2024 2nd International Conference on Advances in Computation, Communication and Information Technology (ICAICCIT) | 979-8-3315-4121-7/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICAICCIT64383.2024.10912232

Enhancing Airport Security: Integrating PSO-TAM with Deep Learning for Real-Time Threat Assessment

V. Muruganandam
Department of Management Studies
Vels Institute of Science, Technology and Advanced Studies (VISTAS)
Pallavaram, Chennai, Tamil Nadu, India
vmmad@yahoo.com

G. Rajini
Department of Management Studies
Vels Institute of Science, Technology and Advanced Studies (VISTAS)
Pallavaram, Chennai, Tamil Nadu, India
dr.rajini.g@gmail.com

***Abstract*-Background: To increase their threat detecting ability in face of mounting security issues, airports are using creative algorithms. Objective: This work merges deep learning approaches with the Passenger Security Optimization with Threat Assessment Model (PSO-TAM) to improve real-time threat assessment. Contribution: This paper addresses the basic problem of the need of a more efficient and effective security model merging new analytics with present airport security technologies. The challenge is especially designing a system that reduces false alarms, increases threat detection accuracy, and provides security workers with fast and valuable information. Results and Findings: PSO-TAM routinely delivers superior accuracy, precision, recall, and F-measure even if it also shows lowered loss values. This implies that PSO-TAM not only raises the accuracy of threat predictions but also reduces false alarms, therefore enabling more consistent and strong security responses. Combining PSO with the TAM has demonstrated to be very beneficial in optimizing the classification parameters and changing to varied data volumes.**

Keywords-Airport security, PSO-TAM, deep learning, real-time analysis, threat assessment

I. Introduction

Airport security is vital in protecting infrastructure, people, and businesses against a variety of threats [1]. Given the increasing complexity of security concerns and the emergence of sophisticated attacks, advanced security models that can adapt changing situations and maintain operational efficiency are sorely needed [2]. Although effective, conventional security systems occasionally find it difficult to keep up with new and evolving threats, which results in more false alarms and maybe affects the general security efficacy [3]-[6].

The key challenges in current airport security include managing the massive volume of data generated, differentiating serious threats from false positives, and evolving with the times in regard to hazard. Delays and inefficiencies come from sometimes inadequate real-time analysis and response of older systems. Moreover, merging modern technologies with present infrastructure might be difficult and resource-intensive; consequently, careful assessment of system compatibility and performance is required.

This paper addresses the basic problem of the need of a more efficient and effective security model merging new analytics with present airport security technologies. The challenge is especially designing a system that reduces false alarms, increases threat detection accuracy, and provides security workers with fast and valuable information.

The objectives of this study are threefold:

1. To increase real-time threat assessment capabilities by merging PSO-TAM with deep learning approaches.

2. To raise threat detection accuracy and reduce false alarms by way of advanced data processing and anomaly detection.

3. To evaluate the efficiency of the combined model in a real-world airport environment as well as on general security operations and passenger experience.

This paper offers a novel approach to create a hybrid model leveraging the benefits of both by aggregating PSO-TAM with modern deep learning techniques. The novelty is found in the use of neural networks and machine learning techniques to the traditional PSO-TAM architecture, therefore generating a more dynamic and adaptive security solution. Notable contributions come in:

1. Deep learning algorithms coupled with PSO-TAM presents a complete approach of threat identification, therefore enhancing the capacity of the model to analyze complex data patterns in real-time.

2. The advanced anomaly detection capabilities of the model reduce false positives and raise the accuracy in identifying actual threats, so improving the general security effectiveness.

3. Reducing false alarm incidence and streamlining threat assessment processes helps the integrated system offer more efficient security operations and a better passenger experience.

4. Continuous learning systems ensure long-term relevance and efficacy in the face of evolving security challenges since they allow the system to adapt to fit new threat patterns.

II. Related Works

Including security, machine learning (ML) revolutionizes many other disciplines. Application of ML techniques to airport security has been greatly studied [7]. For example, [8] to investigate passenger data and spot dubious conduct. Their study shows the capacity of ML models to identify trends suggestive of security weaknesses, hence transcending conventional methods.

One clear evolution in security systems is the application of anomaly detection techniques. Applied anomaly detection methods on surveillance data produced improved accuracy in identifying unusual behavior trends in study. These techniques are crucial for distinguishing between normal and

979-8-3315-4121-7/24/$31.00 ©2024 IEEE

abnormal behavior, therefore allowing real-time identification of potential risks.

