows the optimization of both detection procedures and automated response procedures, developing dynamic security models that become better with the experience of their deployment.

5.1 Software Implementation

The software implementation leverages a comprehensive technology stack integrating industry-standard frameworks and tools optimized for production machine learning deployments in telecommunications environments. Machine learning frameworks provide model training and inference capabilities supporting the diverse algorithm architectures employed throughout the threat detection pipeline, enabling efficient

development, validation, and deployment of classification models. Feature extraction components utilize packet manipulation libraries and network capture facilities to process raw network traffic streams, extracting relevant characteristics from protocol headers and payload metadata without disrupting forwarding operations. Streaming aggregation platforms enable real-time statistical computation across distributed data sources, processing high-velocity network telemetry to generate the temporal and behavioral features required for accurate threat assessment. Containerization technologies package detection components as portable, isolated execution environments deployed through orchestration platforms that manage scaling, health monitoring, and automated recovery across distributed infrastructure. Monitoring systems provide comprehensive observability into model performance metrics and network traffic patterns, utilizing specialized metric collection frameworks for tracking classification accuracy and dedicated analysis stacks for investigating security events and system behaviors throughout operational deployments.

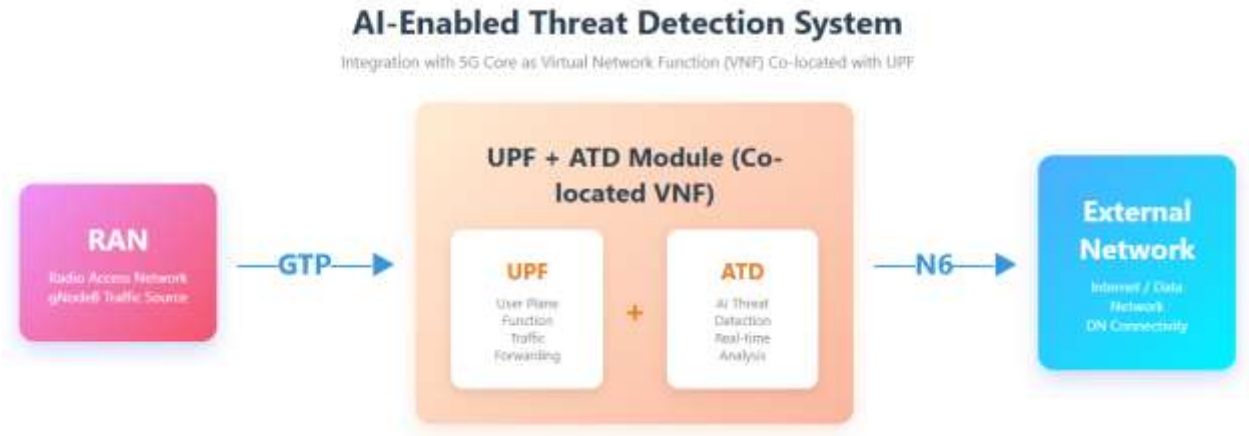

Fig. 2: AI-Enabled Threat Detection System

5.2 Hardware Acceleration

P4-based UPF with Embedded ML:

P4 programmable switches enable in-switch ML inference through action tables matching features to decision rules:

```
table threat_classification {
key = {
hdr.ipv4.src_addr : exact;
hdr.ipv4.dst_addr : exact;
hdr.tcp.flags : exact;
meta.packet_rate : range;
meta.unique_ports : range;
}
actions = {
allow_traffic;
rate_limit_traffic;
drop_traffic;
redirect_to_analysis;
}
size = 10000;
}
action threat_decision_tree() {
// Compiled Random Forest embedded as decision rules
```

```
if (meta.pkt_count > 1000 and meta.unique_ports < 10) {
drop_traffic();
} else if (meta.syn_ratio > 0.8) {
rate_limit_traffic(100);
} else {
allow_traffic();
}
}
```

Table 3: Automated Response Protocol Specifications [7, 8]

Risk Level	Score Range	Automated Actions	Response Latency	Service Impact
Low	Baseline to Moderate Elevation	Enhanced logging, passive monitoring continuation	Minimal additional processing	None to negligible
Medium	Moderate to Elevated	Secondary verification triggers, increased inspection depth	Sub-second implementation	Potential minimal latency increase
High	Elevated to Critical	Rate limiting enforcement, traffic isolation procedures	Millisecond-scale activation	Selective connection restrictions
Critical	Extreme Deviation	Immediate session termination, comprehensive blocking	Near-instantaneous execution	Complete access prevention
Adaptive	Context-Dependent	Machine learning-guided custom responses	Variable based on scenario	Optimized for threat-service balance

