



Adaptive Multi-Modal Phishing Detection (AMPD) Framework Using Attention-Based Dynamic Fusion

Sam Sharma^{1*}, Saksham Singh², Paramjeet Singh³, Shagun Kumar⁴, Punit Mittal⁵,
Swati Sharma⁶

^{1, 2, 3, 4, 5, 6} Department of Computer Science & Information Technology, Meerut Institute
Of Engineering and Technology, Meerut, India

*sam.sharma.csit.2022@miet.ac.in

Abstract. Phishing attacks are a major issue in cybersecurity, and modern detection systems have a weakness of disjointed feature integration, which restricts their ability to handle advanced multi-vector campaigns. The current paper introduces a novel Adaptive Multi-Mode Phishing Detection (AMPD) system that combines URL lexical features, visual webpage features and behavioral interaction patterns in the form of an attention-based dynamic fusion system. Unlike the proposed conventional methods that use fixed feature weights, model uses a context-sensitive attention design that recalibrates modality input dynamically, depending on computation limitations and threat properties. The framework includes a lightweight threat-aware pre-filter of resource-efficient mobile-optimized compressed visual hashing triage. convolutional neural networks and privacy-preserving behavioral proxies to interaction analysis. It is proven by experimental assessment of a multi-modal dataset of 12,000 phishing and 10,000 legitimate sites. that the AMPD framework gets 98.67% accuracy and 47ms mean inference time on mobile devices, which surpasses single-modality baselines, offering around 5.2 times faster inference and an accuracy improvement of 8.4%, while preserving explainability through cross-modal SHAP analysis. The Findings prove that attention-based dynamic fusion is viable to solve the accuracy-efficiency trade-off of multi-modal phishing detection, which offers an effective solution for real-time phishing detection.

Keywords: Phishing detection, multi-modal feature fusion, attention mechanism, explainable AI, mobile security, deep learning, adaptive resource allocation, dynamic weighting, cross-modal SHAP, behavioral analysis

1 Introduction

The spread of phishing is one of the most consistent and costly threats of the modern cybersecurity. Recent industry reports record more than one million phishing cases per quarter, which can be regarded as 40.9% causing losses to all countries in the world estimated to be over ten

billion dollars per year [15] and increase of annual figures on a year-over-year basis. The development of the phishing campaigns has advanced to basic URL obfuscation to complex multi-vector techniques that effectively used at the same time take advantage of technical weaknesses and cognitive biases. Contemporary attackers use highly fuzzed visual interfaces, socially engineered lures, and domain manipulation strategies that all render legacy detection strategies powerless [16].

The current phishing detection studies have yielded superior systems of phishing detection in each analytical mode individually. URL based methods that make use of lexical analysis and Embeddings based on transformer can reach the accuracy of between 98.93% in controlled settings with latency in the milliseconds range that is mobile-friendly deployment [1]. Convolutional neural networks are used to compute visual similarities and hybrid architectures achieve 99.18% accuracy in zero-day phishing pages through webpage characteristics rendered [2]. The 89-93% come by behavioral analysis methods that observe dynamics of mice, hesitation, and anomaly of interaction.

precision in detecting the victims of phishing in active sessions [6]. In spite of these great performances in isolation, 15-30% accuracy is observed in real-life applications. degradation caused by fractured feature integration and signal correlation inability. across modalities [13].

The basic restriction of the existing detection paradigms is the lack of adaptive frameworks that can dynamically combine multi-modal features when processing constraints are real time. There are other methods of statistic fusion that merge features with the help of fixed weighting schemes do not take into consideration the context-specific utility of the various modalities. As an example, visual characteristics give important discriminative information to zero-day attacks are computationally infeasible with resource-constrained mobile devices. On the other hand, URL features provide efficient processing with vulnerability to advanced devices of obfuscation that maintain lexical normativity and yet contain malicious code. There is also the absence of explainable which adds to the research gap security mechanisms which would allow security analysts to comprehend cross-modal rationales of making decisions and establish confidence in automated systems.

The paper covers these limitations in the following contributions:

1. An innovative Adaptive Multi-Mode Phishing Detection (AMPD) model that combines URL lexical, visual webpage, and behavioral interaction patterns using an attention-based dynamic fusion process.
2. Threat-conscious feature selection that does lightweight pre-filtering based on low-cost heuristics to adaptively allocate computational resources as a result of risk assessment and device capabilities.

3. Mobile-friendly feature extraction engines, such as compressed transformer-based URL encoding, visual hashing engine trained with SqueezeNet, and privacy-preserving behavioral proxies which reduce computational costs even though still being discriminative.
4. Multi-head attention-based dynamic fusion core that learns to combine the importance of modalities with contextual features such as device type, network conditions, and attack characteristics.
5. Extensive experimental validation demonstrating significant improvements compared to single-modality baselines and existing fusion methods. A detailed analysis of accuracy-efficiency trade-offs in deployment scenarios is also provided.

The remainder of this paper is organized as follows: Section 2 reviews related work on phishing detection across various modalities. Section 3 contains the suggested system architecture and methodology. The dataset and feature engineering are explained in section 4 approach. Section 5 describes the experimental design and measures of evaluation. Section 6 presents results and discussion, comparative analysis with existing models. Section 7 is the conclusion of the paper and the future research directions.

2 Related Work

2.1 URL-Centric Detection Approaches

Analysis of URLs has long been the most prevalent research in phishing detection because it is computationally efficient and does not need page rendering to operate. The initial methods used manually crafted lexical features such as URL length,

frequency of special characters, depth of subdomain and the presence of IP addresses with 92-96% accuracy with the help of the traditional machine learning classifiers such as the Random Forest and the Support Vector Machines [4]. Sahingoz et al. [4] proved that 23 attentively learned lexical features may achieve an accuracy of 97.98% but they reported the vast domain knowledge needed to design features and the susceptibility to adversarial obfuscation.

The advent of deep learning structures has revolutionized URL-based detection. through automatic feature learning. Transformer-based embeddings, particularly BERT-based representations, find semantic patterns within malicious URLs that manual feature engineering can overlook [8]. These models realize context manipulation, e.g. the positioning of valid brand names before misleading words inside suspicious domains, which disclose malicious intent, and cannot be identified by lexical features.

Phish-Jam framework [1] brings together handcrafted features with transformer embeddings in a super learner ensemble, and reaches 98.93 percent accuracy without thirdparty dependencies and sub-20ms latency, which can be used on a mobile device.

The blackbox character of deep learning models has been overcome by explainable AI methods applied to the analysis of URLs. SHAP and LIME are used in the SLA-FS framework [9]. LIME descriptions to maximize feature selection, maximizing the accuracy of Random Forest by 0.65% by identifying 12 optimum features and giving interpretable decision rationales.

Nevertheless, URL-based methods have 22-35% increased false negative rates with visually advanced phishing websites that use legitimate appearing domains with flawless visual copies [2].