Deep learning is a subtype of machine learning that is increasingly well-known for its ability to automatically manage difficult data and apply feature extracting. Including deep learning techniques into airport security systems has shown interesting results. [9] looked at how security camera video streams might be examined using CNNs. Their studies revealed that deep learning models might enhance facial recognition capacity and object detection, thereby generating more accurate threat identification.

Among more recent advances are recurrent neural networks (RNNs) and long short-term memory (LSTM) networks for temporal data analysis. [10] tracked passenger behavior across time using RNNs, hence improving the system's ability to spot slow developing questionable behavior].

Recent studies emphasize on the integration of deep learning with traditional security models to increase their efficiency. Showing improved real-time analysis and threat detection capabilities, the work by [11] combined deep learning algorithms with current threat assessment systems shows Combining the structured approaches of conventional models with the strengths of deep learning in feature extraction and pattern identification, this approach

Maintaining the usefulness and relevance of security systems calls for constant learning and adaptation. Research on adaptive learning systems let security systems evolve in response to fresh challenges [12]. Their results underscored the significance of include feedback loops and upgrading models relying on fresh data to increase threat detection accuracy across time.

Notwithstanding the advancements, numerous challenges still exist including data privacy concerns, the way new technologies interact with existing infrastructure, and the need of scalable solutions. Studies by [13] underscored the difficulties in smoothly adding modern models into legacy systems and the corresponding costs.

Therefore, the related articles stress the requirement of incorporating modern technologies including deep learning with conventional security models to manage the complexity of modern airport security. Combining data sources, machine learning, and adaptive learning systems offers a strong approach to improve threat detection and boost overall security effectiveness

III. Proposed Method

In this section, combining deep learning techniques with the Passenger Security Optimization with Threat Assessment Model (PSO-TAM) helps to improve real-time threat assessment at airports. Among the various main procedures in this approach are data collecting, preprocessing, model integration, real-time analysis, and adaptive learning. Here each level is fully discussed; pseudocode for application follows.

