International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 13 Issue: 11 | Nov 2026 www.irjet.net p-ISSN: 2395-0072

An AI-Driven Framework for Identifying Malicious QR Codes

Atharva Sawardekar[1], Abhishek Gosavi[2]

[12]*Department of Master of Computer Applications, D.Y. Patil University, Navi Mumbai, India*

Abstract - *QR codes conceal their destination until they are decoded, which makes them an attractive vector for phishing ("quishing"), malware delivery, and payment fraud. This paper proposes a safe and measurable framework for identifying suspicious QR codes before the user opens the destination, by combining QR-image analysis with analysis of the decoded URL. The framework separates decoding from navigation, normalizes the payload while preserving the raw value, extracts image and URL features, and fuses them in a calibrated risk engine that returns an explainable low-risk, needs-review, or high-risk decision. A review of recent image-based, URL-based, hybrid, and adaptive approaches highlights inconsistent evaluation protocols and the risk of domain- and URL-level leakage. The paper therefore defines research questions, hypotheses, a leakage-resistant evaluation protocol, metrics, and statistical tests. No newly trained model results are claimed; literature values are reported from the cited studies and the results table is left for measured values.*

Key Words: QR code security, Quishing, Phishing detection, Malicious URL, Machine learning, Hybrid model, Cybersecurity

1. INTRODUCTION

QR codes encode information in a two-dimensional visual pattern that can be read by a camera or scanning application. A single code can represent a web address, plain text, contact information, a payment instruction, an authentication token, or other machine-readable data. The low cost of generating and printing QR codes has accelerated their use across physical and digital environments. Users are typically encouraged to scan first and inspect the destination later, if at all.

This convenience creates an important trust gap. The visual appearance of a QR code usually does not tell the user whether its destination is legitimate. A malicious actor can replace a genuine sticker, place a deceptive code beside a trusted brand, or distribute an image through email, social media, messaging applications, posters, and advertisements. The victim may see only a familiar logo or a short instruction such as "scan to verify" while the actual destination remains hidden until the scan is performed. Common quishing strategies include look-alike domains, deceptive subdomains, shortened URLs, redirects, fake login forms, and counterfeit payment pages. Because the QR image itself can be harmless-looking, security decisions should consider both the image and the payload decoded from it.

Traditional URL blocklists and reputation systems remain valuable, but they struggle with newly registered domains, short-lived campaigns, and compromised legitimate websites. Artificial intelligence can complement these controls by combining multiple signals into a risk score. Nevertheless, a model prediction is probabilistic rather than proof of maliciousness; a safe system should communicate uncertainty and never open an untrusted destination merely to inspect it.

1.1 Research Aim

The aim of this research is to design a safe and measurable framework for suspicious QR-code detection by combining QR-image analysis with decoded-URL analysis. The study focuses on whether multimodal evidence can improve detection quality while keeping false alarms, latency, privacy risk, and deployment complexity within practical limits.

2. PROBLEM STATEMENT

The technical problem is to classify a QR code and its decoded destination as benign, suspicious, or high-risk before the user opens the destination. The classifier must handle legitimate long URLs, URL shorteners, redirects, IP-based URLs, unusual domain structures, visually degraded QR images, newly registered malicious domains, and changing attacker behavior.

A model trained on duplicated or closely related samples can produce inflated scores, so the research problem also includes constructing a reliable evaluation protocol in which related URLs and domains do not leak across training and test sets.

3. BACKGROUND AND THREAT MODEL

A QR security system should be designed around a realistic threat model. The attacker is assumed to control the QR image, the encoded URL, or the destination web content, but not the user's security software. The attacker may distribute codes through posters, stickers, invoices, emails, social-media posts, messaging applications, or counterfeit advertisements. The goal may be credential theft, payment fraud, malware delivery, tracking, or social engineering. The attacker may also attempt to evade detection by shortening URLs, using redirects, registering look-alike domains, changing landing pages, or generating many visually different QR codes for the same destination.

The defender has access to the QR image and, after safe decoding, the payload string. The defender may compute lexical features and use reputation information where lawful and available. A key safety rule is that decoding must be separated from navigation: the scanner should extract the URL as data and analyze it without automatically requesting the destination. If external reputation checks are used, they should follow privacy and rate-limit requirements and should not expose sensitive user data unnecessarily.

