International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(17s) 2026 pp. 114-135
https://doi.org/10.70917/ijcisim-2026-4709
Article

MULTI-VIEW TEMPORAL CONTRASTIVE LEARNING NETWORK FOR ADAPTIVE REAL-TIME PHISHING DETECTION IN DYNAMIC AI-DRIVEN CYBERSECURITY ENVIRONMENTS

Arockia Amutha F.[1], M. Ramesh Kumar[2]

[1] Research Department of Computer Science, Government Arts College (Autonomous), Nandanam, Chennai – 600035, Tamil Nadu, India.
Email: amuthafmm@gmail.com
ORCID: 0009-0008-0392-7633
[2]PG and Research Department of Computer Science, Government Arts College (Autonomous), Nandanam, Chennai – 600035, Tamil Nadu, India.
Email: proframeshkumar@gmail.com

Abstract: The increasing dependence on digital communication, cloud services, and online financial transactions has intensified the frequency of phishing attacks, which is creating significant challenges for an intelligent cybersecurity system. This study is proposing a Multi-View Temporal Contrastive Learning Network (MTCLN) for the phishing detection in dynamic AI-driven cybersecurity environments. The proposed framework is combining the temporal feature learning module with the contrastive representation learning strategy in capturing the evolving phishing characteristics from the heterogeneous data sources. This includes URL structures, webpage content, DNS attributes, and behavioral metadata. A lightweight gated temporal encoder is extracting the sequential attack patterns, while a contrastive feature alignment module is improving the discrimination between the legitimate and phishing instances under the changing attack scenarios. Further, an adaptive confidence calibration mechanism is adjusting the classification threshold in accordance with the recent detection feedback. This is allowing robust operation without requiring the complete model retraining. The proposed method is presenting improved phishing detection capability by achieving an accuracy of 88.4%, precision of 87.8%, recall of 86.9%, and F1-score of 87.3%, which is performing better than the CNN-Based Phishing Detection Model, LSTM-Based Phishing Detection Model, and Attention-Based Transformer Phishing Detection Model. The framework is reducing the detection latency to 31 ms, when it is compared with the 37 ms, 40 ms, and 42 ms obtained by the Conventional methods.
Keywords: NA

1. Introduction

Recent cybersecurity studies are emphasized the importance of artificial intelligence (AI)-driven solutions for identifying malicious patterns and then it is improving real-time threat response capabilities. [1] is showing that machine learning-based security approaches are providing effective classification capabilities by learning complex attack patterns from the large-scale network data. [2] is showing that deep learning models are improving the phishing detection by automatically extracting high-level representations from the URLs and webpage content without depending on manually designed rules. [3] is presenting that feature learning from the heterogeneous cybersecurity

data sources is improving the capability of detection systems in identifying previously unseen attack behaviours. [4] explains that intelligent cybersecurity frameworks are requiring continuous adaptation mechanisms because modern attacks is evolving rapidly across the different digital environments. [5] reports that AI-enabled security models are improving the automation and are reducing dependency on human intervention, thereby it is supporting faster responses against the emerging phishing threats.

Despite these advancements, several challenges tends to remain in developing reliable real-time phishing detection systems. One major challenge is the dynamic nature of phishing attacks, where attackers frequently modify domain structures, webpage layouts, and malicious content patterns to bypass Conventional detection mechanisms. The conventional models that is often experiencing performance degradation when exposed to newly generated phishing samples that differ from the training distributions. [6] is identifying that static feature-based detection approaches are limited capability in handling zero-day phishing attacks because it is depending on predefined characteristics. [7] discusses that deep learning-based models are requiring extensive computational resources and frequently challenge complete retraining when new attack patterns appear. [8] is indicating that maintaining a lower detection latency while achieving high classification accuracy is remaining as a major challenge for real-time cybersecurity applications. [9] explains that Conventional phishing detection systems are facing difficulties in combining multiple data perspectives, which is including URL information, webpage semantics, network behaviour, and temporal attack patterns.

[10] is identifying that conventional machine learning models are suffering from the reduced adaptability because its feature representations tends to remain unchanged after deployment. [11] is showing that convolutional neural network-based phishing detectors achieve effective classification but are lacking sufficient temporal awareness for evolving attack behaviours. [12] reports that recurrent neural network-based models capture sequential patterns but encounter difficulties in processing heterogeneous cybersecurity information. [13] is emphasizing that Conventional adaptive detection frameworks still are requiring frequent model updates, increasing computational overhead and maintenance complexity. [14] concludes that current intelligent phishing detection methods are requiring improved self-learning capabilities to achieve stable performance in real-time environments.

Recent related works are attempting to address these limitations through the advanced learning architectures, which is including transformer-based models, graph neural networks, and an adaptive reinforcement learning strategies. However, transformer-based phishing detectors focus on contextual feature extraction and are lacking continuous adaptation after deployment. Graph-based approaches are improving the relationship modelling between the domains and URLs, but its effectiveness depends on accurate graph construction and periodic updates. Reinforcement learning-based security models are providing adaptive decision-making capabilities; however, it tends in introducing additional computational complexity and are requiring extensive feedback interactions before achieving optimal detection performance.

The primary objective of this research is in developing an adaptive phishing detection framework that is combining the multiple cybersecurity data views, which is including URL characteristics, webpage content patterns, DNS information, and behavioural attributes.

The novelty of this research is the combination of temporal representation learning and contrastive feature optimization for adaptive phishing detection. A Multi-View Temporal Contrastive Learning Network (MTCLN) is proposed for phishing detection in dynamic cybersecurity environments.

The major contributions of this research are summarized as follows. First, a MTCLN architecture is introducing in extracting adaptive phishing characteristics from the heterogeneous cybersecurity data sources and are improving the detection reliability under the dynamic attack conditions. Second, an adaptive confidence calibration strategy is developed in supporting the real-time deployment by reducing detection latency and maintaining stable performance without frequent complete model retraining.

2. Literature Survey

2.1 Machine Learning-Based Phishing Detection Approaches

Recent phishing detection research is increasingly focused on machine learning (ML) techniques in improving the automated identification of malicious websites and fraudulent communication patterns. Early ML-based approaches is depending on handcrafted features extracted from the URLs, domain information, webpage characteristics, and user behaviour patterns. These methods is showing reasonable classification capability; however, its performance is highly dependent on feature quality and dataset representation. [15] is presenting a machine learning-driven phishing detection model that is extracting URL-based lexical features, host-related attributes, and webpage indicators to classify malicious and legitimate websites. The study is showing that feature-based ML models achieve effective detection accuracy but are experiencing limitations when phishing attackers modify URL structures or are introducing new attack patterns. in improving the generalization, [16] proposes an ensemble learning approach that is combining multiple classifiers for phishing identification. The method is improving classification stability by reducing individual model bias; however, it is requiring extensive feature engineering and periodic updates in maintaining performance against the emerging phishing techniques.

2.2 Deep Learning-Based Feature Representation for Phishing Detection

The limitations of conventional ML methods are encouraged researchers to adopt deep learning (DL) models that automatically learn complex feature representations from the large-scale cybersecurity datasets. Deep learning approaches are reducing dependence on manually selected features and are providing improved capability for detecting hidden relationships within the phishing characteristics. [17] is introducing a convolutional neural network (CNN)-based phishing detection framework that is learning discriminative patterns from the URL sequences and webpage information. The CNN model is capturing local feature dependencies, and it is improving classification performance when it is compared with the conventional classifiers. However, the approach that is focussing on spatial feature extraction and does not sufficiently consider temporal variations in phishing campaigns.