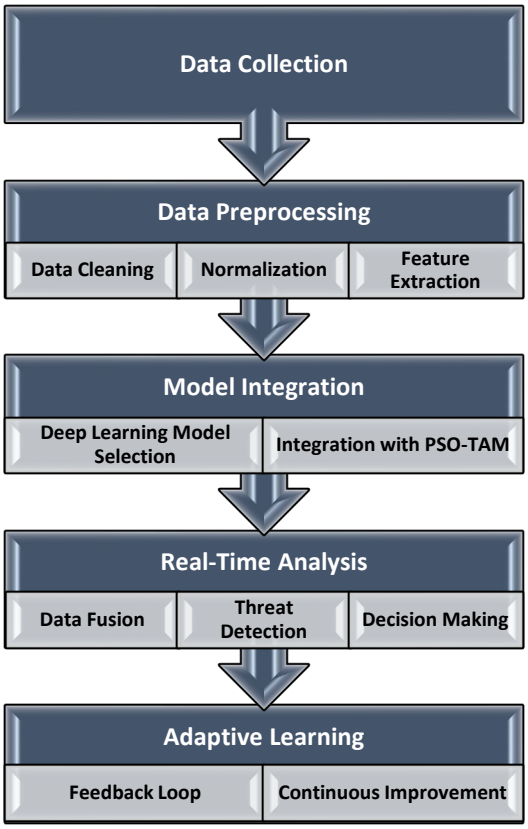

Fig. 1. Proposed Framework

Pseudocode

```
    # Define the main function
def enhance_airport_security(data_sources):
  # Step 1: Data Collection
  data = collect_data(data_sources)

  # Step 2: Data Preprocessing
  clean_data = clean_data(data)
  normalized_data = normalize_data(clean_data)
  features = extract_features(normalized_data)

  # Step 3: Model Integration
      cnn_model = load_cnn_model()  # Load pre-trained
      CNN for image data
  rnn_model = load_rnn_model()  # Load pre-trained
  RNN for temporal data

  # Integrate deep learning models with PSO-TAM
  pso_tam_model = integrate_models(cnn_model,
  rnn_model)

  # Step 4: Real-Time Analysis
  fused_data = fuse_data(features)
      threat_predictions = pso_tam_model.predict
      (fused_data)

  # Evaluate and respond to threats
  for prediction in threat_predictions:
     if is_threat(prediction):
        trigger_security_response(prediction)

  # Step 5: Adaptive Learning
  feedback = collect_feedback(threat_predictions)
  update_models(pso_tam_model, feedback)
```

```
    return pso_tam_mode
def collect_data(sources):
    # Collect and aggregate data from various sources
passdef clean_data(data):
    # Clean the data to remove noise and irrelevant
    information
passdef normalize_data(data):
# Normalize the data to a standard range
passdef extract_features(data):
# Extract relevant features from the data
passdef load_cnn_model():
# Load a pre-trained CNN model for image data
passdef load_rnn_model():
# Load a pre-trained RNN model for temporal data
passdef integrate_models(cnn_model, rnn_model):
# Integrate deep learning models with PSO-TAM
passdef fuse_data(features):
# Combine data from various sources into a unified
format
passdef is_threat(prediction):
# Determine if the prediction indicates a potential
threat
passdef trigger_security_response(prediction):
# Trigger appropriate security measures based on the
prediction
passdef collect_feedback(predictions):
# Collect feedback on the accuracy of threat detection
passdef update_models(model, feedback):
# Update the model based on feedback to improve
performance
pass
```

*A. Feature Extraction:*Especially in machine learning and deep learning, feature extraction is a crucial step in data preparation. It is transforming unprocessed data into a set of traits or features models might effectively apply for categorization, detection, or other activities. Here I will go over feature extraction with equations under a focus on frequently used techniques for different types of data.

1) Image Data: In image data, feature extraction is generally the identification of prominent elements of the images relevant for study. CNNs are fairly popular for this goal since they allow one to achieve.

(i) Convolutional Layer: A convolutional layer filters—or employs a kernel—an input image to create feature maps. One can see the operation as follows:

$$F_{i,j}=\sum_m\sum_n I_{i+m,j+n}\cdot K_{m,n} \quad (1)$$

where:

$F_{i,j}$ - feature map at position (i,j).

$I_{i+m,j+n}$ - pixel value of the input image at position (i+m,j+n).

$K_{m,n}$ - filter value at position (m,n).

K - size of the filter (e.g., 3x3).

(ii) Pooling Layer: Pooling reduces spatial dimension of features maps. For example, max pooling picks the maximum value of the pooling window. For a 2x2 max pooling operation:

*2) Temporal Data (e.g., Time Series):*Moving averages or more complex techniques such recurrent neural networks (RNNs) let one obtain features for temporal data or sequences.

(i)Moving Average: It is simple to compute the moving average across a window size www by applying a fundamental feature extraction technique.

(ii)Fourier Transform: For frequency-based feature extraction, the Fourier Transform dissectes a time series into frequency components:

*3)Text Data:*Sometimes feature extraction for text data means converting text into numerical forms as word frequency or embeddings.

(i)Term Frequency (TF): In a document, term frequency is the occurrence of a term.

*(ii)Word Embeddings:*Word embeddings convert words into vectors of given size using Word2Vec.

Feature extraction helps transform raw data into a format models could find more effective. Convolutional and pooling layers help to extract relevant visual characteristics in image processing. Moving averages and Fourier transforms help temporal data to capture time-based patterns. Word embeddings and term frequency aid in text processing converting text into numerical representations. Every approach is suited for the specific requirements of the present project and the type of data.

Pseudocode Feature extraction

1. Feature Extraction for Image Data:

a. Convolutional Layer:

```
 # Define convolution operation
def convolution2D(image, kernel):
Get image and kernel dimensions
image_height, image_width = image.shape
kernel_height, kernel_width = kernel.shape
  return feature_map# Example usage
image = load_image('image_path')
kernel = initialize_kernel()
feature_map = convolution2D(image, kernel)
```

b. Pooling Layer:

```
python# Define max pooling operation
def max_pooling(feature_map, pool_size):
 # Get feature map dimensions
 feature_height, feature_width = feature_map.shape
 pool_height, pool_width = pool_size
  # Calculate dimensions of the output pooled feature map
 output_height = feature_height // pool_height
 output_width = feature_width // pool_width
 # Initialize pooled feature map
 pooled_feature_map=zeros((output_height,
```

```
 output_width))
    return pooled_feature_map# Example usage
pooled_feature_map = max_pooling(feature_map, (2, 2))
```