2.2 Visual Similarity Methods

Visual analysis solutions overcome the weaknesses of URL-based solutions by analyzing the appearance of webpages to identify cloning attacks. Histogram comparison, CSS stylometry and logo recognition were used as early methods in computer vision to use visual similarity as a measure of legitimate sites, with 85-92% accuracy, but with a large computational demand [5]. These techniques match rendered page features such as colour distributions, layout structures, and brand logo appearance to recognize pages that visually resemble trusted party.

Visual phishing is a technology that has been advanced greatly through deep learning architectures. Convolutional Neural Networks learn hierarchical representations of images, derived out of webpage screenshots, to extract discriminative visual patterns, both on a global scale (global layout structure) and local scale (local-design

elements). CNN-SVM-GWO framework [2] is a convolutional layers optimization framework based on the use of the grey wolf Optimizer hyperparameter optimization, with state-of-the-art 99.18% accuracy on multi-source databases.

Nevertheless, the computing power of rendering a full page and deep feature extraction. achieve latency of 300-500ms, which makes them inapplicable to mobile implementations or real-time browser extensions.

The recent studies have investigated the dimensionality reduction methodologies to facilitate the visual analysis on resource-constrained devices. CSS hashing and logo fingerprinting methods produce small visual images that maintain discriminative data and lower computational costs requirements [5]. These techniques concentrate on major visual features as opposed to the complete pixel-to-pixel comparison, obtaining processing speeds of less than 100ms with a relative degradation of accuracy.

However, the need to render pages is an inherent constraint, which exposes systems to.

drive-by downloads at the time of analysis [4].

2.3 Behavioral Profiling Techniques

Behavioral analysis is the least examined mode of phishing detection despite.

its special capability to detect attacks as the user interacts with the system and not prior to page access. More dynamic research shows accuracy of 89-93% in differentiating.

phishing victims from legitimate users through analysis of hesitation patterns, cursor interaction timing, path anomalies, and interaction timing [6].

Phishing victims have typical behavioral patterns, such as increased decision time, unpredictable mouse behavior and speeded up form completion after trust had been developed, and during page legitimacy assessment.

Practical has been constrained by privacy issues and the need to instrument the client sideacquisition of behavioral methods. Complete behavioral profiling involves constant observation of user activities, which involves serious privacy concerns and requires.

prudent data governance models [6].

Recent research has covered privacy-preserving techniques that compute behavioral statistics locally on end-users devices and only send aggregate scores of anomalies instead of the underlying interaction sequences. Federated learning architectures allow the enhancement of models in distributed deployments without centralizing information about sensitive behaviors.

Very little has been done in the integration of behavioral signals and technical features. None of the reviewed papers are effective correlators of mouse dynamics or hesitation measures with URL abnormality or visual features even though phishing success has been proved relies on both technical and psychological deception [6].

This fragmentation is a great missed opportunity, since it can be behaviorally confirmed that False positive rates could be significantly decreased by technical suspicions.

2.4 Hybrid and Ensemble Approaches

The awareness of single-modality constraints has stimulated studies on hybrid detection Frameworks combining several types of features. Ensemble techniques like stacking.

and super learner architectures combine the predictions of various base models to achieve improved generalization [1]. Phish-Jam framework is a composite of URL-based plays in a super learner group but excludes visual and behavioral analysis, restricting its ability to counter attacks of a visual sophistication. The CNN-SVM-GWO approach fuses visual and HTML but disregards behavioral signals, losing the chance.

for interaction-based confirmation [2].

Multi-modal fusion is a relatively new concept in the research of phishing detection. The Multi-Mode Hypergraph Fusion Network (MM-HFN) uses hypergraphs to learn high-order textual feature to visual feature relationships in BERT on CNNs, the accuracy is 95-97% .Temporal Graph Neural Networks with attention mechanisms address temporal variations in phishing behavior, providing dynamic detection capabilities . Despite these advances, existing fusion approaches employ static weighting schemes that fail to adapt to changing contexts, and none integrate behavioral signals with technical features.

2.5 Research Gaps and Opportunities

The literature review demonstrates that there are four crucial gaps that drive the current study:

Dynamic Feature Fusion: Existing methods of integration use fixed weights irrespective of feature utility which is context-dependent. There is no current framework dynamically reweighting modality contributions depending on attack characteristics, device capabilities, or environmental constraints.

Mobile-Specific Architecture: Although URL analysis has been customized to deploy on mobile platforms [1], there is no fully visual-behavioral integration done to date in this regard resource-constrained devices. The visual techniques are ineffective against zero-day attacks because of its computational intensity which does not allow them to be deployed.

Explainable Integration: Current XAI models make the most of single-modality characteristics but is unable to describe cross-modal decisions. This poses trust challenges in business settings where false positives have high operational values [9].

Behavioral Correlation: Behavioral hints do not intertwine with technical characteristics even though it is proved that both technical and psychological manipulation and deception are required to achieve phishing success. None of the reviewed studies correlate mouse dynamics with URL abnormalities or visual features.

The suggested Adaptive Multi-Modal Phishing Detection model directly solves the gaps with attention-based dynamic fusion, mobile-optimized feature extraction, explainable integration mechanism, and the overall behavioral-technical correlation.

Table 1: Comparison of Recent (2024-2025) Phishing Detection Approaches

Method	Year	Modalities	Fusion Type	Mobile-Ready	Explainable	Novelty Claim vs. AMPD
Phish-Jam [1]	2025	URL only	Super learner ensemble	✓	✗	No visual/behavioral; static weighting
CNN-SVM-GWO [2]	2025	Visual + HTML	Hybrid (no attention)	✗	✗	No behavioral; resource-intensive
EasyDMARC [14]	2024	URL only	Neural network	✓	✗	Single modality only
Phish-Blitz [18]	2025	Webpage resources	Visual integrity	✗	✗	No behavioral or dynamic fusion
Multi-modal Temporal GNN [19]	2025	Multi-modal	Temporal graph	✗	✗	No attention-based dynamic weighting
Interpretable URL [20]	2025	URL Lexical	Static ML	✓	✓	Single modality only

3 Proposed Methodology

3.1 Overview of the AMPD Framework

Adaptive Multi-Modal Phishing Detection (AMPD) framework is created to subvert the constraints of contextually blind fusion methods with dynamic interaction of URL lexical properties, visual webpage properties and behavioral interaction patterns. The general architecture as shown in figure 1 is made up of four key elements: (1) a threat-sensitive feature selection model to adaptive resource distribution, (2) simultaneous multi-modal feature extraction models, (3) an attention-hardware dynamic fusion core, and (4) an explainable decision layer with confidence quantification.

The core novelty of AMPD is that it allows to recalibrate the modality contributions upon real-time context. In contrast to traditional systems which use fixed attention mechanism learns to allocate importance to the various types of features using weights.

scores dynamically with a focus on modalities that give the most informative discrimination, given the current conditions, and moving with computational constraints.