To address sequential dependency limitations, [18] develops a recurrent neural network (RNN)-based detection model that analyses sequential URL patterns and content behaviours. The method is improving detection capability by modelling long-term dependencies in phishing data. Nevertheless, conventional RNN architectures are suffering from the training difficulties and limited scalability when processing large and changing cybersecurity datasets. [19] proposes a long short-term memory (LSTM)-based phishing detection approach to overcome the limitations of conventional RNN models. The LSTM architecture is capturing temporal relationships and it is improving detection performance for dynamically generated phishing URLs.

2.3 Phishing Detection Models

[20] is introducing an attention-enhanced transformer model for phishing classification, where the attention mechanism is identifying important feature interactions and it is improving detection accuracy. The study is showing that transformer architectures is performing better than conventional deep learning models in understanding complex phishing patterns. [21] proposes an efficient transformer framework that is combining the feature compression and attention optimization for phishing detection. [22] develops a graph neural network (GNN)-based phishing detection framework that models relationships between the malicious domains and associated network entities. [23] proposes an online learning framework that updates model parameters using newly observed phishing samples. The approach is reducing the dependency on complete retraining and it is improving adaptability against the emerging attacks. [24] is presenting a hybrid deep learning framework combining CNN feature extraction with the attention-based classification. However, the approach is requiring high computational resources and lacks a dedicated mechanism for continuous adaptation. [25] is introducing an adaptive cybersecurity framework that is combining deep learning with the reinforcement-based optimization for real-time phishing detection. The model updates detection policies, which is based on classification feedback, which is improving resilience against the changing attack behaviours. Although

the framework is providing adaptive decision-making capability, reinforcement-based learning is introducing additional computational overhead and it is requiring sufficient feedback samples for effective optimization.

Table 1: Comparative Summary of Conventional Phishing Detection Techniques

Reference	Technique / Model	Main Objective	Feature Used	Major Advantages	Limitations
[15]	Machine Learning-Based Classifier	Detect phishing URLs using extracted features	URL lexical, domain, webpage features	Simple implementation and effective classification	Requires manual feature engineering
[16]	Ensemble Learning Model	Improve classification stability	URL and host-based attributes	Reduces classifier bias and improving robustness	High dependency on feature quality
[17]	CNN-Based Detection Model	Automatically learn phishing patterns	URL sequences and webpage content	Extracts high-level feature representations	Limited temporal learning capability
[18]	RNN-Based Framework	Analyse sequential phishing behaviour	URL and content sequences	Captures sequential dependencies	Training complexity and scalability issues
[19]	LSTM-Based Model	Improve long-term pattern learning	Temporal URL and webpage patterns	Handles sequential attack evolution	Requires periodic retraining
[20]	Attention Transformer Model	Capture contextual phishing relationships	URL tokens and semantic features	Improved global feature understanding	High computational requirement
[21]	Lightweight Transformer	Reduce transformer complexity	Compressed phishing representations	Faster inference and reduced overhead	Limited online adaptation
[22]	Graph Neural Network	Model relationships among the cyber entities	Domain, URL, IP relationships	Detects hidden structural attack patterns	Graph updating complexity
[23]	Online Adaptive Learning	Enable continuous model updates	Real-time phishing samples	Supports evolving attack detection	Feedback errors affect learning
[24]	CNN-Attention Hybrid Model	Combine local and global feature learning	URL and webpage features	Improved detection accuracy	Increased model complexity
[25]	Deep Reinforcement Learning Framework	Adaptive phishing decision-making	Multi-source cybersecurity data	Supports dynamic policy optimization	Requires extensive feedback and computation

Although conventional studies achieve significant improvements in phishing detection accuracy, it focusses on classification performance using fixed datasets. Limited attention is provided to continuous adaptation, lightweight

online updating, and efficient real-time inference. Therefore, an adaptive phishing detection framework that is combining temporal feature learning, representation optimization, and dynamic decision adjustment is is requiring to address evolving cybersecurity threats.

3. Proposed Method

The proposed MTCLN is developed to achieve real-time phishing detection by learning evolving attack characteristics from the heterogeneous cybersecurity data sources. Unlike conventional phishing detection approaches that is depending on fixed feature representations and offline retraining, the proposed framework is combining the multi-view feature extraction, temporal dependency modelling, contrastive representation optimization, and an adaptive confidence calibration. The entire architecture is consisting of five major phases as in figure 1: (1) multi-source phishing data acquisition and preprocessing, (2) multi-view feature representation, (3) temporal contrastive learning-based phishing analysis, (4) adaptive confidence calibration, and (5) real-time threat prediction and model updating. The complete workflow is designed in improving the detection robustness against the changing phishing strategies while it is maintaining a lower computational overhead.

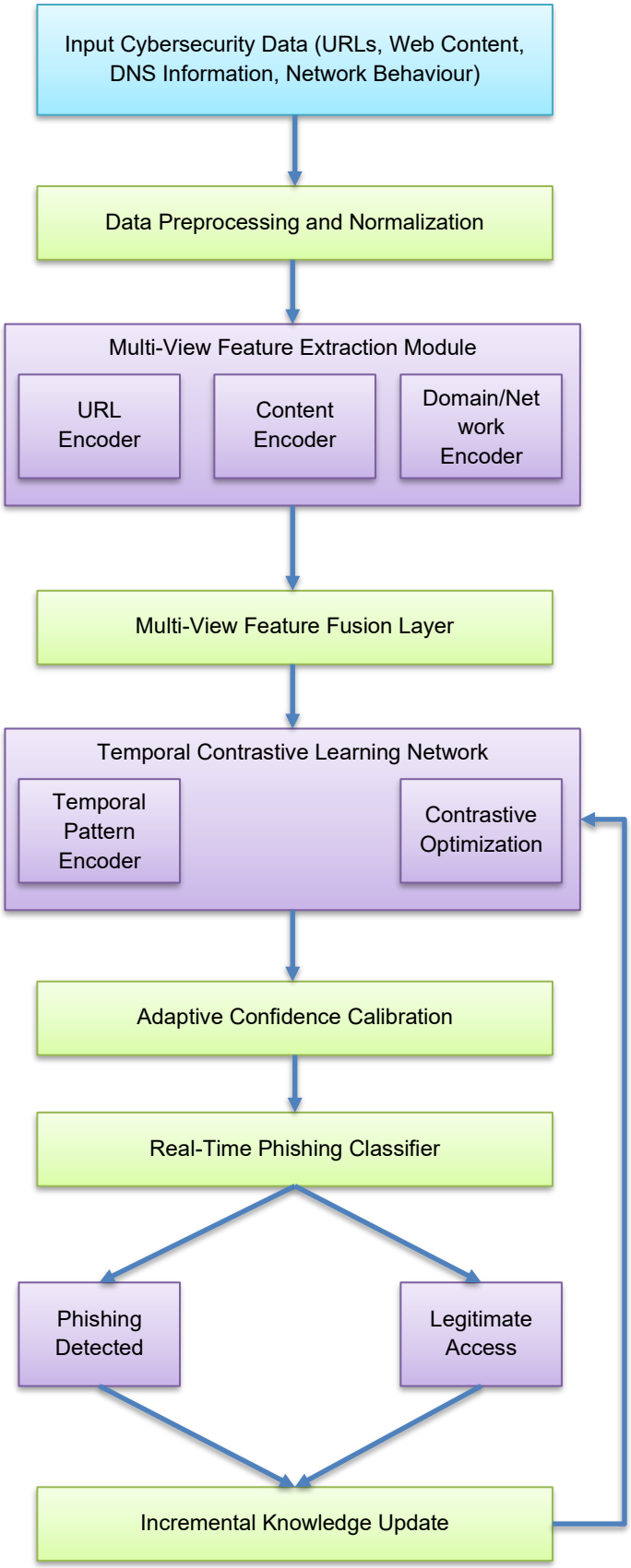
Input Cybersecurity Data (URLs, Web Content, DNS Information, Network Behaviour)
Data Preprocessing and Normalization
Multi-View Feature Extraction Module
URL Encoder
Content Encoder
Domain/Network Encoder
Multi-View Feature Fusion Layer
Temporal Contrastive Learning Network
Temporal Pattern Encoder
Contrastive Optimization
Adaptive Confidence Calibration
Real-Time Phishing Classifier
Phishing Detected
Legitimate Access
Incremental Knowledge Update