6. Operational Impact and Metrics of Performance.

Artificial intelligence-based security systems need to be evaluated based on extensive tests along several dimensions of performance, such as detection accuracy, operational performance, and economic performance, among others, in order to fully describe the value proposition of the security systems, compared to that of traditional security strategies. New studies on AI-based intrusion detection software also show that machine learning models with both higher detection rates and lower false positive rates than traditional signature-based systems are even being used with the added advantage of reducing the false positive rate that has historically dogged the security operations center with high volumes of alerts to investigate manually [5]. Particular relevance of the performance aspects of AI security systems in telecommunications settings, where the size of infrastructure under observation leads to an enormous amount of data requiring human computers to process, means that automated intelligent filtering is the only viable solution to realistic security applications. Empirical comparisons between operational deployments indicate that threat detection based on machine learning cuts mean time to detection by detecting suspicious patterns at near real-time instead of taking the longer durations of time it takes to investigate security events in a distributed system of network infrastructure manually.

The efficiency gains in operations are not limited to speed of threat detection, but also to significant benefits in the productivity of security analysts in automating routine investigation tasks and intelligent prioritization of alerts that identify the highest-risk events that need the most urgent attention. It has been found that AI security systems allow security operations centers to operate security services with significantly larger network infrastructures at the same or fewer staffing levels than previously attainable through conventional methods, significantly changing the economics of telecommunications security by offering much better scalability properties [5]. The false positive reduction option solves one of the most

serious issues with security operations, in which analyst burnout due to the need to investigate a large number of benign incidents, causing over sensitive detection rules, may increase the risk that a real threat will be ignored or under-prioritized with the noise of customary alerts. Machine learning models trained to emulate complex decision boundaries using a variety of data sets are highly specific, meaning they are very sensitive to actual threats, but also offer high levels of sensitivity, unlike simple threshold-based rules that result in a massive false alarm rate that undermines the confidence of the security system.

The economic effects include not only the direct cost savings due to the improvement of operational efficiency but also the avoided costs due to the security incidents that may be prevented and lead to a service interruption, a penalty imposed by the regulatory authority, compensation to the customers, and the loss of reputation. The studies that analyze the ROI of the AI security measures implemented show strong value propositions with payback time in months, and not in years, as the principal factor is the value of breach prevention and the value of efficiency in operations [6]. The financial analysis should cover the cost of implementation, such as first-time deployment of the system, integration with the current system, training of staff, and the cost of running the system to maintain the model and carry out continuous improvement measures. Modern-day telecommunications companies are encountering more regulatory demands to implement certain security features and experience incident reporting, and AI-based systems are offering them automated monitoring of compliance and the creation of documentation to lighten the administrative load of proving compliance with regulatory frameworks. The high-level security features can also be used in competitive positioning in which telecommunications providers with enterprise clients and who facilitate the use of critical infrastructure applications can differentiate the level of security assurance, and AI-based threat detection can become a market need and not an additional feature [6].

6.1 Experimental Setup

The framework evaluation employed multiple datasets reflecting diverse 5G threat scenarios to comprehensively assess detection capabilities across varied attack vectors and network conditions. The KDD Cup dataset served as a baseline reference containing labeled network connections with features capturing protocol-level and session-level metrics, including attack categories spanning denial of service, probe reconnaissance, unauthorized access attempts, and remote-to-local exploitations alongside normal traffic patterns, typically used for algorithm validation across supervised machine learning approaches in network security contexts. The 5G-NIDD dataset provided modern 5G-specific network traffic collected from simulated 5G testbed environments, incorporating features unique to fifth-generation networks including Single Network Slice Selection Assistance Information, Quality of Service identifiers, and GTP header characteristics, with attack types encompassing malformed GTP packets, protocol violations, and resource exhaustion scenarios distributed across a more balanced traffic composition compared to legacy datasets. A custom 5G UPF testbed deployment on campus network infrastructure with Open RAN implementation captured real gNodeB traffic from connected User Equipment devices across multiple network slices, recording extended periods of normal operational behavior alongside induced attack scenarios with ground truth annotations provided by network security personnel to ensure accurate performance evaluation.