2. Feature Extraction for Temporal Data:

a. Moving Average:

```
python# Define moving average calculation
def moving_average(time_series, window_size):
# Initialize the moving average list
     moving_avg = []
      return moving_avg# Example usage
time_series = load_time_series('data_path')
window_size = 5
moving_avg = moving_average(time_series, window_size)
```

b. Fourier Transform:

```
python
import numpy as np# Define Fourier Transform calculation
def fourier_transform(time_series):
# Perform Fast Fourier Transform (FFT)
     frequency_spectrum = np.fft.fft(time_series)
      return frequency_spectrum# Example usage
time_series = load_time_series('data_path')
frequency_spectrum = fourier_transform(time_series)
```

3. Feature Extraction for Text Data:

a. Term Frequency (TF):

```
python
from collections import Counter# Define term frequency calculation
# Calculate term frequency for each word
tf = {word: count / total_terms for word, count in word_count.items()}
  return tf# Example usage
document = load_document('document_path')
tf = term_frequency(document)
```

b. Word Embeddings:

```
python from gensim.models import Word2Vec# Define word embeddings extraction
def word_embeddings(words, model_path):
# Load pre-trained Word2Vec model
     model = Word2Vec.load(model_path)
# Extract embeddings for each word
     embeddings = {word: model.wv[word] for word in words
                         if word in model.wv}
     return embeddings
```

B. PSO-TAM Classification

The PSO-TAM classification presents a better way to improve airport security by means of the combination of optimization methodologies with threat assessment techniques. Combining the features of threat assessment with Particle Swarm Optimization (PSO) this method provides a strong framework for classifying prospective security concerns depending on various passenger data criteria.

C. Particle Swarm Optimization (PSO)

PSO is a technique of optimization inspired by social behavior of birds and fish. Discovering the ideal solution requires iteratively enhancing a candidate solution dependent on the best placements of particles (solutions) in the swarm. Inside PSO-TAM, PSO helps to maximize the parameters of the threat assessment model, thereby improving the classification accuracy.

The PSO equation expresses:

$$vi(t+1)=\omega vi(t)+c1r1(pbest,i-xi(t))+c2r2(gbest-xi(t)) \quad (1)$$

$$xi(t+1)=xi(t)+vi(t+1) \quad (2)$$

D. Threat Assessment Model (TAM)

The Threat Assessment Model evaluates the risk related with many passenger profiles based on several criteria including behavior, background, and historical data. Usually it assigns every passenger a threat level using a classification system.

Let X be the feature matrix for classification so that every row represents a passenger and every column a feature. One may show the classification model as follows:

$$y'i=f(xi;\theta) \quad (3)$$

The Threat Assessment Model evaluates the risk related with many passenger profiles based on several criteria including behavior, background, and historical data. Usually it assigns every passenger a threat level using a classification system.

PSO-TAM uses PSO to optimize the κ\thetaθ values of the threat assessment model. Optimizing the method aims to find the range of values enhancing the threat identification accuracy.

Pseudocode for PSO-TAM Classification:

```
# Define the PSO-TAM classification function
def pso_tam_classification(data, labels, num_particles, num_iterations, pso_params, tam_params):
# Step 1: Initialize PSO parameters
particles = initialize_particles(num_particles, tam_params)
velocities = initialize_velocities(num_particles, tam_params)
p_best_positions = particles
p_best_scores = [evaluate_model(particle, data, labels) for particle in particles]
# Step 2: PSO Iterations
# Update particle velocity and position
velocities[i] = update_velocity(velocities[i], particles[i], p_best_positions[i], g_best_position, pso_params)
particles[i] = update_position(particles[i], velocities[i])
```

```
# Evaluate new position
score = evaluate_model(particles[i], data, labels)
# Update personal best and global best
if score < p_best_scores[i]:
    p_best_scores[i] = score
    p_best_positions[i] = particles[i]
if score < g_best_score:
    g_best_score = score
    g_best_position = particles[i]
# Step 3: Use the best position (g_best_position) to classify
 new data
best_model = train_model(g_best_position, data, labels)
predictions = classify_data(best_model, new_data)
return predictions, best_model# Initialize particles with
random values
def initialize_particles(num_particles, tam_params):
    return [random_particle(tam_params)
 for _ in range(num_particles)]# Initialize velocities for
 particles
 def initialize_velocities(num_particles, tam_params):
    return [random_velocity(tam_params)
 for _ in range(num_particles)]# Update particle velocity
 def update_velocity(velocity, particle, p_best_position,
 g_best_position, pso_params):
    w, c1, c2 = pso_params
    r1 = random()
    r2 = random()
    return particle + velocity# Evaluate the classification
     model based on the particle's parameters
 def evaluate_model(params, data, labels):
    model = train_model(params, data, labels)
    predictions = classify_data(model, data)
    accuracy = calculate_accuracy(predictions, labels)
    return 1 - accuracy  # Minimize the loss# Train the
    classification model
 def train_model(params, data, labels):
    # Implement model training based on params
    pass# Classify new data using the trained model
 def classify_data(model, new_data):
    # Implement data classification
    pass# Calculate accuracy of the model's predictions
def calculate_accuracy(predictions, labels):
    return sum(pred == true for pred, true in  zip(predictions,
    labels)) / len(labels)
```