3.1 Typical Attack Indicators

- Hostname contains an IP address instead of a conventional domain name.
- Domain or path contains suspicious tokens associated with login, verification, payment, reward, or account-recovery scams.
- Excessive URL length, unusual delimiter frequency, encoded characters, or multiple nested parameters.
- Abnormally deep subdomain structure or a brand name placed inside an unrelated domain.
- Punycode or visually confusing domain representations.
- Use of URL shorteners that conceal the final destination.
- Unexpected redirect chains or destination changes when reputation information is available safely.
- QR images with distortion, overlays, deceptive labels, or placement designed to imitate a trusted code.

4. LITERATURE REVIEW

Recent research covers four broad directions: image-based QR classification, URL-feature machine learning, hybrid QR-and-URL systems, and adaptive models for changing malicious URL patterns. These approaches are complementary. Image models can learn visual representations, while URL models can exploit semantic and structural clues that are difficult to infer from the pixels alone. Hybrid methods attempt to combine both sources of evidence. Adaptive methods address the fact that malicious infrastructure changes over time. A systematic review of QR security attacks and countermeasures [7], a unified phishing and quishing attachment dataset [8], and a comparison of machine-learning and deep-learning models for QR phishing detection [9] provide further context.

Table -1: Selected Recent Studies and Reported Results

Study / Year	Approach / Data	Reported result	Research note
Recognition of Valid QR Codes with Machine Learning (2024) [1]	ResNet50, MobileNetV3 and InceptionV3; 10,000 QR images.	Inception V3: 98.13% accuracy.	Generalization depends on dataset construction and splitting.
Dataset of 1,000 Malicious and Benign QR Images (2025) [2]	500 malicious + 500 benign QR images; 330×330 pixels.	Dataset resource; no universal model score.	Small dataset; provenance and currentness must be checked.
Efficient Malicious QR Code Detection System (2025) [3]	AlexNet features + PCA + GRU/LSTM on two datasets.	GRU: 99.81% and 99.59%; 7.433 ms/sample reported.	High scores require leakage and independent-replication checks.
HybridStack-MLP (2025) [4]	KNN, DT, Gradient Boosting, RF and LR with MLP meta-model; URL/domain features.	91.30% accuracy reported.	Different features and data prevent direct comparison.
Malicious Software in QR Code (2026) [5]	CNN using 316,254 benign and 316,254 malicious URLs rendered as QR images.	99.62% validation; 100% test accuracy reported.	Very high scores require scrutiny of near-duplicates and split independence.
Adaptive Retraining for Dynamic URL Threat Detection (2026) [6]	Adaptive BERT-based malicious URL classification.	Focuses on evolving URL threats.	Drift, validation, versioning and governance remain important.

Table 1 summarizes selected studies reviewed for this paper. Reported scores are retained as literature findings and are not presented as results obtained by the authors. Dataset construction, label definitions, image-generation methods, domain overlap, class balance, and test protocols differ substantially across studies.

4.1 Comparison of Published Results

Fig. 1 plots the accuracy values and dataset sizes reported in the cited studies. The literature suggests that deep-learning methods can achieve strong performance on controlled datasets. However, a high accuracy value can be misleading when the same domain, URL, QR generator, or near-duplicate image appears in both training and testing. For a security application, recall and false-negative rate can be more important than accuracy alone because an undetected malicious code can have a higher practical cost than a benign code being flagged for review.