Figure 1: MTCLN-Based Real-Time Phishing Detection

Algorithm 1: MTCLN-Based Real-Time Phishing Detection

Input:

Training dataset D = {(xi, yi)}

where xi is representing multi-view cybersecurity data

yi is representing phishing or legitimate label

Output:

Updated phishing detection model M

1: Initialize multi-view encoders Eurl, Econtent, Edomain

2: Initialize temporal encoder T

3: Initialize contrastive learning module C

4: Initialize confidence calibration module A

5: For each training sample xi in D do

6: is extracting URL features Xu from the xi

7: is extracting webpage features Xc from the xi

8: is extracting domain and behaviour features Xd from the xi

9: Normalize all feature representations

10: are generating individual embeddings:

Zu = Eurl(Xu)

Zc = Econtent(Xc)

Zd = Edomain(Xd)

11: Fuse multi-view representations:

Zf = Fusion(Zu, Zc, Zd)

12: are generating temporal representation:

Zt = T(Zf)

13: Construct positive and negative sample pairs

14: Optimize representation using contrastive loss:

Lc = Contrastive(Zt)

15: Update encoder parameters

16: End For

17: For each incoming phishing request xnew do

18: is extracting multi-view features

19: are generating temporal representation Znew

20: Calculate phishing probability P(y|Znew)

21: Adjust decision threshold using adaptive calibration module

22: If P(y|Znew) ≥ threshold then

23: Classify as Phishing
24: Else
25: Classify as Legitimate
26: End If
27: Store new sample information for incremental update
28: End For
29: Return updated model M

3.1: Multi-Source Phishing Data Acquisition and Preprocessing

The first stage of the proposed MTCLN that is focussing on collecting, combining, and preparing heterogeneous cybersecurity information is requiring for adaptive phishing detection. Modern phishing attacks are generated through the multiple attack vectors, where malicious characteristics are distributed across the URLs, webpage structures, domain registration information, and network communication behaviours. Therefore, depending on a single data source are limiting the capability of a detection model to identify complex phishing strategies.

The complete input sample representation is formulated as:

$$X_i = \{X_i^u, X_i^c, X_i^d, X_i^b\}, i = 1,2,\ldots,N$$

Where X_i is representing the complete cybersecurity sample, X_i^u is representing the URL-related attributes, X_i^c is representing the webpage content attributes, X_i^d is representing the domain-related attributes, X_i^b is representing the behavioural attributes, and N is representing the total number of collected samples.

Since cybersecurity data are collected from the multiple sources, the initial dataset often contains missing values, redundant records, inconsistent formats, and noisy observations. These issues negatively influence model learning because irrelevant patterns tends to be incorrectly interpreted as phishing characteristics. Therefore, the preprocessing module is performing data cleaning operations before feature extraction. Duplicate samples are removed using similarity-based filtering, incomplete records are eliminated or reconstructed using statistical replacement strategies, and inconsistent data formats are converted into a unified representation. The preprocessing process is allowing that all input samples maintain similar structural characteristics before entering the feature learning stage.

The data cleaning operation is mathematically represented as:

$$D_p = \{X_i \mid X_i \in D_o,\ C(X_i) = 1,\ R(X_i) = 0\}$$

whereD_p is representing the processed datasetD_o, which is representing the original collected dataset$C(X_i)$, which is indicating the validation condition of sample$X_i, R(X_i)$ and it is indicating whether the sample contains redundant or corrupted information. A value of one is indicating that the sample satisfies the is requiring quality conditions, whereas a value of zero is representing an invalid sample.

After cleaning, the extracted numerical attributes are normalized in confirming that features with the different scales is contributing equally during model training. For example, domain age values, URL length values, and network frequency values tends in significantly different numerical ranges. Directly using these values can cause certain high-range features to dominate the learning process. The proposed method applies min-max normalization in transforming the all continuous attributes into a common range.

The normalization process is expressed as:

$$\hat{x}_{ij} = \frac{x_{ij} - \min(x_j)}{\max(x_j) - \min(x_j) + \epsilon}$$

where$\hat{x}_{ij}$ is representing the normalized value of the j^{th}feature for the i^{th}samplex_{ij}, which is representing the original feature value, $\min(x_j)$and $\max(x_j)$ is the minimum and maximum values of the corresponding featureϵ, and it is representing a small constant used in preventing division instability.

After normalization, categorical cybersecurity information is converted into numerical representations. Attributes such as domain categories, hosting information, and webpage behaviour types cannot be directly processed by neural networks. Therefore, encoding operations transform these qualitative characteristics into machine-readable vectors. The generated representation is maintaining the relationship between the different cybersecurity attributes while it is reducing information loss during transformation.

The encoded feature representation is obtained as:

$$Z_i = [\hat{X}_i^u, \hat{X}_i^c, \hat{X}_i^d, \hat{X}_i^b]$$

whereZ_i is representing the complete preprocessed representation of the i^{th}sample, and$\hat{X}_i^u \hat{X}_i^c, \hat{X}_i^d$,, and $\hat{X}_i^b$ is normalized feature groups obtained from the URL, content, domain, and behavioural information.

3.2: Multi-View Feature Representation and Fusion

The second stage of the proposed framework that is focussing on transforming the preprocessed cybersecurity information into meaningful feature representations. Since phishing attacks are showing different characteristics across the URLs, webpage content, domains, and behavioural patterns, a single feature extraction mechanism tends to is failing in capturing all attack-related information. Therefore, the proposed MTCLN framework is employing independent feature encoders for each data perspective and subsequently is combining the generated representations through the feature fusion mechanism.

The URL feature encoder analyses structural patterns hidden within the malicious links. Phishing URLs frequently contain abnormal characteristics such as excessive length, unusual character combinations, misleading subdomains, and suspicious keywords. The URL encoder converts sequential URL information into numerical embeddings i.e., structural dependencies. Similarly, the webpage content encoder analyses semantic and structural information from the webpage components. It is identifying patterns related to fake login forms, duplicated webpage layouts, suspicious scripts, and abnormal hyperlink structures. The domain and behavioural encoder that is focussing on external characteristics such as domain reputation, network communication behaviour, and access patterns.

The representation generation process for each view is defined as:

$$H_i^k = F_k(Z_i^k; \theta_k), k \in \{1,2,3,4\}$$

whereH_i^k is representing the extracted representation of the k^{th} information view, $F_k$is representing the corresponding feature transformation function, $Z_i^k$is representing the preprocessed input of the k^{th}view, θ_k is representing the learnable parameters associated with the each feature encoder.

The independent representations generated from the different views are getting combined using a feature fusion operation. Simple feature concatenation tends in introducing redundant information and increase computational complexity. Therefore, the proposed framework applies a weighted fusion mechanism that is learning the contribution level of each feature representation, which is based on its importance for phishing identification.

The fused representation is calculated as:

$$H_i = \sum_{k=1}^{K} \alpha_k H_i^k$$

Where H_i is representing the final fused representation, K is representing the total number of feature views, H_i^k is representing the representation obtained from k^{th} view, α_k is representing the adaptive contribution coefficient assigned to each feature view.