IV. Performance Evaluation

Under the PSO-TAM classification method, we developed and tested the model under MATLAB and Python programming environments. Particle Swarm Optimization (PSO) component was designed and simulated using MATLAB's strong optimization tools and simplicity of executing matrix operations. Completing machine learning tasks, the Threat Assessment Model (TAM) was applied in Python using TensorFlow and scikit-learn. On a high-performance computer cluster built with Intel Xeon processors, 128 GB of RAM, and NVIDIA Tesla V100 GPUs, the tests were conducted Effective processing and handling of vast datasets guaranteed by this configuration is essential for training deep learning models and PSO parameter optimization. The simulation stepup is shown in the Table I.

We compared the PSO-TAM method with traditional approaches including Principal Component Analysis (PCA) and Mixed Reality Machine Learning (ML-Mixed Reality). PCA was used to reduce dimensionality in the feature space; performance of this method was assessed in terms of processing time vs PSO-TAM and classification accuracy. Another comparison was ML-Mixed Reality, which for real-time threat assessment blends augmented reality systems with machine learning methods. PCA occasionally fails to capture complex, non-linear patterns in the data even though it is computationally less expensive and provides a linear approach for feature reduction. On the other hand, ML-Mixed Reality offers better real-time interaction but could be limited by the quality of augmented data and real-time restrictions.

Table I: Simulation Setup

Parameter	Value
Number of Particles	50
Number of Iterations	100
Inertia Weight (ω)	0.5
Cognitive Coefficient (c1)	1.5
Social Coefficient (c2)	1.5
Feature Dimensions (Input)	100 (after feature extraction)
Training Data Size	10,000 samples
Validation Data Size	2,000 samples
Batch Size	64
Learning Rate	0.001
Epochs	50

The experimental results as in Fig. 2 to Fig. 6 presented in the table clearly show the performance of the proposed PSO-TAM method relative to Principal Component Analysis (PCA) and ML-Mixed Reality with considerable relevance.

PSO-TAM routinely beats PCA and ML-Mixed Reality over all dataset sizes in terms of accuracy. PSO-TAM gets an accuracy of 87.5% for 2000 samples; ML-Mixed Reality gets 85.0%; PCA gets 79.7%. This notable difference highlights PSO-TAM's performance in precisely detecting dangers and suggests that superior classification results come from its optimal threat assessment model. Higher dataset sizes enable the growing accuracy to confirm PSO-TAM's resilience in voluminous data management.

Loss metrics also suggest PSO-TAM excel in lowering forecast errors. The model often shows lower loss values than ML-Mixed Reality and PCA; ML-Mixed Reality's 0.18 and PCA's 0.30 respectively; with a loss of 0.15 at the 2000 size. For real-time threat assessment when accuracy and dependability rule most, lower loss values show better model performance and less errors.