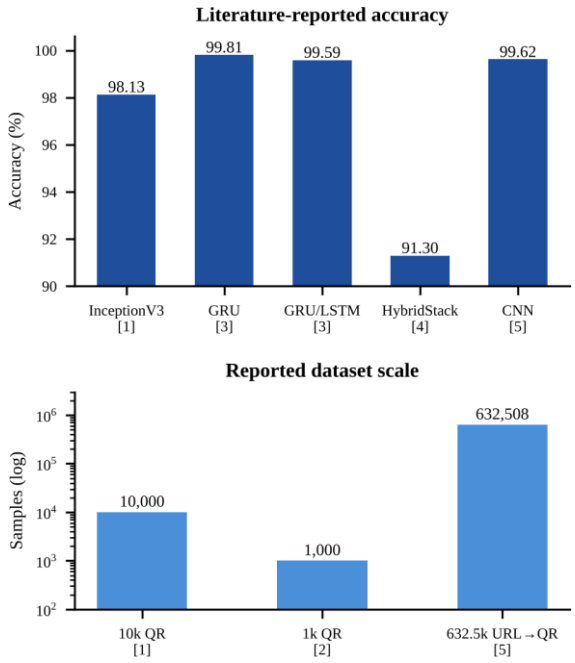


Fig -1: Selected accuracy values and dataset scales reported by the cited studies (mean 97.69%, median 99.59%, min 91.30%, max 99.81%, range 8.51 pp). The values are contextual literature figures, not a controlled head-to-head benchmark.

4.2 Research Gap

- Evaluation protocols are not standardized across QR-generation methods, image quality, class balance, and URL sources.
- Domain-level and URL-level leakage can inflate test performance.
- Temporal generalization is underemphasized even though malicious infrastructure changes quickly.
- Operational measures such as latency, privacy, false-positive cost, and mobile constraints are often underreported.
- Users need understandable reasons for warnings rather than only a binary malicious/benign label.
- Robustness to blur, glare, rotation, occlusion, low resolution, screenshots, printing, and adversarial modification requires systematic study.

5. RESEARCH QUESTIONS

RQ1: Does combining QR-image features with decoded-URL features improve suspicious-code detection compared with URL-only and image-only baselines?

RQ2: How do precision, recall, F1-score, false-positive rate, false-negative rate, and inference time change across model families?

RQ3: How well does the proposed approach generalize to domains and URLs not represented in training?

RQ4: Can the system provide useful and understandable warning reasons without opening the destination?

RQ5: How robust is the detector under realistic image degradation and changing URL patterns?

6. HYPOTHESES

H1: The hybrid image-plus-URL model achieves a higher F1-score than the URL-only baseline on a domain-separated test set.

H_{01}: There is no statistically significant difference in F1-score between the hybrid model and the URL-only baseline.

H2: The hybrid model has a lower false-negative rate than an image-only model on a test set containing unseen domains and varied image quality.

H_{02}: There is no statistically significant difference in false-negative rate between the hybrid model and the image-only model.

These hypotheses can be tested only after models are trained and evaluated on the same independent examples. No synthetic p-value or invented confusion matrix is treated as evidence.

7. PROPOSED SYSTEM ARCHITECTURE

The architecture, shown in Fig. 2, begins with an image or camera frame. The QR decoder extracts the payload as data, not as an instruction to navigate. The payload is normalized so that equivalent representations can be compared consistently. A feature-extraction stage then produces URL and image signals. These signals are passed to a risk engine that can be implemented as a traditional machine-learning classifier, a neural model, or a calibrated fusion of multiple models. The final layer generates a risk category and an explanation for the user.

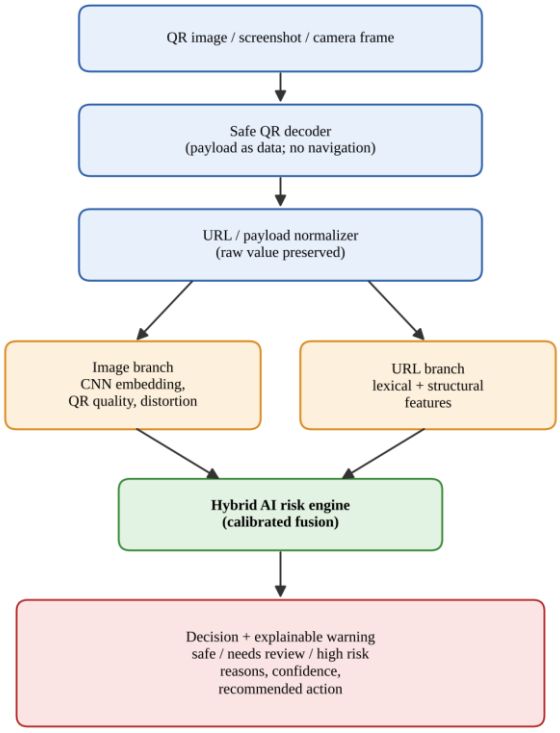


Fig -2: Proposed safe hybrid architecture for suspicious QR-code detection. The user decides; the destination is never opened automatically.