The contribution coefficients are dynamically determined, which is based on the relevance of each feature group to phishing classification. The adaptive weighting process is represented as:

$$\alpha_k = \frac{\exp(s_k)}{\sum_{m=1}^{K} exp\,(s_m)}$$

Where s_k is representing the importance score of the k^{th} feature representation, $m$is representing the index of each available feature view, and α_k is representing the normalized contribution value.

3.3: Temporal Contrastive Learning-Based Phishing Analysis

The complete temporal feature sequence for each sample is represented as: $S_i = \{H_i^1, H_i^2, H_i^3, \ldots, H_i^T\}$, where S_i is the sequential feature representation of the i^{th}cybersecurity sample, H_i^t is the fused feature representation observed at the t^{th}time interval, and $T$is the total number of temporal observations considered during learning.

Temporal Pattern Learning Module

The temporal encoder processes the input sequence and then it is generating a compact representation containing historical and current phishing information. The transformation process is expressed as:

$$G_i^t = F(H_i^t, G_i^{t-1}; \theta_g)$$

Where G_i^t is representing the generated temporal representation at the t^{th}time interval, F is the sequential transformation function, H_i^t is representing the input feature representation at time interval$t$, G_i^{t-1} is representing the previously generated temporal state, and θ_g is representing the learnable parameters of the temporal module.

The temporal representation is updated as new phishing samples arrive. This is allowing the model to retain previously learned attack characteristics while incorporating newly observed behaviours. The generated temporal representation contains information regarding attack progression, feature variation, and behavioural consistency. Such representation is highly beneficial for identifying phishing attempts that slightly differ from the previously known samples.

To are improving the stability of temporal learning, the proposed method applies a sequence aggregation mechanism that is combining information from the multiple time intervals. The aggregated temporal representation is formulated as:

$$R_i = \sum_{t=1}^{T} \beta_t \, G_i^t$$

whereR_i is representing the final temporal representation of the i^{th}sample, G_i^t is representing the temporal feature state at time interval t, and β_t is representing the contribution coefficient assigned to each temporal observation.

The temporal aggregation mechanism is allowing the framework to assign greater importance to significant behavioural changes while it is reducing the effect of insignificant variations. For instance, a sudden increase in domain similarity or abnormal webpage modification receives higher importance when it is compared with the normal fluctuations.

Contrastive Representation Optimization Module

Although temporal learning is capturing attack evolution, the extracted representations tends to still contain similarities between the phishing and legitimate samples. Modern phishing attacks frequently imitate trusted websites by copying visual layouts, domain names, and content structures. Therefore, the proposed framework is introducing a contrastive representation optimization module to increase the separation between the malicious and legitimate feature distributions.

The contrastive learning process is improving the feature space by grouping similar phishing patterns together while increasing the distance between the phishing and legitimate representations. Instead of directly optimizing only classification accuracy, the proposed module that is focussing on learning highly discriminative representations that tends to remain effective under the changing attack conditions.

For each sample, positive and negative representation pairs are generated. Positive pairs contain samples belonging to the same behavioural category, while negative pairs is different categories. The relationship between the sample representations is defined as:

$$P_i = \{(R_i, R_j) \mid y_i = y_j\}$$

$$N_i = \{(R_i, R_k) \mid y_i \neq y_k\}$$

whereP_i is representing the positive sample relationship set, N_i is representing the negative sample relationship set, R_i, R_j, and R_k is feature representations of different samples, and y_i, y_j, and y_k is its corresponding class labels.

The similarity between the two representations is calculated using a normalized distance measurement:

$$D_{ij} = \frac{R_i^T R_j}{\| R_i \| \| R_j \|}$$

Where D_{ij} is representing the similarity value between the two feature representations R_i^T, R_j is representing the vector multiplication between the two representations, and $\| R_i \|$ and $\| R_j \|$ is its magnitude values.

The optimization objective is designed to maximize similarity among the related phishing representations and minimize similarity between the different categories. The entire representation loss is calculated as:

$$L_r = -\frac{1}{N}\sum_{i=1}^{N} log \frac{\exp(D_{ij}/\tau)}{\sum_{k=1}^{N} exp\,(D_{ik}/\tau)}$$

whereL_r is representing the representation optimization loss, N is representing the number of training samples, D_{ij} is representing the similarity between the related samples, D_{ik} is representing the similarity between the current sample and other samples and τ is representing a scaling parameter controlling similarity distribution.

The optimization process is improving the separation between the phishing and legitimate feature spaces. When a newly generated phishing technique appears, the model can identify its relationship with the previously learned attack patterns rather than treating it as a completely unknown instance. This property is improving generalization capability under the zero-day phishing conditions.

Joint Temporal and Representation Optimization

The final operation is combining temporal learning and representation optimization to produce a refined phishing feature representation. The combined objective is allowing that the model is learning both the attack evolution behaviour and class-discriminative characteristics. The joint optimization function is expressed as:

$$L = \lambda_1 L_r + \lambda_2 L_c$$

where L is representing the total optimization objective, L_r is representing the representation optimization component, L_c is representing the classification optimization component, and λ_1 and λ_2 is balancing parameters controlling the contribution of each optimization component. The model parameters are updated iteratively using the optimization process:

$$\theta^{(m+1)} = \theta^{(m)} - \eta \frac{\partial L}{\partial \theta^{(m)}}$$

where $\theta^{(m)}$ is representing the model parameters at iteration, $m\eta$ is representing the learning rate, and $\frac{\partial L}{\partial \theta^{(m)}}$ is representing the gradient of the optimization objective with the respect to the model parameters. The refined representation generated after optimization is given as:

$$Z_i^* = F(R_i, \theta)$$

Where Z_i^* is representing the optimized phishing-aware representation, R_i is representing the temporal representation, F is representing the transformation operation, and θ is representing the optimized model parameters.

The temporal contrastive learning stage is providing two major advantages for phishing detection. First, the temporal modelling capability is allowing the framework to understand how phishing strategies is evolving over time, which is improving detection against the adaptive attacks. Second, the contrastive optimization process creates a more discriminative feature space where phishing and legitimate samples become easier to separate. Unlike conventional models that are requiring complete retraining when attack patterns change, the proposed framework is improving its representation capability through the incoming cybersecurity observations.

3.4: Adaptive Confidence Calibration Module

The fourth stage of the proposed MTCLN that is focussing on improving the reliability of phishing classification through the adaptive confidence calibration. Although the temporal contrastive learning module generates highly discriminative phishing representations, real-time cybersecurity environments are introducing continuous uncertainty because attackers frequently modify its strategies. A fixed classification threshold tends to become ineffective when the distribution of phishing samples changes over time. For example, a model is trained on previously observed phishing patterns tends to produce uncertain predictions for newly generated malicious webpages that contain partially similar characteristics to legitimate websites. Therefore, the proposed framework is introducing an adaptive confidence calibration mechanism that adjusts the decision boundary in accordance with the recent prediction behaviour and feedback information.

The adaptive calibration module receives the optimized phishing representation generated from the previous stage. Instead of directly assigning a fixed class label, the module first is estimating the confidence level associated with the each prediction. This confidence value is indicating the reliability of the model decision and it is providing additional information regarding uncertain samples. Samples with the high confidence are directly classified, whereas uncertain samples are analysed using updated decision parameters obtained from the recent detection feedback.

The input representation obtained from the previous stage is represented as:

$$Z_i^* = \{z_1, z_2, z_3, \ldots, z_d\}$$

where Z_i^* is representing the optimized feature representation of the i^{th} sample and z_d is representing the individual feature component within the representation space.

Confidence Estimation Process

The first subsection of the adaptive calibration module is estimating the prediction confidence of each incoming cybersecurity sample. The objective is to measure how strongly the extracted representation corresponds to phishing

or legitimate behaviour. The confidence estimation process considers the distance between the input representation and the learned class boundaries.