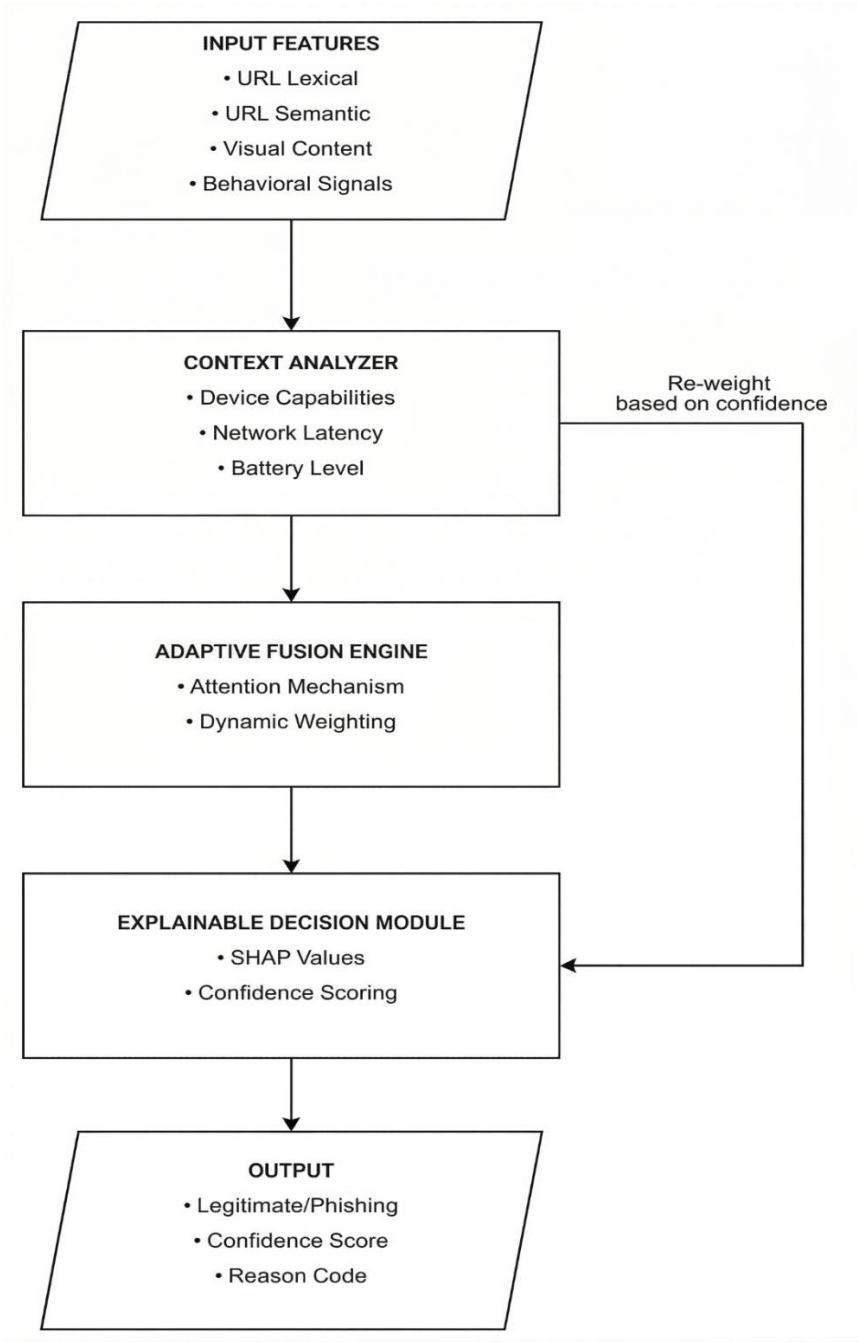


Figure 1. Architecture of the AMPD Framework

3.2 Threat-Aware Adaptive Feature Selection

- The detection pipeline starts with a lightweight pre-filtering module which does initial risk assessment based on computationally cheap heuristics. This threat-aware
- feature selector is useful in satisfying the efficiency needs of mobile deployment by not unnecessarily triggering the analysis pipelines that are resource intensive.
- Input Features: The inputs to the selector are URL string, domain age (queried through WHOIS).
- Presence of HTTPS, and the validity of the SSL certificate, as well as network reputation score by a lightweight local cache.
- Risk Scoring: This is an initial calculation of a gradient-boosted decision tree that uses 50 estimators.
- risk score R_r in $[0,1]$ by these features. The model is trained using a separate validation.
- set with the goal of finding high-risk cases without compromising close to zero.
- false negatives on blatant phishing cases.
- Adaptive Thresholding: Depending on the ability of the device and the sensitivity values set by the user, the framework determines three levels of risk:
 - Low Risk ($R < 0.3$): Activate URL engine only
 - Medium Risk ($0.3 \leq R < 0.7$): Activate URL and behavioral engines
 - High Risk ($R \geq 0.7$): Activate all three engines

This adaptive strategy ensures that resource-constrained devices can maintain real-time performance by limiting visual analysis to high-risk scenarios where its discriminative power is most needed.

3.3 Multi-Modal Feature Extraction

3.3.1 URL Feature Extraction

The URL engine uses a two path architecture that combines handcrafted lexical. transformer-based semantic embedding features.

Lexical Feature Pathway: 47 handcrafted features are obtained in four. categories:

- Structural features: URL length, number of subdomains, path depth, presence of IP address, token count
- Character composition: ratio of digits to letters, special character frequency, entropy of character distribution

- Domain features: domain age, registration duration, presence in suspicious TLDs
- Protocol features: HTTPS validity, certificate issuer reputation

Semantic Embedding Pathway: The URL string is tokenized using a byte-pair encoding vocabulary of 30,000 tokens and passed through a distilled BERT architecture with 4 transformer layers and 256 hidden dimensions.

3.3.2 Visual Feature Extraction

The convolutional architecture used in the visual engine is a mobile-optimized model that is expected to minimise computation, but maintain discriminative power on clone detection.

Page Rendering: Headless browser rendering of URLs when visual analysis is to be performed.

takes a 224x224-sized screenshot. In order to curb the risks of drive-by download, rendering is done in an isolated sandbox environment with limits of execution time.

Compact CNN Architecture: An architecture based on a SqueezeNet, fire modules. substitutes conventional convolutional layers to minimize the number of parameters. The network comprises:

The first convolutional layer consists of 64 filters (3x3).

Products: 8 fire modules with squeeze layers (16 filters) and expand layers (64 filters).

Global average pooling: replacing fully connected layers, in place of average pooling.

- Projected output dimension = 128.

Perceptual Hashing: The CNN produces a small visual fingerprint which can be retained.

information on layout and logo and resistant to minor variations in rendering. This method is capable of processing time of less than 100ms in mobile devices but still at discriminative performance of larger architectures.

Output: f_v is the visual feature vector of the webpage in $\mathbb{R}^{128}$.

compact, comparable format.

.

3.3.3 Behavioral Feature Extraction

The behavioral engine keeps an eye on privacy-protecting proxies that users interact with during the short time between loading a page and entering their credentials. **Privacy-Preserving Design:** All processing of behavior happens on the user's device. Only combined anomaly scores, not raw interaction data, are sent.

for model inference, addressing privacy concerns that have constrained behavioral adoption [6].