7.1 QR Image Processing

Image preprocessing should improve decoding reliability without destroying security-relevant information. Typical operations include resizing, grayscale conversion, contrast normalization, denoising, and geometric correction. The pipeline should also record whether the original code was captured from a camera, screenshot, printed material, or downloaded image because these conditions can influence the error profile. Rather than assuming that a visually unusual QR code is malicious, image features should be treated as supporting evidence.

7.2 Safe Decoding and Payload Normalization

A decoder should extract the encoded payload and return it as a string or structured object. The application should not automatically open the decoded URL. Normalization may include lower-casing the hostname, separating scheme and host, decoding safe representations for analysis, counting parameters, and canonicalizing equivalent URL forms. Any normalization step must preserve the raw value so that investigators can reproduce the decision and so that potentially dangerous transformations do not hide suspicious content.

7.3 Hybrid Decision Layer

A hybrid model can combine the outputs of an image branch and a URL branch. Early fusion concatenates learned features before classification, while late fusion combines probability or risk scores from separate models. A simpler and more explainable implementation can concatenate engineered URL features with a compact image embedding and feed the result into logistic regression, random forest, gradient boosting, or a small multilayer perceptron. Calibration is important because a raw classifier score should not automatically be interpreted as a probability of maliciousness.

7.4 Implementation Pipeline

The detector can be built as an auditable nine-step pipeline for a mobile or web prototype: (1) Capture the image from camera, gallery, document, or screenshot; (2) Decode locally and return the payload as text; (3) Isolate URL-like payloads from plain text, contacts, or payment strings; (4) Normalize scheme, host, port, path, query, and fragment while keeping the raw value; (5) Extract lexical, structural, and image features without fetching the destination; (6) Score with the baseline or hybrid model to obtain a calibrated risk score; (7) Explain the key signals in plain language; (8) Decide low risk, needs review, or high risk, never navigating automatically; and (9) Log only the minimum information needed for evaluation and security monitoring.

8. RESEARCH METHODOLOGY

8.1 Research Design

A quantitative experimental design is proposed. Three model families are evaluated under the same data partition: Model A, a URL-feature machine-learning baseline; Model B, an image-based CNN; and Model C, the proposed hybrid model. The comparison should use the same independent test examples and should select thresholds only from validation data. The final test set must remain untouched until model selection is complete.

8.2 Dataset and Sampling

Public datasets can provide initial examples of benign and malicious QR codes. The 1,000-image Mendeley dataset [2] is one possible resource, while larger phishing/quishing benchmarks can support broader experiments. The final study should document dataset version, download date, source, licence, class counts, URL provenance, and image-generation process.

Sampling should avoid repeated URLs and should group records by domain before splitting. If multiple QR images encode the same URL, they should not be distributed across training and test sets. A practical design maintains separate training, validation, and test partitions:

the training set fits parameters, the validation set is used for hyperparameter and threshold selection, and the final test set is reserved for one-time evaluation. A temporal test split can additionally place newer URLs in the test set to approximate real-world deployment.

8.3 Model Training

For the URL baseline, logistic regression and random forest are suitable starting points because they are fast and relatively interpretable. The image branch can use a transfer-learning CNN such as MobileNetV3 or InceptionV3, with the final layer adapted to the classification task. The hybrid branch can combine normalized URL features with an image embedding. Hyperparameters should be selected using validation data, and the random seed and software versions should be recorded for reproducibility.

9. EVALUATION METRICS

Accuracy is useful when class proportions are balanced, but it can hide important security failures. Precision measures how many flagged samples are actually malicious, while recall measures how many malicious samples are detected. F1-score balances precision and recall. False-positive rate measures the proportion of benign samples incorrectly flagged, whereas false-negative rate measures the proportion of malicious samples missed by the detector. Latency measures the time needed to produce a decision and is important for mobile or interactive scanning. The metrics are summarized in Table 2.