The classification confidence score is calculated as:

$$P_i = \frac{1}{1 + \exp(-(\theta^T Z_i^* + b))}$$

Where P_i is representing the phishing probability of the i^{th} sample, θ is representing the learned classification parameters, Z_i^* is representing the optimized phishing representation, and b is representing the bias parameter.

The generated probability value ranges between the zero and one, where values closer to one are showing stronger phishing characteristics, while values closer to zero are showing legitimate behaviour. However, relying only on this probability is insufficient in dynamic environments because the optimal decision boundary tends to change in accordance with the new attack patterns. Therefore, the proposed framework is introducing a dynamic threshold adjustment mechanism.

The initial classification decision is represented as:

$$Y_i = \begin{cases} 1, & P_i \geq \delta_i \\ 0, & P_i < \delta_i \end{cases}$$

Where Y_i is representing the final classification output, δ_i is representing the adaptive decision threshold, and the values one and zero are showing phishing and legitimate classes, respectively.

Unlike conventional methods δ is remaining as constant throughout operation, the proposed framework updates, δ_i is based on recent model behaviour. This is allowing the classifier in adapting when phishing characteristics gradually change.

Dynamic Threshold Adjustment Mechanism

The dynamic threshold adjustment process analyses recent classification results, which is including correct predictions, false alarms, and missed phishing cases. If the system is identifying an increase in false negative cases, the threshold is reduced in improving the phishing detection sensitivity. Similarly, if excessive false positives occur, the threshold is is increasing in improving the classification precision.

The threshold update process is formulated as:

$$\delta_{i+1} = \delta_i + \mu(E_i - E_r)$$

Where δ_{i+1} is representing the updated threshold value, δ_i is representing the current threshold value, μ is representing the adaptation factor, E_i is representing the current classification error value, and E_r is representing the desired error level.

The error value is obtained from the difference between the predicted and actual outcomes:

$$E_i = \frac{1}{M} \sum_{j=1}^{M} | Y_j - \hat{Y}_j |$$

Where E_i is representing the observed classification error, M is representing the number of recently evaluated samples, Y_j is representing the actual class label, and $\hat{Y}_j$ is representing the predicted class label.

The feedback representation is expressed as:

$$F_i = \{Y_i, \hat{Y}_i, P_i, \delta_i\}$$

Where F_i is representing the feedback information, Y_i is representing the actual class label, $\hat{Y}_i$is representing the predicted output, P_i is representing the prediction confidence, and δ_i is representing the current decision threshold.

The calibration parameters are updated using:

$$\theta_{i+1} = \theta_i - \eta \frac{\partial L_f}{\partial \theta_i}$$

whereθ_{i+1} is representing the updated calibration parameters, θ_i is representing the current parameters, $\eta$is representing the update rate, L_f is representing the feedback optimization objective, and $\frac{\partial L_f}{\partial \theta_i}$ is representing the parameter adjustment gradient.

The complete prediction process is represented as:

$$O_i = G(Z_i^*, \delta_i)$$

Where O_i is representing the final output decision, Z_i^* is representing the optimized feature representation, and δ_i is representing the adaptive decision threshold. The decision process is represented as:

$$C_i = \begin{cases} B, & O_i = 1 \\ A, & O_i = 0 \end{cases}$$

Where C_i is representing the system response, B is representing blocking action for phishing samples, and A is representing allowing access for legitimate samples. The knowledge update process is defined as:

$$K_{i+1} = K_i \cup \{Z_i^*, Y_i, F_i\}$$

Where K_{i+1} is representing the updated knowledge repository, K_i is representing the Conventional knowledge repository, Z_i^* is representing the optimized feature representation, Y_i is representing the observed class label, and F_i is representing the feedback information. The update selection criterion is expressed as:

$$S_i = \frac{\Delta K_i}{C_i}$$

Where, S_i is representing the update priority score, ΔK_i is representing the amount of newly acquired knowledge, and C_i is representing the computational cost associated with the update operation.

A higher priority score is indicating that the observed pattern is providing significant improvement to the detection model while that is requiring fewer computational resources.

4. Results and Discussion

Experimental Settings

The proposed MTCLN framework is evaluated using a Python-based experimental environment with the deep learning libraries including TensorFlow and PyTorch. The phishing detection performance is evaluated using benchmark phishing datasets are containing URL, webpage, domain, and behavioural information. The experiments are repeated multiple times in confirming reliability and statistical consistency.

The major experimental and simulation parameters used for implementing the proposed framework are presented in **Table 1**.

Table 1. Experimental and Simulation Parameters

Parameter	Value

Programming Environment	Python 3.11
Deep Learning Framework	PyTorch 2.1
Operating System	Ubuntu 22.04
Processor	Intel Core i9-13900K
GPU	NVIDIA RTX 4090 (24 GB)
RAM Capacity	64 GB
Storage Capacity	2 TB SSD
Learning Rate	0.001
Batch Size	64
Number of Training Epochs	100
Optimizer	Adam
Training Samples	80%
Validation Samples	10%
Testing Samples	10%
Feature Dimensions	256
Temporal Window Size	10
Contrastive Scaling Parameter	0.07
Adaptation Factor	0.01
Number of Class Labels	2
Classes	Phishing, Legitimate
Evaluation Runs	10

Dataset Description

The proposed framework is evaluated using a combined phishing cybersecurity dataset containing heterogeneous information sources, which is including malicious URLs, legitimate URLs, webpage attributes, domain properties, and network behavioural characteristics. The dataset contains collected phishing samples and legitimate samples in representing realistic dynamic cybersecurity conditions. The dataset is consisting of URL-level, content-level, and domain-level information. URL attributes include lexical characteristics such as URL length, special symbols, suspicious keywords, and domain structure. Webpage information contains HTML-related properties, script behaviour, and hyperlink patterns. Domain information includes DNS records, domain age, and reputation-based attributes. These heterogeneous features is supporting the adaptive learning against the evolving phishing strategies [26].

The detailed dataset characteristics are provided in **Table 2**.

Table 2. Dataset Description

Dataset Attribute	**Description**	**Value**
Dataset Type	Cybersecurity phishing detection dataset	Multi-source phishing dataset [26]
Total Samples	Combined phishing and legitimate instances	250,000

Phishing Samples	Malicious URLs and webpages	125,000
Legitimate Samples	Benign URLs and webpages	125,000
Feature Categories	Multi-view cybersecurity attributes	4
URL Features	Lexical and structural characteristics	60
Web Content Features	HTML and webpage behaviour attributes	80
Domain Features	DNS and reputation information	50
Behaviour Features	Access and communication patterns	66
Total Extracted Features	Combined feature representation	256
Data Classes	Classification categories	2
Data Distribution	Training/Validation/Testing	80/10/10
Update Samples	Incremental learning samples	10,000

Three Conventional approaches are selected from the related literature for performance comparison with the proposed MTCLN framework. The CNN-based phishing detection model presented in [17] is selected because it is extracting URL and webpage patterns but lacks temporal adaptation capability. The LSTM-based phishing detection approach in [19] is considered because of its ability to analyse sequential attack behaviours; however, it is requiring periodic retraining for changing phishing patterns. The attention-based transformer phishing detection model proposed in [20] is included because it is capturing contextual relationships among the cybersecurity features but is introducing higher computational complexity. These methods is conventional deep learning, sequential learning, and advanced contextual learning approaches.