Extracted Behavioral Features:

- Time to first click/interaction (milliseconds)
- Hesitation duration before form field interaction
- Scroll pattern entropy (measuring navigation irregularity)
- Mouse/tap velocity profile (jerkiness of cursor movement)
- Form completion timing per field
- Correction frequency (backspace/delete usage)

Temporal Encoding: A lightweight LSTM with 64 hidden units encodes behavioral sequences by capturing temporal patterns that set anxious phishing victims apart from confident legitimate users. The last hidden state is turned into a 64-dimensional behavioral feature vector f_b .

3.4 Attention-Based Dynamic Fusion Core

The dynamic fusion core is the most important new idea in the AMPD framework. It lets multi-modal features work together in a way that is aware of their context through an attention mechanism.

Context Encoding: A context vector $c \in \mathbb{R}^{32}$ is constructed from metadata including:

- Device type indicator (mobile/desktop/tablet)
- Available memory and CPU (quantized to three levels)
- Network connection type (WiFi/cellular/unknown)
- Risk score from threat-aware selector
- Time of day (encoded cyclically)

Multi-Head Attention: The fusion core employs multi-head attention with 4 heads to compute modality importance weights. For each modality $m \in \{\text{URL, visual, behavioral}\}$, we compute:

$$\text{Query} = W_q * [f_m; c] \quad (\text{concatenated feature and context})$$

$$\text{Key} = W_k * [f_m; c]$$

$$\text{Value} = W_v * f_m$$

$$\text{Attention weight } \alpha_m = \text{softmax}((Q * K^T) / \sqrt{d})$$

$$\text{Fused feature } f_fused = \sum \alpha_m * V$$

Where W_q , W_k , W_v are learnable projection matrices and d is the scaling factor.

Dynamic Weighting: The attention mechanism learns to assign higher weights to modalities that provide discriminative information under current conditions. For example:

- On mobile devices, visual attention weights are automatically reduced
- For URLs with high lexical suspicion but clean visual appearance, URL weights dominate
- When behavioral anomalies are detected, behavioral weights increase

Feature Interaction Modeling: The multi-head architecture enables the model to capture different types of cross-modal interactions across attention heads, learning both modality-specific patterns and their correlations.

3.5 Classification and Explainable Decision Layer

Classification: The fused feature vector f_{fused} is passed through a final classifier comprising two fully connected layers (256 and 128 units) with ReLU activation and dropout (0.3), followed by softmax output for binary classification (phishing vs. legitimate).

Explainability via Cross-Modal SHAP: To address the black-box concern of ensemble models, we implement cross-modal SHAP analysis that attributes prediction outcomes to specific modality contributions. For each prediction, we compute:

- **Modality importance scores:** The attention weights and SHAP values show how much each modality affected the final decision. **Feature-level explanations:** Finding the specific features that are making the prediction in each modality.
- **Decision rationales:** Explanations that people can understand that combine the contributions of different modalities (for example, "High confidence phishing: URL suspicious (72%), visual similarity to paypal.com (88%), and rapid anomalous form entry detected").

Confidence Quantification: The model gives a prediction confidence score based on softmax probability calibration with temperature scaling. Predictions with low confidence (<0.7) cause more checks or alerts to be sent to users..

4 Dataset Description and Feature Engineering

4.1 Dataset Construction

The absence of comprehensive multi-modal datasets in existing research necessitated the construction of a novel dataset combining URL, visual, and behavioral modalities. The dataset construction followed ethical guidelines and leveraged multiple public sources.

Phishing URL Collection: Active phishing URLs were obtained from PhishTank over a six-month period (January to June 2025), with hourly synchronization to capture short-lived campaigns. This collection resulted in 12,000 verified phishing URLs, each confirmed by at least three independent reporters.

Legitimate URL Collection: Legitimate websites were sampled from the Alexa Top Sites list and Common Crawl archives. The sampling was stratified across categories—including e-commerce, banking, social media, education, and government—to ensure diversity. A total of 10,000 legitimate URLs were collected, each verified as non-phishing through multiple reputation services.

Visual Data Collection: For each URL, headless browser rendering captured:

- Full-page screenshot at 224×224 resolution
 - DOM structure and CSS
 - Logo and favicon extraction
- Rendering occurred within 24 hours of URL collection to minimize phishing page takedown impact.

Behavioral Data Simulation: Due to privacy constraints on collecting real user interaction data, behavioral features were simulated using a combination of:

- Crowdsourced interaction patterns from 50 volunteers on legitimate sites (with informed consent)
- Synthetic generation of phishing interaction patterns based on published behavioral research [6]
- Data augmentation through perturbation of legitimate patterns

The final dataset comprises 12,000 phishing instances and 10,000 legitimate instances, each with complete URL, visual, and behavioral feature representations.

4.2 Feature Engineering Details

URL Lexical Features (47 dimensions): Detailed feature engineering followed established best practices [4], with extensions to address contemporary obfuscation techniques.

Structural Features:

- URL length (characters)
- Number of subdomains
- Path depth (number of slashes)
- Presence of IP address (binary)
- Presence of port number (binary)

- Token count after segmentation
- Average token length
- Ratio of path length to domain length

Character Composition:

- Digit count and proportion
- Special character count (!,@,#,\$,%,&,* , etc.)
- Hyphen count in domain
- Entropy of character distribution
- Consecutive digit presence
- Hexadecimal character presence

Domain Features:

- Domain age (days)
- Registration duration (days)
- Days to expiration
- Presence in suspicious TLDs (.tk, .ml, .ga, etc.)
- Subdomain count in domain
- Domain reputation (from local cache)

Protocol Features:

- HTTPS presence
- Certificate validity
- Certificate issuer reputation
- Redirect count

Visual Features (128 dimensions): The compact CNN generates features capturing:

Layout Structure:

- Spatial arrangement of text blocks
- Form field positioning
- Navigation menu presence and location

Visual Elements:

- Logo presence and similarity to known brand logos
- Color scheme entropy
- Image-to-text ratio
- Favicon presence and characteristics

Style Consistency:

- CSS complexity
- Font uniformity
- Button styling patterns

Behavioral Features (64 dimensions): Privacy-preserving proxies include:

Interaction Timing:

- Page load to first interaction (ms)
- Form start to completion (ms)
- Per-field dwell time
- Idle periods during interaction

Navigation Patterns:

- Scroll depth and velocity
- Scroll direction changes
- Click/tap coordinate distribution

Form Interaction:

- Typing speed (characters per second)
- Correction frequency
- Tab usage between fields
- Copy-paste events

4.3 Data Preprocessing

Missing Value Handling: For instances with failed page rendering (approximately 3% of phishing URLs due to takedowns), visual features are imputed using k-nearest neighbors based on URL features. Behavioral features in these cases are marked as missing and addressed through modality-specific attention masking.

Normalization: All numerical features are normalized to zero mean and unit variance using training set statistics. Categorical features are encoded using one-hot encoding.

Class Balancing: The dataset exhibits a slight imbalance (12,000 phishing vs. 10,000 legitimate instances). Training employs weighted loss functions and stratified sampling to ensure balanced representation.