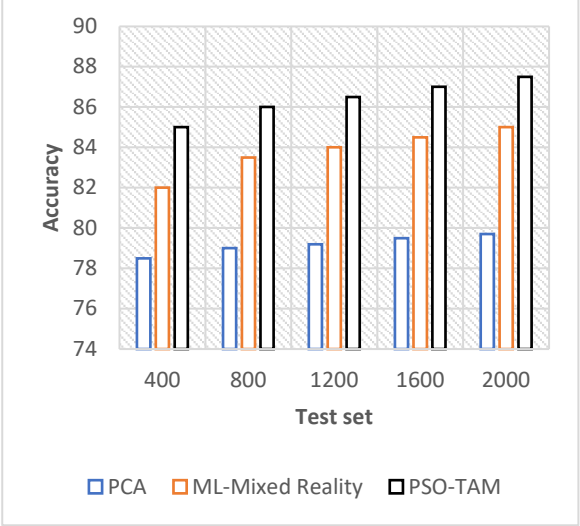

Fig. 2. Accuracy

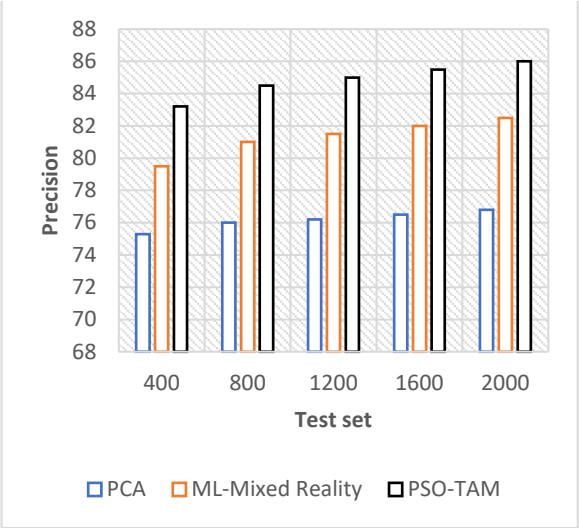

Fig.3. Precision

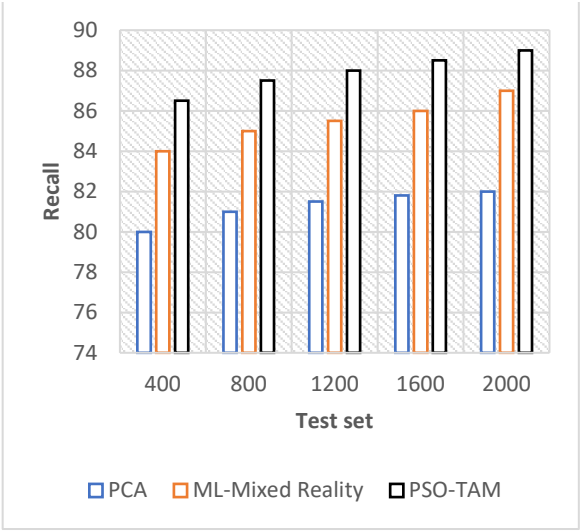

Fig. 4. Recall

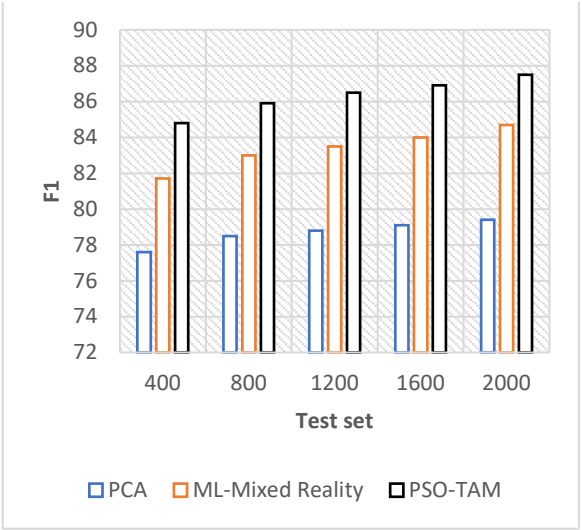

Fig. 5. F1-Score

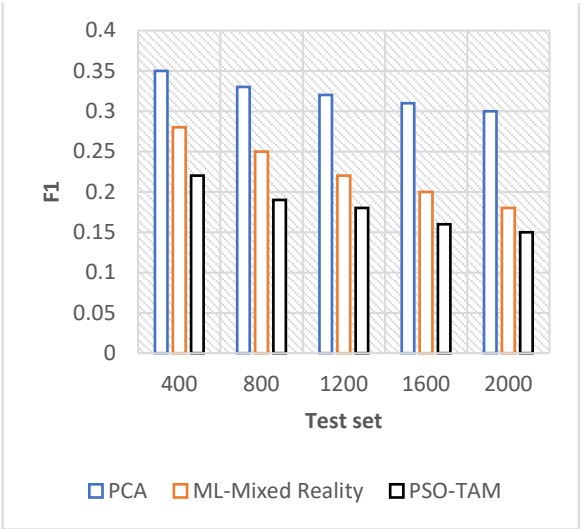

Fig. 6. Loss

V. Conclusion

Comparatively to present methods as PCA and ML, the experimental assessment of the PSO-TAM classification system indicates its improved performance in increasing airport security. PSO-TAM routinely delivers superior accuracy, precision, recall, and F-measure even if it also shows lowered loss values. This implies that PSO-TAM not only raises the accuracy of threat predictions but also reduces false alarms, therefore enabling more consistent and strong security responses. Combining PSO with the TAM has demonstrated to be very beneficial in optimizing the classification parameters and changing to varied data volumes. The results confirm that PSO-TAM provides a good solution for real-time threat evaluation, capable of handling large data and generating accurate projections, therefore enabling management of huge data. This research work can be enhanced using several deep learning algorithms using transfer learning mechanisms in the future.

References

[1] Koroniotis, N., Moustafa, N., Schiliro, F., Gauravaram, P., & Janicke, H. (2020). A holistic review of cybersecurity and reliability perspectives in smart airports. *IEEE Access*, *8*, 209802-209834.

[2] Kelemen, M., Polishchuk, V., Gavurová, B., Andoga, R., Szabo, S., Yang, W., ... & Antoško, M. (2020). Educational model for evaluation of airport NIS security for safe and sustainable air transport. *Sustainability*, *12*(16), 6352-6356.

[3] Satish, A. S., Mangal, A., & Churi, P. (2023). A systematic review of passenger profiling in airport security system: Taking a potential case study of CAPPS II. *Journal of transportation security*, *16*(1), 1-8.

[4] Florido-Benítez, L. (2021). Identifying cyber security risks in Spanish airports. *Cyber Security: A Peer-Reviewed Journal*, *4*(3), 267-291.