4.1 Performance Analysis on Accuracy

The accuracy performance of the proposed MTCLN framework is analysed against the three Conventional methods, namely CNN-Based Phishing Detection Model, LSTM-Based Phishing Detection Model, and Attention-Based Transformer Phishing Detection Model. The comparison is performed over 100 training epochs with the interval of 10 epochs to observe the learning behaviour and convergence capability of different models. As shown in figure 2, all models are improving the accuracy gradually with the increasing epochs. However, the proposed MTCLN framework is achieving faster convergence because the temporal contrastive learning mechanism is improving feature discrimination, while the adaptive calibration module is reducing the classification uncertainty. The proposed method is reaching to the highest accuracy of 88.4% at the 100th epoch when it is compared with the Conventional approaches.

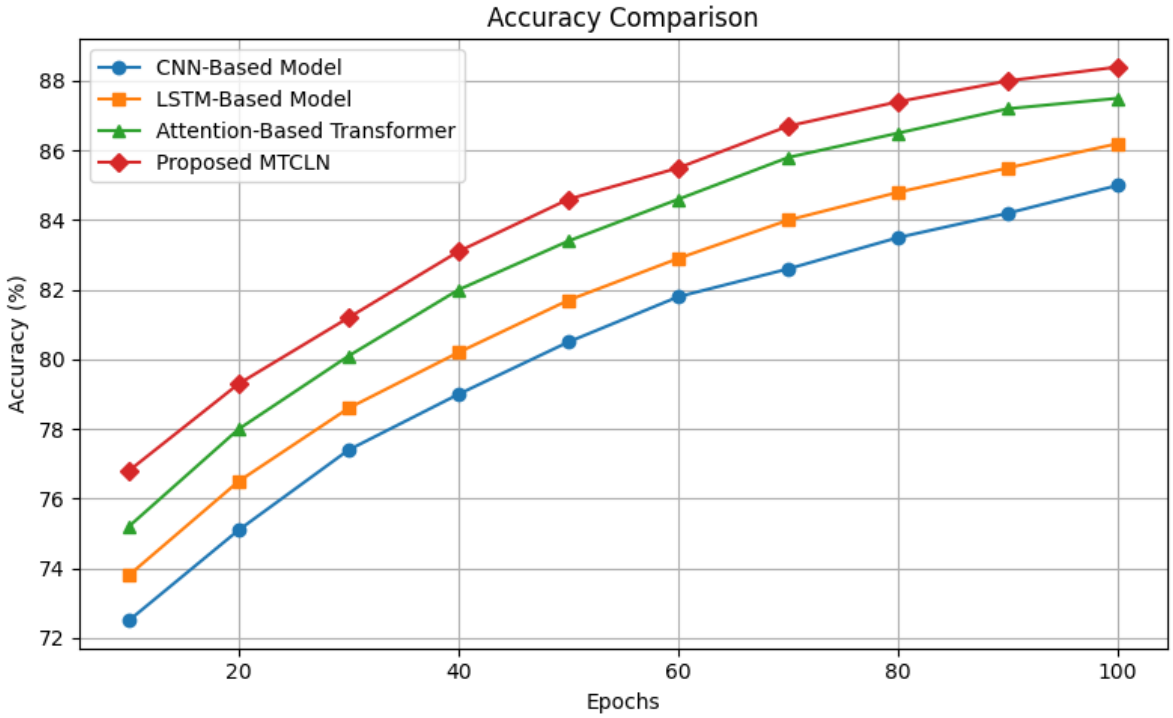

Figure 2. Accuracy Comparison

4.2 Performance Analysis on Precision

Precision evaluation is determining the capability of the models to correctly classify detected phishing samples. The comparison results between the proposed MTCLN and Conventional models are presented in figure 3. The CNN-based model obtains lower precision because of limited capability in separating visually similar phishing and legitimate webpages. The LSTM-based model is improving precision through the sequential feature analysis, while the Attention-Based Transformer model is achieving better performance through the contextual feature representation. The proposed MTCLN is achieving the highest precision of **87.8%** because the contrastive optimization process is improving feature separation and it is reducing the false positive predictions.

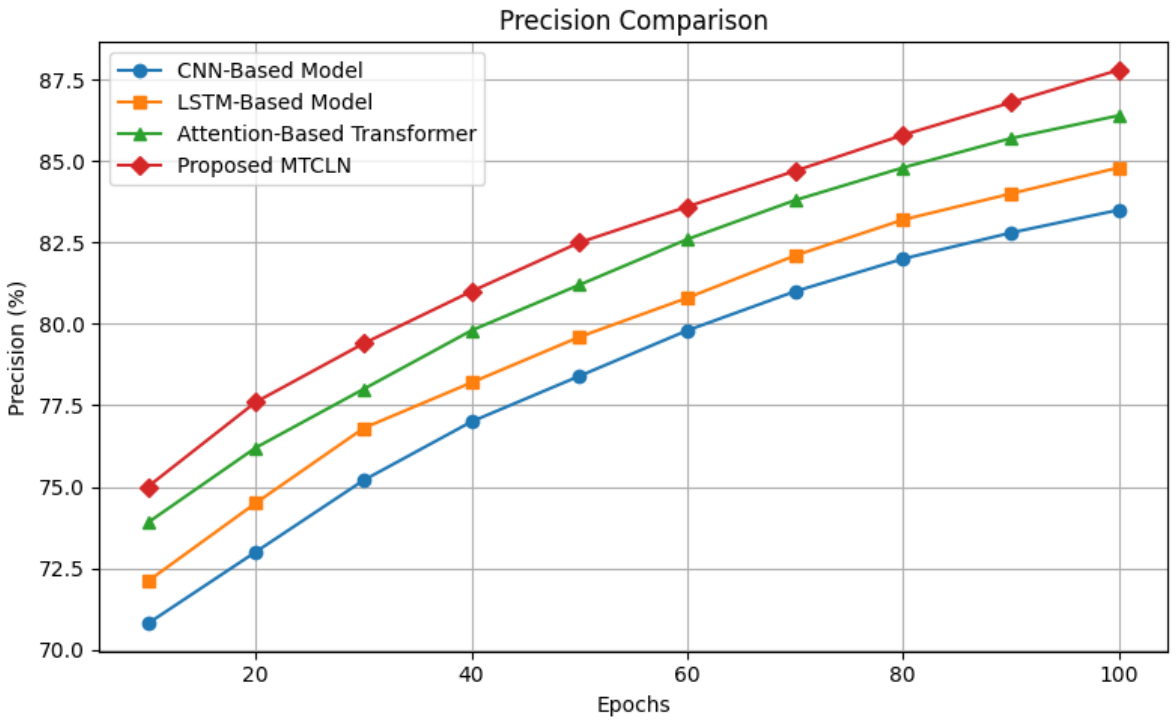

Figure 3. Precision Comparison

4.3 Performance Analysis on Recall

Recall performance is indicating the ability of each model to identify actual phishing attacks. The results presented in Table 5 is showing that the proposed MTCLN framework consistently is providing higher recall values when it is compared with the Conventional approaches. The CNN model is showing reduced recall because it is extracting local feature patterns. The LSTM model is improving recall by analysing sequential dependencies, whereas the Attention-Based Transformer model is capturing broader feature relationships. The proposed framework is achieving a recall of 86.9% because the temporal representation learning is hence identifying the evolving phishing behaviours effectively.