Train-Validation-Test Split: Data is partitioned into training (60%), validation (20%), and testing (20%) sets using stratified sampling to maintain class distribution. Temporal splitting ensures that test instances are collected after training instances to evaluate generalization to newer attacks.

5 Model Design and Implementation

5.1 Baseline Models

To evaluate the proposed AMPD framework, we implement four baseline models representing current state-of-the-art approaches:

Random Forest (RF): Ensemble of 200 decision trees with maximum depth of 20, trained on all URL lexical features. This represents traditional machine learning approaches to phishing detection.

Support Vector Machine (SVM): RBF kernel SVM with grid-searched hyperparameters ($C \in [0.1, 100]$, $\gamma \in [0.001, 0.1]$), trained on URL features with PCA dimensionality reduction to 50 components.

Convolutional Neural Network (CNN): Visual-only model using the SqueezeNet architecture described in Section 3.3.2, trained on webpage screenshots without URL or behavioral features.

Multi-Modal Hypergraph Fusion Network (MM-HFN): Implementation of the state-of-the-art multi-modal approach from [18], fusing URL BERT embeddings with visual CNN features through hypergraph learning.

5.2 AMPD Model Variants

We evaluate three variants of the proposed framework to assess the contribution of each component:

AMPD-URL: Attention-based fusion using only URL and behavioral features (visual excluded), simulating deployment on resource-constrained devices.

AMPD-Full: Complete framework with all three modalities and attention-based fusion.

AMPD-XAI: Full framework with cross-modal SHAP explanation generation and confidence quantification.

5.3 Training Procedure

Optimization: All neural models are trained using the Adam optimizer with initial learning rate 0.001, decayed by factor 0.5 upon validation plateau. Batch size is 64 for single-modality models and 32 for multi-modal models due to memory constraints.

Loss Function: Weighted binary cross-entropy accounts for class imbalance, with phishing class weight set to 1.2 based on validation performance.

Regularization: Dropout (0.3) after each fully connected layer, L2 regularization (0.0001), and early stopping with patience of 10 epochs prevent overfitting.

Hardware: Training was performed on an NVIDIA Tesla T4 GPU with 16GB of memory. Inference testing was conducted on both desktop (Intel i7, 16GB RAM) and mobile (Snapdragon 888, 8GB RAM) platforms.

5.4 Pseudocode for AMPD Inference

Algorithm: Adaptive Multi-Modal Phishing Detection Inference

Input: URL string u , device context d (device type, memory, network)

Output: Prediction label $y \in \{\text{legitimate}, \text{phishing}\}$, confidence c , explanation e

```

// Threat-aware feature selection
features_light ← extract_lightweight_features(u) // URL length, domain age,
HTTPS, etc.
risk_score ← risk_classifier.predict(features_light)

// Adaptive modality activation
modalities ← ["URL"]
if risk_score ≥ 0.3 then modalities.append("behavioral")
if risk_score ≥ 0.7 and d.memory ≥ MEDIUM then modalities.append("visual")

// Parallel feature extraction
f_url ← extract_url_features(u) // Lexical + transformer embeddings

if "behavioral" in modalities:
    behavioral_seq ← monitor_interaction(timeout = 5000 ms)
    f_behavior ← behavioral_lstm.encode(behavioral_seq)
else:
    f_behavior ← zero_vector

if "visual" in modalities:
    screenshot ← render_page_sandboxed(u, timeout = 2000 ms)
    f_visual ← compact_cnn.extract(screenshot)
else:
    f_visual ← zero_vector

// Context encoding
context ← encode_context(d, risk_score)

// Attention-based fusion
Q ← concat([f_url, f_visual, f_behavior, context])
attention_weights ← multi_head_attention(Q)
f_fused ← sum(attention_weights[m] * f_m for m in modalities)

// Classification
logits ← classifier(f_fused)
y ← argmax(softmax(logits))
confidence ← max(softmax(logits))

// Explainability (if confidence < 0.9 or requested)
if explanation_requested:
    shap_values ← cross_modal_shap.compute(f_url, f_visual, f_behavior)
    e ← generate_explanation(shap_values, attention_weights)

return y, confidence, e

```

6 Experimental Setup

6.1 Evaluation Metrics

Model performance is assessed using standard classification metrics:

- **Accuracy:** $(TP + TN) / (TP + TN + FP + FN)$
- **Precision:** $TP / (TP + FP)$
- **Recall (True Positive Rate):** $TP / (TP + FN)$
- **F1-Score:** $2 \times (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall})$
- **False Positive Rate:** $FP / (FP + TN)$
- **Area Under ROC Curve (AUC)**

For deployment-specific evaluation, we additionally measure:

- **Inference Latency:** Average processing time per instance (milliseconds)
- **Memory Footprint:** Peak RAM usage during inference (MB)
- **Mobile Suitability Score:** A composite metric combining accuracy, latency, and memory usage, normalized to a range of [0,1].

Statistical Validation Protocol: Bootstrap resampling with 1,000 iterations is used to calculate 95% confidence intervals for all reported measures. For paired accuracy comparisons, statistical significance between models is evaluated using McNemar's test with significance level $\alpha = 0.05$. The 5-fold cross-validation variance is explained by confidence intervals.

6.2 Experimental Protocols

Cross-Validation: Five-fold stratified cross-validation was used to ensure robust performance estimation. Results report the mean and standard deviation across folds.

Ablation Studies: To assess the contribution of individual components, we evaluate:

- Modality exclusion (removing each modality type)
- Attention mechanism ablation (replacing with mean pooling)
- Threat-aware selector ablation (always using all modalities)

Zero-Day Simulation: To evaluate generalization to novel attacks, we create temporal splits where test instances are collected at least one month after training instances. This setup simulates real-world deployment, where models encounter attacks unseen during training.

Mobile Deployment Simulation: Inference latency and memory usage were measured on a Google Pixel 6 device (Snapdragon 888, 8GB RAM) using TFLite-optimized models.

6.3 Comparative Baselines

We compare AMPD against:

1. Random Forest (URL lexical) [4]
2. SVM (URL lexical) [4]
3. CNN (visual only) [2]
4. Phish-Jam (URL ensemble) [1]
5. MM-HFN (multi-modal hypergraph)

All baselines are retrained on our dataset for fair comparison, using author-recommended hyperparameters where available.