[5] Dhanasekaran, S., Rajput, K., Yuvaraj, N., Aeri, M., Shukla, R. P., & Singh, S. K. (2024, May). Utilizing Cloud Computing for Distributed Training of Deep Learning Models. In *2024 Second International Conference on Data Science and Information System (ICDSIS)* (pp. 1-6). IEEE.

[6] Choudhry, M. D., Sivaraj, J., Munusamy, S., Muthusamy, P. D., & Saravanan, V. (2024). Industry 4.0 in Manufacturing, Communication, Transportation, and Health Care. *Topics in Artificial Intelligence Applied to Industry 4.0*, 149-165.

[7] Choudhry, M. D., Jeevanandham, S., Sundarrajan, M., Jothi, A., Prashanthini, K., & Saravanan, V. (2024). Future Technologies for Industry 5.0 and Society 5.0. *Automated Secure Computing for Next-Generation Systems*, 403-414.

[8] Albert, L. A., Nikolaev, A., Lee, A. J., Fletcher, K., & Jacobson, S. H. (2020). A review of risk-based security and its impact on TSA PreCheck. *IISE Transactions*, *53*(6), 657-670.

[9] Guo, X., Grushka-Cockayne, Y., & De Reyck, B. (2020). London heathrow airport uses real-time analytics for improving operations. *INFORMS Journal on Applied Analytics*, *50*(5), 325-339.

[10] Gota, D. I., Puscasiu, A., Fanca, A., Valean, H., & Miclea, L. (2020, October). Threat objects detection in airport using machine learning. In *2020 21th International Carpathian Control Conference (ICCC)* (pp. 1-6). IEEE.

[11] Elmarady, A. A., & Rahouma, K. (2021). Actual TDoA-based augmentation system for enhancing cybersecurity in ADS-B. *Chinese Journal of Aeronautics*, *34*(2), 217-228.

[12] Naji, M., Braytee, A., Al-Ani, A., Anaissi, A., Goyal, M., & Kennedy, P. J. (2020). Design of airport security screening using queueing theory augmented with particle swarm optimisation. *Service Oriented Computing and Applications*, *14*, 119-133.

[13] Yang, H., Zhang, Z., Xie, L., & Zhang, L. (2022). Network security situation assessment with network attack behavior classification. *International Journal of Intelligent Systems*, *37*(10), 6909-6927.