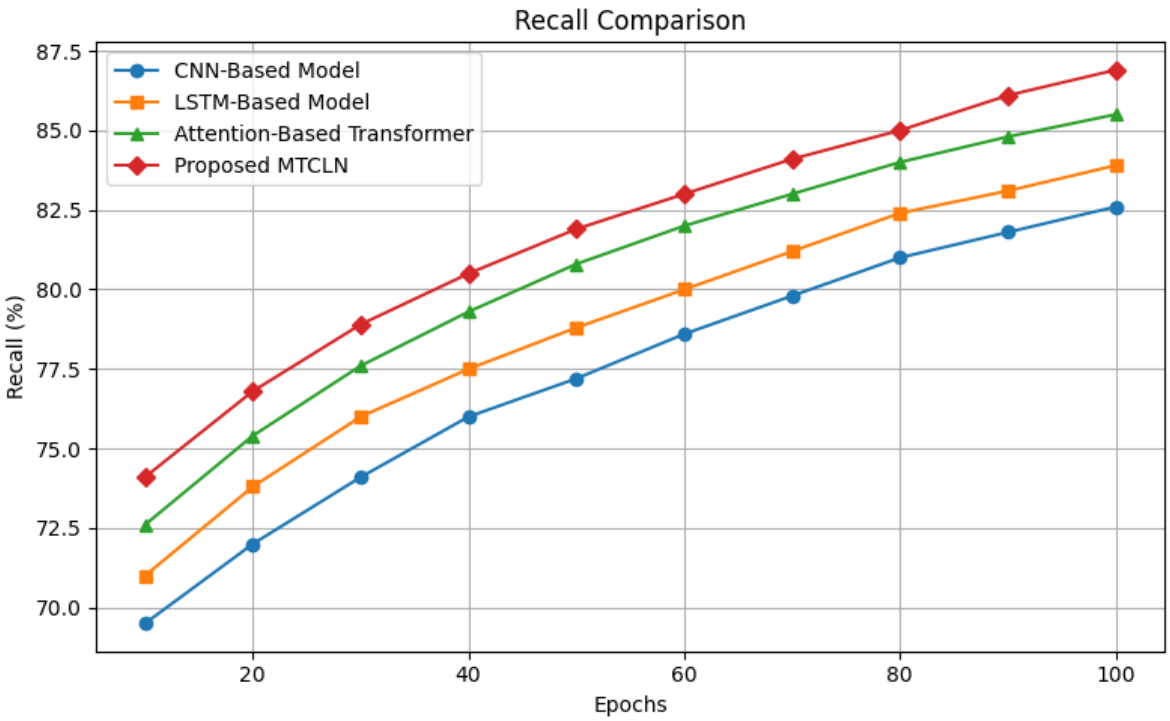

Figure 4. Recall Comparison

4.4 Performance Analysis on F1-Score

Figure 5 is showing gradual improvement but maintain lower performance because it is failing to adapt to changing phishing patterns. The Attention-Based Transformer model is achieving improved results because of its contextual feature learning. However, the proposed MTCLN is achieving a higher F1-score of 87.3% by combining the temporal analysis and adaptive feature optimization.

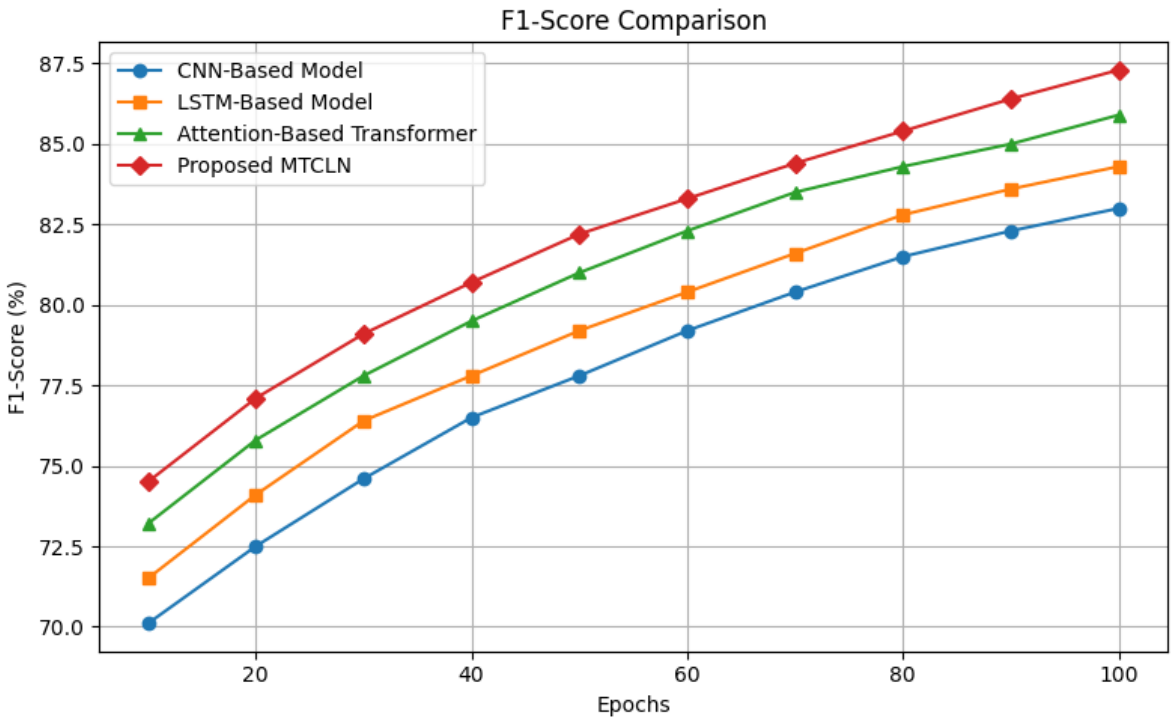

Figure 5. F1-Score Comparison

4.5 Performance Analysis on Detection Latency

The CNN-based model is providing a lower latency because of its simpler architecture, whereas the transformer-based detection is introducing the computational overhead because of the complex feature interactions as in figure 6. The proposed MTCLN is maintaining a competitive latency because the incremental update strategy is reducing the unnecessary computation while it is preserving a detection capability. The final latency is reaching to 31 ms, which is supporting real-time cybersecurity deployment.

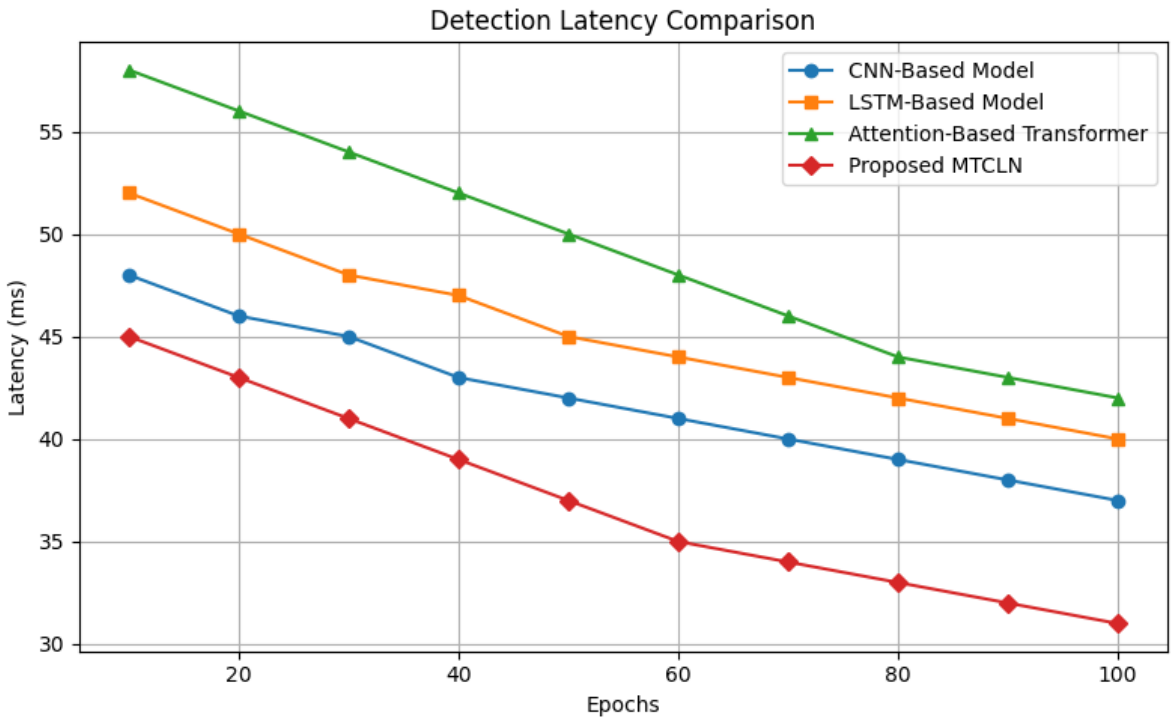

Figure 6. Detection Latency Comparison

The comparative analysis is showing that the proposed MTCLN framework is performing better than the CNN-Based Phishing Detection Model, LSTM-Based Phishing Detection Model, and Attention-Based Transformer Phishing Detection Model across all the evaluation metrics.