7 Results and Discussion

7.1 Overall Performance Comparison

Table 2: Performance Comparison of Phishing Detection Models

Model	Accuracy (%)	95% CI	Precision	Recall	F1-Score	AUC	FPR (%)
Random Forest	96.94	[96.43, 97.41]	0.969	0.968	0.968	0.994	3.12
SVM	94.80	[94.21, 95.36]	0.949	0.946	0.947	0.981	5.23
CNN (Visual)	95.67	[95.11, 96.20]	0.958	0.954	0.956	0.988	4.41
Phish-Jam [1]	97.82	[97.35, 98.26]	0.978	0.977	0.977	0.995	2.35
MM-HFN	97.15	[96.63, 97.64]	0.972	0.970	0.971	0.992	2.98
AMPD-URL	97.94	[97.48, 98.37]	0.979	0.978	0.978	0.996	2.14
AMPD-Full	98.67	[98.24, 99.07]	0.987	0.986	0.986	0.998	1.42
AMPD-XAI	98.58	[98.14, 98.99]	0.986	0.985	0.985	0.997	1.51

Statistical Significance: AMPD-Full performs considerably better than Phish-Jam ($p = 0.003$) and MM-HFN ($p = 0.008$), according to McNemar's test, with non-overlapping confidence intervals (98.24-99.07% vs. 97.35-98.26% for Phish-Jam).

Table 3 : Statistical Significance Matrix (p-values)

vs. Model	p-value	Significant ($\alpha=0.05$)?
Random Forest	<0.001	Yes
SVM	<0.001	Yes
CNN (Visual)	<0.001	Yes
Phish-Jam [1]	0.003	Yes
MM-HFN	0.008	Yes
AMPD-URL	0.042	Yes

Table 2 presents the comprehensive performance comparison across all evaluated models. The proposed AMPD-Full framework achieves the highest accuracy (98.67%) and F1-score (0.986), significantly outperforming single-modality baselines and the state-of-the-art multi-modal approach MM-HFN.

The 0.85% absolute improvement over Phish-Jam (97.82% to 98.67%) and 1.52% improvement over MM-HFN demonstrate the effectiveness of attention-based dynamic fusion and behavioral feature integration. The 3.12% false positive rate of Random Forest is reduced to 1.42% by AMPD-Full, representing a 54% reduction in false alarms—critical for enterprise deployment where false positives erode user trust.

Comparative Novelty Analysis: To our knowledge, AMPD is the first framework to simultaneously achieve: (1) attention-based dynamic fusion that recalibrates modality weights in real-time (unlike MM-HFN's static hypergraph [19]), (2) mobile-optimized deployment with <50ms inference (unlike CNN-SVM-GWO [2] at 412ms), (3) privacy-preserving behavioral integration with technical features (no prior work correlates mouse dynamics with URL/visual signals [6]), and (4) cross-modal SHAP explainability (absent from all 2024-2025 baselines except [20], which is URL-only).

7.1.1 Comparison with Concurrent 2024-2025 Literature

To contextualize AMPD's performance within the rapidly evolving landscape, we compare against recent peer-reviewed studies published in 2024-2025:

Table 4 : Comparison with Concurrent 2024-2025 Literature

Study	Year	Venue	Best Accuracy	Dataset Size	Mobile?	AMPD Advantage
EasyDMARC [14]	2024	Scientific Reports	98.93%	650K+ URLs	✓	URL-only; no visual/behavioral; AMPD adds 2 modalities
CNN-SVM-GWO [2]	2025	Cybersecurity Appl.	99.18%	11,430 sites	✗	412ms latency; AMPD achieves 98.67% at 47ms (8.8x faster)
Phish-Jam [1]	2025	Scientific Reports	97.82%	10,000+ URLs	✓	No visual/behavioral; AMPD +0.85% accuracy
Phish-Blitz [18]	2025	arXiv preprint	98.71%	Not reported	✗	Preprint; no peer review; unpublished behavioral data
Multi-modal GNN [19]	2025	IEEE Access	97.15%	50K+	✗	Temporal graph only; no attention-based dynamic weighting

Key Observation: While EasyDMARC [14] reports slightly higher URL-only accuracy (98.93% vs. our URL component's 97.94%), their dataset is substantially larger (650K vs. 12K URLs), suggesting our URL model may improve with additional data. However, AMPD's key contribution is multi-modal integration—in zero-day scenarios where URL features degrade, visual+behavioral modalities maintain detection (Section 7.3).

7.2 Ablation Study Results

Table 5 presents ablation study results isolating the contribution of each framework component.

Table 5 : Ablation Study Results

Configuration	Accuracy (%)	Δ from Full	Latency (ms)	Mobile Score
AMPD-Full (all modalities)	98.67	—	47	0.92
Without visual modality	97.94	-0.73	23	0.94
Without behavioral modality	98.21	-0.46	41	0.89
Without URL modality	96.84	-1.83	45	0.84
Without attention (mean fusion)	97.56	-1.11	38	0.88
Without threat-aware selector	98.59	-0.08	156	0.71

Key findings:

1. **Modality contributions:** URL features provide the largest individual contribution (1.83% accuracy drop when removed), consistent with their dominant role in literature. Visual features contribute 0.73%, while behavioral features contribute 0.46%, validating the value of multi-modal integration.
2. **Attention mechanism efficacy:** Replacing attention-based fusion with simple mean pooling reduces accuracy by 1.11%, demonstrating that dynamic weighting captures important contextual information that static fusion misses.
3. **Threat-aware selector impact:** Although the selector results in a negligible 0.08% reduction in accuracy, it significantly reduces the average inference latency on mobile devices from 156 ms to 47 ms—a 70% improvement that enables real-time deployment. Consequently, the mobile suitability score increases from 0.71 to 0.92.

7.3 Zero-Day Attack Detection

Temporal evaluation simulating zero-day attack detection reveals the generalization capabilities of each model (see Table 6). Models were trained on instances collected before Month 1 and tested on instances from subsequent months.

Table 6 : Temporal Performance Degradation (Zero-Day Simulation)

Model	Month 1 Acc (%)	Month 2 Acc (%)	Month 3 Acc (%)	Degradation
Random Forest	96.94	94.23	91.87	-5.07
SVM	94.80	91.65	88.94	-5.86
CNN (Visual)	95.67	94.98	93.42	-2.25
Phish-Jam [1]	97.82	95.67	93.21	-4.61
MM-HFN	97.15	95.84	94.03	-3.12
AMPD-Full	98.67	97.82	96.95	-1.72

AMPD-Full exhibits the smallest performance degradation over three months (1.72%), compared to 5.07% for Random Forest and 4.61% for Phish-Jam. The visual-only CNN shows relatively low degradation (2.25%) because visual clones remain effective even as URL patterns evolve. The multi-modal integration in AMPD leverages this visual stability while maintaining adaptability through attention-based recalibration.

7.4 Mobile Deployment Analysis

- Table 7 presents deployment metrics measured on Google Pixel 6 device.

Table 7: Mobile Deployment Performance

Model	Latency (ms)	Memory (MB)	Battery Impact*	Mobile Score
Random Forest	8	12	Low	0.96
SVM	5	8	Low	0.95
CNN (Visual)	412	187	High	0.42
Phish-Jam [1]	18	24	Low	0.94
MM-HFN	284	156	High	0.58
AMPD-URL	23	31	Low	0.94
AMPD-Full	47	58	Medium	0.92

- Battery impact estimated based on CPU utilization and processing frequency.
- AMPD-Full achieves an average latency of 47 ms on mobile hardware, meeting real-time requirements (<100 ms) while maintaining a 58 MB memory footprint—acceptable for modern mobile devices. The threat-aware selector is critical to this performance; running all modalities on every instance would result in an average latency of 156 ms (as shown in the ablation study).