Table -2: Evaluation Metrics for the Proposed Detector

Metric	Formula	Security relevance
Accuracy	(TP+TN) / (TP+TN+FP+FN)	Overall correctness; can mislead under class imbalance.
Precision	TP / (TP+FP)	Controls unnecessary warnings and user fatigue.
Recall	TP / (TP+FN)	Ability to catch malicious QR codes.
F1-score	2PR / (P+R)	Balances precision and recall.
FPR	FP / (FP+TN)	Benign codes incorrectly flagged.
FNR	FN / (FN+TP)	Critical: missed malicious codes may harm users.
Latency	Input-to-decision time	Important for real-time scanning.

9.1 Statistical Testing

If the same test samples are scored by two classifiers, paired statistical procedures are preferable to treating the two result sets as independent. Bootstrap confidence intervals can be used for differences in F1-score, while paired tests such as McNemar's test can be used for paired classification outcomes under suitable assumptions. A statistically significant result should also be interpreted in terms of effect size and practical significance. A small numerical improvement may not justify higher computational cost or reduced interpretability.

10. RESULTS AND DISCUSSION FRAMEWORK

This paper defines the experimental framework rather than claiming newly trained-model results. The result structure in Table 3 should be populated after the actual experiment is completed. All three models must use the same test set, and the threshold-selection process must be identical in principle. The table is intentionally left without invented values.

Table -3: Recommended Results Table for the Completed Experiment (To Be Filled With Measured Values)

Model	Acc.	Prec.	Rec.	F1	FPR	Latency
URL-only baseline	---	---	---	---	---	---
Image-only CNN	---	---	---	---	---	---
Hybrid model	---	---	---	---	---	---

10.1 Error Analysis

Failures should be categorized rather than reported only as aggregate scores. False negatives may arise from newly registered domains, compromised legitimate sites, shorteners, redirect chains, or poor image quality; false positives may occur when legitimate services use long URLs, payment parameters, or unusual subdomains. Excessive false positives cause warning fatigue, so the operating threshold should follow a documented cost policy rather than maximizing accuracy alone.

10.2 Robustness Testing

A realistic detector should be evaluated under controlled transformations such as rotation, blur, brightness changes, perspective distortion, compression, partial occlusion, glare, and screenshot capture. The goal is not to make the model tolerant of every possible manipulation, but to quantify performance degradation and identify conditions where the system should abstain or request a clearer image. Robustness testing should be

separated from ordinary test evaluation so that the benchmark remains interpretable.

11. SAFETY, PRIVACY, AND DEPLOYMENT

The most important safety property is that the scanner must not automatically navigate to a decoded URL. The decoded destination should be treated as untrusted data until risk assessment is complete. If external services are queried for reputation information, the system should minimize the information sent, document retention policies, and consider whether the URL itself may contain sensitive tokens. For mobile deployment, an on-device URL classifier can reduce data sharing and latency, while heavier reputation checks can be optional and policy-controlled.

Warnings should be understandable. Instead of displaying only "Malicious QR," a useful interface can show the decoded domain, risk category, and a short list of reasons such as "IP address used as hostname," "unusually long URL," or "suspicious domain structure." The system should distinguish between confirmed malicious evidence and uncertainty. An "unknown" or "needs review" category is preferable to pretending that every unfamiliar URL is malicious.

11.1 Applications and Maintenance

Practical deployments include mobile scanners with pre-open warnings, browser extensions that inspect QR images on webpages, payment applications that add a risk check before confirmation, and enterprise gateways that analyze QR images in emails and documents. Because attacker campaigns drift over time, the detector should be maintained as a security component: monitor class and feature distributions and decoder failure rates, retrain with version control, regression tests, and rollback, and publish a model card (data sources, threshold, intended use, limitations, latency). In financial or authentication workflows the AI score should remain one layer of a defence-in-depth design alongside reputation services, browser protections, human review, and user education.

12. LIMITATIONS

Public datasets may not represent current threat campaigns, languages, regions, or attacker infrastructure. Labels may be imperfect, and a benign website can later become compromised. QR image appearance is also influenced by the rendering process, meaning an image model may accidentally learn generator artifacts rather than security-relevant patterns. External reputation APIs may have rate limits, coverage gaps, or privacy implications. Very high accuracy on a static benchmark does not guarantee resilience to unseen domains or adversarial manipulation. The present paper does not report a newly trained model; its literature values are reported from cited studies and its result table is intentionally unfilled.