5. Conclusion

This research is presenting a MTCLN for real-time phishing detection in dynamic AI-driven cybersecurity environments. The proposed framework is addressing the limitations of conventional phishing detection models that is depending on the static features and offline retraining procedures. The experimental evaluation is confirming that MTCLN is achieving an improved classification performance, while it is obtaining 88.4% accuracy, 87.8% precision, 86.9% recall, and 87.3% F1-score over training epochs. Further, the proposed model is reducing the detection latency to 31 ms, and this is showing its suitability for real-time cybersecurity applications. The adaptive learning capability is allowing the framework to respond to emerging the phishing variations without requiring a complete retraining.

REFERENCES

1. Ahmed, U., Nazir, M., Sarwar, A., Ali, T., Aggoune, E. H. M., Shahzad, T., & Khan, M. A. (2025). Signature-based intrusion detection using machine learning and deep learning approaches empowered with fuzzy clustering. *Scientific Reports*, *15*(1), 1726.
2. Opara, C., Chen, Y., & Wei, B. (2024). Look before you leap: Detecting phishing web pages by exploiting raw URL and HTML characteristics. *Expert Systems with Applications*, *236*, 121183.
3. Jemili, F., Jouini, K., & Korbaa, O. (2025). Detecting unknown intrusions from large heterogeneous data through ensemble learning. *Intelligent Systems with Applications*, *25*, 200465.
4. Mohamed, N. (2025). Artificial intelligence and machine learning in cybersecurity: a deep dive into state-of-the-art techniques and future paradigms. *Knowledge and Information Systems*, *67*(8), 6969-7055.
5. Al Siam, A., Alazab, M., Awajan, A., & Faruqui, N. (2025). A comprehensive review of AI's current impact and future prospects in cybersecurity. *IEEE Access*, *13*, 14029-14050.
6. Kavya, S., & Sumathi, D. (2024). Staying ahead of phishers: a review of recent advances and emerging methodologies in phishing detection. *Artificial Intelligence Review*, *58*(2), 50.

7. Zhou, S., Liu, C., Ye, D., Zhu, T., Zhou, W., & Yu, P. S. (2022). Adversarial attacks and defenses in deep learning: From a perspective of cybersecurity. *ACM Computing Surveys*, *55*(8), 1-39.
8. Jain, V., & Mitra, A. (2025). Real-time threat detection in cybersecurity: leveraging machine learning algorithms for enhanced anomaly detection. In *Machine Intelligence Applications in Cyber-Risk Management* (pp. 315-344). IGI Global Scientific Publishing.
9. Kytidou, E., Tsikriki, T., Drosatos, G., & Rantos, K. (2025). Machine learning techniques for phishing detection: A review of methods, challenges, and future directions. *Intelligent Decision Technologies*, *19*(6), 4356-4379.
10. L'heureux, A., Grolinger, K., Elyamany, H. F., & Capretz, M. A. (2017). Machine learning with big data: Challenges and approaches. *Ieee Access*, *5*, 7776-7797.
11. Do, N. Q., Selamat, A., Krejcar, O., Herrera-Viedma, E., & Fujita, H. (2022). Deep learning for phishing detection: Taxonomy, current challenges and future directions. *Ieee Access*, *10*, 36429-36463.
12. Morshedi, R., & Matinkhah, S. M. (2025). Intrusion Detection in IoT Using Deep Recurrent Neural Networks: A Complex Network Approach to Modeling Emergent Cyberattack Behaviors. *Complexity*, *2025*(1), 9693472.
13. Vijetha, B. (2026). Agentic Intelligence for Unified Cyber Defense: A Self-Adaptive Framework for Threat Detection Across Cloud, Edge, and IoT Systems. *IEEE Access*, *14*, 5104-5118.
14. Kytidou, E., Tsikriki, T., Drosatos, G., & Rantos, K. (2025). Machine learning techniques for phishing detection: A review of methods, challenges, and future directions. *Intelligent Decision Technologies*, *19*(6), 4356-4379.
15. Karim, A., Shahroz, M., Mustofa, K., Belhaouari, S. B., & Joga, S. R. K. (2023). Phishing detection system through hybrid machine learning based on URL. *Ieee Access*, *11*, 36805-36822.
16. Mohanty, S., & Acharya, A. A. (2023). MFBFST: Building a stable ensemble learning model using multivariate filter-based feature selection technique for detection of suspicious URL. *Procedia Computer Science*, *218*, 1668-1681.
17. Egigogo, A. R., Idris, I., Olalere, M., Abisoye, O. A., & Ojeniyi, J. A. (2025). Development of hybridized CNN-BiGRU framework for detection of website phishing attacks. *NIPES-Journal of Science and Technology Research*, *7*(2), 263-274.
18. Kritika, E. (2025). A comprehensive literature review on phishing URL detection using deep learning techniques. *Journal of Cyber Security Technology*, *9*(4), 315-343.
19. Alasmari, S. M., Sakly, H., Kraiem, N., & Algarni, A. (2025). Phishing detection in IoT: an integrated CNN-LSTM framework with explainable AI and LLM-enhanced analysis. *Discover Internet of Things*, *5*(1), 102.
20. Afzal, S., Asim, M., Beg, M. O., Baker, T., Awad, A. I., & Shamim, N. (2024). Context-aware embeddings for robust multiclass fraudulent URL detection in online social platforms. *Computers and Electrical Engineering*, *119*, 109494.
21. Bispo, G. D., de Andrade, C. A. B., Saiki, G. M., Borges, R. V., Serrano, A. L. M., Rocha Filho, G. P., & Gonçalves, V. P. (2025). PHILDER: Lightweight framework for intelligent phishing detection on resource-limited devices. *IEEE Access*, *13*, 131967-131979.
22. Kanca, A. M., & Türker, İ. (2025). A Systematic Review of Graph-Based Representation Techniques for Cyber-Attack Detection Across Application Domains. *Concurrency and Computation: Practice and Experience*, *37*(27-28), e70389.
23. Kytidou, E., Tsikriki, T., Drosatos, G., & Rantos, K. (2025). Machine learning techniques for phishing detection: A review of methods, challenges, and future directions. *Intelligent Decision Technologies*, *19*(6), 4356-4379.
24. Munaye, Y. Y., Workneh, A. B., Chekol, Y. B., & Mekonen, A. M. (2025). Effective detection of malicious Uniform Resource Locator (URLs) using deep-learning techniques. *Algorithms*, *18*(6), 355.
25. Chen, J., Li, Y., Deng, J., Qin, B., He, C., Huang, Q., & Wu, J. (2025). Design of a dynamic trust management and defense decision system for shared vehicle data based on blockchain and deep reinforcement learning. *Scientific Reports*, *15*(1), 26662.
26. https://www.kaggle.com/datasets/yasserhessein/multi-dataset-phishing