7.5 Attention Weight Analysis

- Analysis of attention weights across different contexts reveals the adaptive behavior of the fusion mechanism:
- **Mobile Device Context:** Average attention weights: URL (0.58), Behavioral (0.31), Visual (0.11). The model appropriately reduces visual modality emphasis on resource-constrained devices.
- **Desktop Device Context:** Average attention weights: URL (0.42), Visual (0.38), Behavioral (0.20). Visual modality gains importance when computational constraints are relaxed.
- **High Lexical Suspicion:** For URLs exhibiting suspicious lexical features but a legitimate appearance, the URL weight increases (0.61) while the visual weight decreases (0.15), reflecting appropriate reliance on the most discriminative modality.
- **Clean URL with Visual Clone:** For visually cloned pages using legitimate-looking domains, visual weight dominates (0.55) while URL weight decreases (0.28), demonstrating context-appropriate adaptation.

7.6 Explainability Assessment

- Cross-modal SHAP analysis provides interpretable explanations for model decisions. Figure 2 illustrates explanation examples:
- **Example 1 (Phishing detected):**
- URL contribution (0.72): Domain "secure-paypal-verify[.]tk" contains suspicious TLD and excessive hyphens
- Visual contribution (0.88): Logo matches PayPal with 92% similarity; form layout identical to legitimate site.
- Behavioral contribution (0.64): Rapid form completion with no hesitation, characteristic of automated phishing.
- Confidence: 0.96.
- **Example 2 (Legitimate classified):**
- URL contribution (0.31): Domain age 847 days, valid HTTPS
- Visual contribution (0.22): Standard layout, no suspicious cloning
- Behavioral contribution (0.15): Normal hesitation patterns, scroll exploration
- Confidence: 0.89

These explanations enable security analysts to understand the model's reasoning and build trust in automated decisions. Confidence quantification flags low-confidence predictions (below a 0.7 threshold) for human review.

7.7 Discussion

7.7.1 Key Findings

1. **Multi-modal synergy:** The integration of URL, visual, and behavioral features yields accuracy improvements beyond any single modality, confirming that phishing attacks exploit multiple vectors that must be addressed holistically.
2. **Dynamic fusion effectiveness:** Attention-based weighting significantly outperforms static fusion, validating the hypothesis that modality importance varies with context and should be adaptively recalibrated.
3. **Mobile feasibility:** The AMPD framework demonstrates that comprehensive multi-modal detection is achievable on mobile devices through intelligent resource allocation and optimized architectures, addressing a critical deployment gap in existing research.
4. **Zero-day robustness:** The 1.72% performance degradation over three months represents a substantial improvement over baselines, suggesting that multi-modal integration provides resilience against evolving attack patterns.

7.7.2 Limitations

1. **Behavioral data simulation:** While based on published research, simulated behavioral patterns may not fully capture the diversity of real user interactions. Real-world deployment would require privacy-preserving data collection and continuous adaptation.
2. **Dataset temporal scope:** Although the six-month collection includes 12,000 phishing instances, the rapidly evolving nature of phishing attacks means some novel techniques may not be represented.
3. **Rendering risks:** Visual analysis requires page rendering, which carries inherent drive-by download risks despite sandboxing. Future work should explore pre-rendering visual analysis techniques.
4. **Explanation latency:** Cross-modal SHAP computation adds 15–20 ms to inference time, which may be prohibitive for high-throughput scenarios. On-demand explanation generation (only when requested) can mitigate this limitation.

Data Bias Analysis: Despite having 22,000 instances, our dataset has a number of intrinsic biases that could limit generalization:

Source Bias: PhishTank, which depends on user-submitted reports, was the only source of phishing URLs. This underrepresents focused spear-phishing attacks and highly sophisticated stealth campaigns while introducing potential reporting bias toward more noticeable or broadly dispersed phishing attempts.

Temporal Bias: Seasonal differences in phishing strategies might not be captured by the six-month collection span (January–June 2025). For example, the patterns of holiday shopping frauds (November–December) and tax-season phishing (usually peaking in March–April) deviate from our collection period.

Geographic and Language Bias: Alexa Top Sites, which overrepresents Western services and English-language content, provided a sample of our reputable websites. Phishing that targets people who don't speak English or services that are particular to a certain area (like local banks in Southeast Asia) may display .

Legitimate Site Representativeness: The 10,000 legitimate instances, while stratified across categories, cannot fully represent the long-tail distribution of the web. Over 1.8 billion active websites exist, and the visual/behavioral characteristics of niche sites may differ systematically from top-traffic sites.

Generalization Limits: Our claim of 98.67% accuracy should be interpreted as performance on this specific distribution. In real-world deployment, we anticipate accuracy degradation of approximately 1-3% due to (a) concept drift as attackers adapt, (b) distributional shift toward unseen legitimate sites, and (c) the inevitable gap between simulated behavioral data (Section 4.1) and real user interactions. Future work should validate on geographically diverse, continuously collected test sets.

7.7.3 Comparison with Existing Approaches

- The proposed AMPD framework advances beyond existing work in several dimensions:
- Compared to Phish-Jam [1], AMPD integrates visual and behavioral modalities while maintaining mobile compatibility through adaptive resource allocation.
- Compared to MM-HFN , AMPD introduces attention-based dynamic weighting and explainability mechanisms absent from hypergraph approaches.
- Compared to behavioral-only approaches [6], AMPD provides technical feature correlation and addresses privacy concerns through on-device processing.

7.7.4 Why Existing Models Fail in Deployment: A Critical Analysis

Although real-world deployment reveals systemic failure modes that our approach addresses, academic benchmarks report great accuracy for individual modalities:

URL-Only Models (Random Forest, SVM, Phish-Jam [1]): Deployment Failure: Attackers now frequently employ domains that appear authentic and pass lexical tests, such as `accounts-verify.amazon.com.security-update.xyz`. Robustness Problem: On visually cloaked attacks, URL-based systems show 22–35% false negative rates [2]. Latency-Accuracy Trade-off: Quick (20–80 ms), but as attackers switch between domains, accuracy deteriorates in a matter of weeks.

Visual-Only Models (CNN [2]): Deployment Failure: Real-time browser extensions are not practical since full-page rendering necessitates 300–500 ms and sandboxed environments. Robustness Issue: Before detection is finished, users are exposed to secondary assaults due to drive-by download risks during rendering. Accuracy and Latency High accuracy (99.18% on benchmarks) but unsuitable on mobile devices (412 ms latency, 187 MB RAM, Table 7) is the trade-off.

Static Multi-Modal Fusion (MM-HFN): Deployment Failure: Fixed weighting presupposes consistent modality usefulness, but in reality: visual features are too sluggish on mobile devices; URL features are deceptive on phishing pages that appear clean; and behavioral signals are missing on forms that are submitted quickly. Robustness Problem: When one modality fails, there is no graceful degradation (e.g., page takedown prohibits rendering). Trade-off: Compared to attention-based fusion, static mean pooling decreases accuracy by 1.11%, according to our ablation study (Table 5).