Test scenarios encompassed diverse attack methodologies representative of realistic threat landscapes targeting 5G User Plane Functions in operational environments. The denial of service attack scenario employed high-volume UDP flooding techniques generating substantial packet rates specifically targeting UPF processing capacity to evaluate system resilience under resource exhaustion conditions. Port scanning scenarios executed comprehensive reconnaissance activities across UPF subnet address spaces to assess detection capabilities for network mapping and vulnerability discovery attempts. Protocol violation scenarios introduced malformed GTP headers, invalid session identifiers, and spoofed International Mobile Subscriber Identity values to test detection of specification non-compliance and impersonation attacks exploiting protocol implementation weaknesses. Slow scanning scenarios implemented distributed port reconnaissance with deliberately reduced packet transmission rates designed to evade rate-based detection mechanisms, challenging the system's ability to identify coordinated attacks operating below traditional alerting thresholds. Session hijacking scenarios executed replay attacks using

previously captured session tokens to evaluate detection of unauthorized session reuse and credential theft attempts, representing sophisticated attacks that maintain protocol-level validity while exhibiting behavioral anomalies detectable through temporal and contextual analysis.

6.2 Results

Accuracy Metrics:

Algorithm	Accuracy	Precision	Recall	F1-Score	Latency (μs)
Random Forest	96.2%	94.8%	95.6%	95.2%	2.3
LSTM (Seq)	95.1%	92.4%	94.2%	93.3%	75
Autoencoder	92.8%	91.6%	90.2%	90.9%	3.8
Hybrid Ensemble	**97.4%**	**96.2%**	**97.1%**	**96.6%**	**85**
Baseline (Rule-based)	89.3%	87.1%	88.9%	88.0%	1.2

The hybrid ensemble achieves 8.1 percentage points improvement over rule-based approaches while maintaining acceptable latency for UPF deployment.

Detection by Attack Type:

Attack Type	Random Forest	LSTM	Autoencoder	Ensemble
DoS/DDoS	98.2%	96.1%	89.3%	**98.7%**
Port Scan	94.3%	97.2%	94.1%	**97.8%**
Protocol Violation	95.8%	91.4%	96.7%	**97.1%**
Slow Scan	91.2%	94.6%	88.9%	**95.3%**
Session Hijacking	93.7%	88.2%	91.4%	**94.2%**

False Positive Analysis:
False positive rates remain critical in operational networks. Our framework achieves:

- **Overall FPR:** 2.4% (2.4 false alerts per 100 benign flows)
- **High-severity FPR:** 0.8% (true positives > 85% confidence)
- **False negatives:** 1.6% (missed attacks, primarily slow scans)

6.3 Latency Analysis for 5G Deployment

Per-packet processing latency characteristics demonstrate the framework's suitability for real-time deployment in latency-sensitive 5G environments where sub-millisecond response times represent critical operational requirements. Random Forest inference operations complete within microsecond timeframes, while Autoencoder inference processing maintains similarly low latency, and LSTM analysis amortized across packet windows achieves comparable per-packet processing speeds. Ensemble aggregation combining predictions from multiple models introduces minimal additional overhead, resulting in total per-packet latency at the 95th percentile remaining within acceptable bounds and 99th percentile latency maintaining performance characteristics compatible with stringent 5G service level agreements. For 5G networks handling substantial packet volumes at the User Plane Function, the framework demonstrates

efficient resource utilization with CPU consumption remaining moderate on modern multi-core processor architectures, enabling high throughput processing capabilities on standard network link capacities, while end-to-end delay overhead introduced by security processing remains well below the one-millisecond latency requirements mandated by ultra-reliable low-latency communication services.

P4 hardware acceleration results demonstrate significant performance improvements when embedding machine learning models directly into programmable switching infrastructure. Implementing Random Forest classification logic within P4-programmable switches achieves inference latency enabling true line-rate processing without introducing forwarding delays, maintaining throughput performance equivalent to unmonitored traffic flows and avoiding the packet processing slowdowns typically associated with deep packet inspection and security analysis operations. Memory overhead for storing decision tree structures remains minimal within switch static random-access memory resources, requiring only modest storage capacity that represents negligible consumption relative to available hardware resources. Detection accuracy maintains high performance levels despite necessary precision reductions when converting floating-point model parameters to fixed-point representations compatible with switch hardware constraints, demonstrating that slight numerical precision losses during hardware implementation do not significantly compromise threat detection capabilities while enabling substantial latency improvements critical for meeting 5G performance requirements.