13. FUTURE SCOPE

- Evaluate multimodal transformer architectures and calibrated score fusion.
- Use temporal evaluation where training data precedes test data to simulate emerging attacks.
- Add explainability through URL-feature contributions and visual saliency checks.
- Study adversarial robustness and image degradation under practical scanning conditions.
- Deploy lightweight on-device models to reduce latency and raw-data sharing.
- Explore privacy-preserving or federated learning where organizations cannot centralize scan data.
- Create a continuously refreshed benchmark with domain-level deduplication and verified labels.
- Investigate multilingual warning messages and accessibility features for non-technical users.

14. CONCLUSION

Suspicious QR-code detection is an important cybersecurity problem because QR codes hide their destination until decoding and can be abused for phishing, malware delivery, payment fraud, and social engineering. Recent literature demonstrates that CNNs, engineered URL features, sequence models, ensemble learning, and adaptive approaches can achieve promising results under their respective experimental settings. However, reported accuracy values from separate studies are not directly comparable because datasets, splits, labels, QR-generation procedures, and test protocols differ.

This paper proposes a safe hybrid workflow that analyzes QR images and decoded URLs before a link is opened. The framework combines interpretable URL indicators with image-level evidence and defines measurable research questions, hypotheses, metrics, and statistical procedures. The central research requirement is leakage-resistant evaluation on independently partitioned data, including unseen domains and realistic image conditions. The final system should prioritize low false-negative rates while controlling false positives, provide understandable warning reasons, preserve privacy, and maintain reproducibility. The literature chart in this paper is contextual evidence rather than experimental proof, and the empty results table must be replaced with measured outputs before making empirical claims.

REFERENCES

[1] "Recognition of Valid QR Codes with Machine Learning," in Proc. IEEE 13th Int. Conf. Communication Systems and Network Technologies

(CSNT), 2024, doi: 10.1109/CSNT60213.2024.10546171.

[2] S. Galadima, "Dataset of 1000 Images of Malicious and Benign QR Codes 2025," Mendeley Data, Version 1, Feb. 20, 2025, doi: 10.17632/cmhh7744sp.1.

[3] A. A. Alsulami et al., "Efficient Malicious QR Code Detection System Using an Advanced Deep Learning Approach," Computer Modeling in Engineering & Sciences, vol. 145, no. 1, pp. 1117–1140, 2025, doi: 10.32604/cmes.2025.070745.

[4] "HybridStack-MLP: Advanced Ensemble Learning for Malicious QR Code and URL Detection," in Proc. IEEE 2nd Int. Conf. for Women in Computing (InCoWoCo), 2025, doi: 10.1109/InCoWoCo68239.2025.11407094.

[5] N. N. Maree, "A Classification Model to Detect Malicious Software in QR Code," NTU Journal of Engineering and Technology, vol. 5, no. 1, 2026, doi: 10.56286/ngcs4e85.

[6] H. Almousa and S. A. Alsuhibany, "QR code security: an adaptive retraining approach for dynamic URL-based threat detection," Scientific Reports, vol. 16, Art. no. 13143, 2026, doi: 10.1038/s41598-026-43002-z.

[7] D. Njuguna and J. Ndia, "Quick Response Code Security Attacks and Countermeasures: A Systematic Literature Review," Journal of Cyber Security, vol. 7, pp. 1–20, 2025, doi: 10.32604/jcs.2025.059398.

[8] Canadian Institute for Cybersecurity, "CIC-Trap4Phish 2025 Dataset: A Unified Multi-Format Dataset for Phishing and Quishing Attachment Detection," University of New Brunswick.

[9] Y. M. Tashtoush, M. Khashim, and N. H. Alsaedi, "Comparing Machine Learning and Deep Learning Models for QR Code Phishing Detection," in Proc. Int. Conf. Cybersecurity and AI-Based Systems (Cyber-AI), 2025, pp. 291–297, doi: 10.1109/cyberai66431.2025.11233820.