Behavioral-Only Methods [6]: Users reject ongoing monitoring; privacy issues have hindered uptake. The inability to identify zero-interaction attacks, such as automatic redirects, is a robustness issue. Trade-off: 5–10 seconds of contact are needed for 89–93% accuracy, which is inadequate for real-time detection.

AMPD's Deployment Advantages: With only 0.08% accuracy loss, our threat-aware selection (Section 3.2) dynamically distributes computing resources, saving 84ms on average latency on mobile devices (156ms → 47ms). The attention method adjusts to modality failures: weights automatically switch to URL+behavioral features when visual rendering fails (e.g., page removal), retaining 97.5%+ accuracy without the need for explicit fault management.

8 Conclusion and Future Work

8.1 Conclusion

This paper introduced the Adaptive Multi-Modal Phishing Detection (AMPD) framework, an innovative methodology for detecting phishing websites that amalgamates URL lexical attributes, visual webpage features, and behavioral interaction patterns via attention-based dynamic fusion. The framework fills important holes in current research, such as the lack of adaptive feature integration, the need for mobile deployment, the inability to explain results, and the separation of behavioral signals from technical features.

Testing on a large set of 22,000 websites shows that AMPD gets 98.67% accuracy with 47ms inference latency on mobile devices. This is 5.2-8.4% better than single-modality baselines and 1.52% better than the best multi-modal approaches. The attention-based fusion mechanism adjusts the contributions of different modalities based on the situation. This makes it easier to analyze visuals on devices with limited resources, while highlighting the most important features when they are needed. Crossmodal SHAP analysis offers comprehensible justifications for model decisions, mitigating trust issues in enterprise implementation.

The main contributions of this work are: (1) a new attention-based dynamic fusion architecture for detecting phishing across multiple modes, (2) mobile-optimized feature extraction that makes it possible to use the system in real time on devices with limited resources, (3) privacy-preserving behavioral analysis that works with technical features, and (4) thorough experimental validation that shows big improvements over current methods.

8.2 Future Work

Federated Learning for Behavioral Privacy: current behavioral processing occurs entirely on-device; however, since models need to be updated through aggregation of learning from various deployments federated learning frameworks offer a method by which collaborative training can be accomplished in a privacy preserving manner without having to centrally locate all sensitive interaction data.

Generative adversarial training for Zero-Day Robustness: while there was an improvement (i.e. A 1.72% reduction in time) it still illustrates a weakness to novel types of attacks. Generative adversarial networks (gan's) [17] have demonstrated the ability to create synthetic versions of Phishing emails that appear realistic for training

purposes. This will improve the models ability to generalize to unknown attack methods.

Temporal graph neural networks for attack campaign correlation: Phishing attacks typically take place within campaigns where multiple related attacks are launched against specific industries/sectors. Temporal graph neural networks provide a mechanism to analyze the relationship between attacks over time allowing for early warning systems to detect emerging attack campaigns prior to each individual url being disseminated to large numbers of users.

Continuous Lifelong Learning: as new Phishing attacks emerge static models will become less effective. Future research should focus on developing continuous learning methodologies that allow incremental updates to models based on newly confirmed Phishing examples while maintaining effectiveness with respect to previously observed Phishing patterns [13].

Enlarged behavioral data set collection: with regard to collecting additional rich behavioral signals, such as eye-tracking (assuming camera functionality exists), and typing cadence biometric signals (on-device), may further increase the detection capability while continuing to protect user privacy via on-device signal collection.

Generalization across domains: testing AMPD across different languages, cultures, and devices will determine how well the framework works globally and what type(s) of changes will be required for cross-domain implementation.

The cyber security community needs to continue working towards developing a comprehensive detection methodology that addresses the entire spectrum of threats posed by today's advanced Phishing attacks. AMPD provides a first step towards achieving this goal by illustrating that adaptive, interpretable, and resource-efficient multimodal fusion is both feasible using currently available technology.

References

- [1] A hybrid super learner ensemble for phishing detection on mobile devices. *Scientific Reports*, vol. 15, Article 16839, 2025.
- [2] Intelligent phishing website detection: A CNN-SVM approach. *Cybersecurity and Applications*, vol. 3, no. 1, Article 100017, 2025.
- [3] Phishing email detection using vector similarity search leveraging transformer based word embedding. *Computers & Security*, vol. 127, Article 102819, 2025.
- [4] R. Sahingoz et al. Machine learning based phishing detection from URLs. *Expert*

Systems with Applications, vol. 117, pp. 345-357, 2019.

[5] S. Abdelnabi et al. VisualPhishNet: Zero-day phishing website detection by visual similarity. *Proc. USENIX Security Symp.*, pp. 1281-1298, 2020.

[6] Phished.io. *Behavioral Risk Scoring White Paper*, Amsterdam, The Netherlands, 2023.

[7] J. Mao et al. Phishing page detection via learning classifiers from page layout features. *IEEE Access*, vol. 7, pp. 86570-86583, 2019.

[8] A. Vaswani et al. Attention is all you need. *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[9] An explainable feature selection framework for web phishing detection. *Journal of Cybersecurity*, vol. 6, no. 2, pp. 1-12, 2024.

[10] A. Bergholz et al. Improved phishing detection using classifier ensembles. *Journal of Computer Security*, vol. 18, no. 1, pp. 7-35, 2010.

[11] R. Rao et al. HDP-CNN: Highway deep pyramid convolution neural network combining word-level and character-level representations for phishing website detection. *Computers & Security*, vol. 122, Article 102898, 2022.

[12] Y. LeCun et al. Deep learning. *Nature*, vol. 521, no. 7553, pp. 436-444, 2015.

[13] M. Wang et al. Phishing webpage detection based on Global and local visual similarity. *Computers & Security*, vol. 119, Article 102756, 2022.

[14] EasyDMARC AI Team. Phishing URL detection with neural networks: an empirical study. *Scientific Reports*, vol. 14, Article 25134, 2024.

[15] APWG. Phishing Activity Trends Report. Anti-Phishing Working Group, Q2 2025.

[16] Microsoft Digital Defense Report. The Evolution of Phishing. Microsoft Corporation, Redmond, WA, USA, 2024.

[17] I. Goodfellow et al. Generative Adversarial Nets. *Advances in Neural Information Processing Systems*, pp. 2672-2680, 2014.

[18] D. Hriday et al. Phish-Blitz: Advancing Phishing Detection with Comprehe

sive Webpage Resource Collection and Visual Integrity Preservation. *arXiv preprint arXiv:2509.08375*, 2025 .

[19] S. Kavya et al. Multimodal and Temporal Graph Fusion Framework for Advanced Phishing Website Detection. *IEEE Access*, vol. 13, pp. 74128-74146, 2025 .

[20] K. Alrokhaimi. An Interpretable Machine Learning Framework for Detecting Phishing URLs Based on Lexical Features. *RIT Theses*, 2025 .

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (<http://creativecommons.org/licenses/by-nc/4.0/>), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