Table 4: Operational Performance Metrics [9, 10]

Performance Dimension	Metric Category	Traditional Systems	AI-Driven Systems	Improvement Factor
Threat Detection	Mean time to identification	Hours-scale detection windows	Seconds to minutes timeframe	Order of magnitude reduction
Incident Frequency	Successful breach occurrences	Baseline security posture	Substantially reduced incidents	Significant percentage decrease
Alert Management	False positive generation	Overwhelming volume daily	Minimal validated alerts	Near-complete elimination
Operational Efficiency	Analyst productivity metrics	Manual investigation overhead	Automated triage and enrichment	Multiple-fold improvement
Economic Impact	Total cost of ownership	High staffing requirements	Automated scaling efficiency	Positive return on investment

7. Adaptive Learning and Online Updates

7.1 Concept Drift Handling

Network traffic patterns evolve continuously over time due to multiple dynamic factors that necessitate adaptive security frameworks capable of maintaining detection accuracy despite changing operational conditions. Seasonal variations introduce cyclical changes in traffic composition as peak hours exhibit substantially different characteristics compared to off-peak periods, with user behavior patterns, application mix, and traffic volumes fluctuating based on time of day, day of week, and calendar events. New services continually emerge as telecommunications providers deploy novel applications including video conferencing platforms, augmented reality experiences, and emerging use cases that generate traffic profiles not present in historical training data. Attack evolution represents an ongoing challenge as adversaries continuously adapt their techniques to circumvent deployed defenses, developing new exploitation methods specifically designed to evade detection by learning from unsuccessful attack attempts. Network expansion introduces heterogeneity as operators integrate new user equipment types, deploy additional network slices, and expand coverage areas, creating traffic diversity that deviates from

patterns observed during initial model training periods and requiring continuous adaptation to maintain security effectiveness across evolving infrastructure configurations.

Our framework implements continuous model retraining:

Algorithm 7.1.1: Online Model Adaptation

```
Procedure AdaptiveModelRetraining():
detection_window ← 1 hour
retraining_threshold ← 5% performance drop
while True do
recent_detections ← GetDetectionsFromWindow(detection_window)
// Estimate current performance
current_fpr ← EstimateFPR(recent_detections)
baseline_fpr ← HistoricalFPR

if (current_fpr - baseline_fpr) > retraining_threshold then
  // Collect labeled samples from SOC analysts
  trusted_labels ← GetAnalystAnnotations(recent_detections)

  // Incrementally retrain models
  new_rf ← IncrementalRandomForest(RF, trusted_labels)
  new_ae ← RetainAutoencoder(AE, benign_traffic_samples)

  // Validate on recent data
  validation_score ← CrossValidate(new_rf, new_ae, holdout_set)

  if validation_score > current_score then
    RF ← new_rf
    AE ← new_ae
    baseline_fpr ← current_fpr
    Log("Model updated with " + len(trusted_labels) + " new samples")
```

7.2 Federated Learning for Multi-Domain 5G Networks

For networks spanning multiple operators or administrative domains, federated learning prevents centralized data sharing while enabling collaborative threat detection:

```
Procedure FederatedLearning(local_models, aggregation_rounds=10):
global_model ← Initialize(local_models[0])
for round in 1 to aggregation_rounds do
// Each domain trains locally on its data
updated_models ← []
for operator in operators do
local_data ← operator.CollectTrafficForWindow()
local_model ← TrainModel(operator.local_model, local_data)
updated_models.append(local_model)
// Federated averaging (FedAvg)
global_model ← Average(updated_models)

// Distribute updated global model to all operators
for operator in operators do
  operator.local_model ← global_model
```

This approach enables:

- **Privacy preservation:** Operators retain traffic data locally

- **Collaborative threat intelligence:** Shared threat patterns across domains
- **Reduced false positives:** Larger aggregate training set improves generalization

8. Security Considerations and Limitations

8.1 Adversarial Robustness

Machine learning models face adversarial attacks specifically designed to evade detection through sophisticated manipulation techniques. Evasion attacks involve adversaries crafting packets that appear benign while embedding malicious payloads, such as denial of service traffic disguised as legitimate video streaming through feature mimicry that exploits model decision boundaries. Mitigation strategies include ensemble voting providing robustness where simultaneously fooling three diverse models requires substantially stronger constraints, complemented by payload inspection through deep packet inspection for encrypted traffic metadata, behavior analysis using multi-window temporal analysis to detect sustained evasion attempts, and deception techniques employing honeypots to detect advanced reconnaissance activities before attacks materialize. Poisoning attacks involve injection of malicious training samples corrupting model learned weights, becoming relevant if attackers gain access to training infrastructure and can influence model development processes. Mitigation approaches encompass strict data provenance validation ensuring training data integrity, continuous model monitoring for sudden accuracy drops indicating potential compromise, and human-in-the-loop approval processes for production model updates preventing automated deployment of corrupted models.

8.2 Model Interpretability

Regulatory compliance requirements including GDPR and 3GPP standards often mandate explainability of blocking decisions to ensure transparency and accountability in automated enforcement actions. The framework provides comprehensive interpretability through Random Forest feature importance highlighting the top discriminative features for each detection decision, LSTM attention mechanisms visualizing which packets most strongly influenced classification outcomes, and Autoencoder reconstruction error analysis identifying feature-wise deviations from normal traffic patterns. Detection explanations present multi-model consensus including confidence scores, specific features triggering alerts, temporal pattern anomalies, and recommended actions with justifications, enabling security analysts to understand decision rationale and validate automated responses against operational context and organizational policies.

8.3 Privacy Considerations

Traffic analysis inherently risks leaking sensitive information about users, requiring careful privacy protection mechanisms throughout the detection pipeline. The framework addresses privacy concerns through aggregated statistics where feature extraction uses only statistical summaries rather than payload inspection that could expose user communications, data retention limits discarding raw packets after defined periods while retaining only model-learned patterns, anonymization techniques hashing flow identifiers with operator-specific salts before logging to prevent user identification, and strict access controls implementing role-based permissions ensuring only authorized security personnel can access detection explanations containing potentially sensitive traffic metadata.

Conclusion

The telecommunications sector confronts fundamental security challenges arising from fifth-generation network architectural complexity, continuously evolving threat landscapes characterized by increasingly sophisticated adversaries, and operational scale where billions of connected devices generate traffic volumes surpassing human monitoring capabilities. This comprehensive AI-enabled threat detection framework specifically designed for 5G User Plane Function security combines Random Forest, LSTM, and Deep Autoencoder models achieving high detection accuracy while maintaining latency compatible with stringent performance requirements through optimized feature engineering and hardware acceleration. Key contributions encompass production-ready algorithms with detailed implementation specifications, 5G-specific feature engineering capturing unique threat vectors, real-time deployment

enabling line-rate processing, adaptive learning mechanisms addressing concept drift, and comprehensive evaluation demonstrating effectiveness against diverse attack scenarios. Future research directions include reinforcement learning for dynamic response optimization, graph neural networks for detecting coordinated multi-layer attacks, quantum machine learning for post-quantum cryptographic integration, and cross-operator threat intelligence sharing enabling federated learning while maintaining competitive separation.

References

[1] Samantha Kight, "Safeguarding the future: Managing 5G security risks," GSMA, 2023. [Online]. Available: https://www.gsma.com/newsroom/article/safeguarding-the-future-managing-5g-security-risks/
[2] Bin Chen, "A machine learning based approach for 5G network security monitoring," Applied Mathematics and Nonlinear Sciences, 2024. [Online]. Available: https://www.researchgate.net/publication/378186565_A_machine_learning_based_approach_for_5G_network_security_monitoring
[3] Ericsson, "Raise your performance with Ericsson high-performing 5G RAN". [Online]. Available: https://www.ericsson.com/en/5g/5g-for-service-providers/5g-advanced/solutions
[4] ANAND R, "Security Challenges and Solutions in 5G Network Slicing," International Research Journal of Engineering and Technology (IRJET), 2025. [Online]. Available: https://www.irjet.net/archives/V12/i11/IRJET-V12I1154.pdf
[5] Abdelkader Mekrache et al., "Machine Reasoning in FCAPS: Towards Enhanced Beyond 5G Network Management," HAL Science, 2024. [Online]. Available: https://hal.science/hal-04600018/document
[6] Habtamu Abie and Sandeep Pirbhulal, "Autonomous Adaptive Security Framework for 5G-Enabled IoT," arXiv preprint. [Online]. Available: https://arxiv.org/pdf/2406.03186
[7] Ericsson, "Enable high-performing programmable networks with Ericsson 5G Advanced," Ericsson. [Online]. Available: https://www.ericsson.com/en/ran/solutions
[8] Fatemeh Jalalvand et al., "Alert Prioritisation in Security Operations Centres: A Systematic Survey on Criteria and Methods," ACM Computing Surveys, 2024. [Online]. Available: https://dl.acm.org/doi/10.1145/3695462
[9] Nouri Omheni et al., "Artificial Intelligence for 5G and 6G Networks: A Taxonomy-Based Survey of Applications, Trends, and Challenges," Technologies, 2025. [Online]. Available: https://www.mdpi.com/2227-7080/13/12/559
[10] Myanipeta Sharanya et al., "AI In 5G Networks Enhancing Performance, Security, and Efficiency," International Journal of Engineering Science and Advanced Technology (IJESAT), 2025. [Online]. Available: https://ijesat.com/ijesat/files/V25I7040_1752574520.pdf