

DEEP LEARNING-BASED PHISHING DETECTION FOR CYBER THREAT INTELLIGENCE: DEVELOPMENT AND EVALUATION OF THE PHISHING GUARD SYSTEM

¹Dr Lavanya Srinivasan, ²Vishwajeet Redhu

²MSc Cyber Security, School of Computing and Engineering, University of West London, United Kingdom, 34125608@student.uwl.ac.uk,
Lavanya.Srinivasan@uwl.ac.uk

Corresponding Author:

34125608@student.uwl.ac.uk

To Cite This Article: DEEP LEARNING-BASED PHISHING DETECTION FOR CYBER THREAT INTELLIGENCE: DEVELOPMENT AND EVALUATION OF THE PHISHING GUARD SYSTEM. (2026). Journal of Advance Research in Computer Science & Engineering (ISSN 2456-3552), 11(1), 33-45. <https://doi.org/10.61841/eja5n288>

ABSTRACT

Background

Phishing is a common cybersecurity attack in which criminals use fake or misleading websites to trick people into sharing sensitive information. This may include usernames, passwords, banking details or personal information. Traditional detection methods such as blacklists and fixed security rules can identify many known threats, but they may be less effective when a phishing website is new or its URL has been changed.

Objectives

This research aimed to develop and evaluate PhishingGuard, a web-based phishing detection and website security analysis system. The study investigated the performance of a deep-learning model for phishing URL classification and then examined how the trained model could be integrated into a practical security application.

Methods

The CNN was developed using the PhiUSIIL Phishing URL dataset. The original dataset contained 235,795 records. After removing 425 duplicate URLs, 235,370 unique URLs remained. A stratified 80:20 split produced 188,296 URLs for training and 47,074 URLs for held-out testing. A character-level Convolutional Neural Network (CNN) was trained to learn patterns directly from URL text. The frozen CNN was then integrated into PhishingGuard together with URL-based lexical analysis and live website security analysis.

Results

The frozen CNN achieved 99.81% accuracy, 99.99% precision, 99.58% recall, 99.78% F1-score and 0.9986 ROC-AUC on the held-out test set. The confusion matrix contained 26,967 true negatives, 3 false positives, 85 false negatives and 20,019 true positives.

An additional balanced evaluation of 10,000 URLs produced 99.85% accuracy. However, this sample came from the same underlying PhiUSIIL source and therefore does not represent independent external validation.

Practical testing also identified an important limitation. Small changes in the way the same legitimate URL was written could cause large changes in the CNN prediction. PhishingGuard therefore introduced validated URL processing and application-level decision logic while keeping the evaluated CNN unchanged.

Conclusion

The findings show that character-level deep learning can provide strong phishing URL classification under the conditions tested in this research. However, practical testing also showed that high model accuracy alone is not enough for a complete phishing detection system. PhishingGuard therefore combines the CNN prediction with URL characteristics and available live website information to provide a broader and easier-to-understand risk assessment.

KEYWORDS: Phishing Detection; Deep Learning; Convolutional Neural Network; Cyber Threat Intelligence; URL Analysis; Website Security; Phishing Guard

1. INTRODUCTION

Phishing is a cyberattack in which an attacker tries to make a person trust a fake message, link or website. A phishing website may look similar to a genuine banking, shopping, email or social-media website. If the user believes that the website is genuine, they may enter passwords, banking information or other sensitive data. Phishing remains an important cybersecurity problem because attackers continue to change both the technical and social methods they use to deceive users (Alkhalil *et al.*, 2021; Safi and Singh, 2023).

One traditional phishing detection method is a blacklist. A blacklist contains URLs or domains that have already been identified as malicious. It can therefore provide a warning when a user tries to access a known phishing website. However, a newly created phishing website may not yet be present in the blacklist.

Another approach is heuristic or rule-based detection. This method looks for characteristics that may indicate a suspicious URL. Examples include unusual URL length, several subdomains, the use of an IP address, suspicious words or unusual symbols. These indicators can be useful, but a single suspicious feature cannot prove that a website is phishing. Legitimate websites can also contain complex URLs (Melo e Silva, Feitosa and Garcia, 2020; Safi and Singh, 2023).

Machine learning provides another approach. Instead of depending only on manually created rules, a machine-learning model learns patterns from examples. During training, it is given examples of legitimate and phishing URLs. It then uses patterns learned from these examples to classify other URLs. Previous research has demonstrated the value of machine-learning methods for URL-based phishing and malicious URL detection (Sahingoz *et al.*, 2019; Reyes-Dorta, Caballero-Gil and Rosa-Remedios, 2024).

Deep learning extends this idea by allowing a model to learn more complex patterns directly from data. This is useful for URL analysis because a web address contains a sequence of letters, numbers and symbols. A deep-learning model can learn from these character sequences instead of requiring every suspicious feature to be manually defined.

Le *et al.* (2018) investigated deep learning for malicious URL detection, while Aljofey *et al.* (2020) investigated a character-level CNN for phishing detection. A systematic review by Catal *et al.* (2022) also showed the growing use of deep learning in phishing detection research.

However, high classification accuracy does not automatically mean that a model will behave reliably in every practical situation. A CNN trained mainly on URL text sees patterns in the characters supplied to it. It does not automatically understand all the security characteristics of the website behind that URL. For example, it does not independently determine everything about the website's certificate, page forms, redirects or other live behaviour.

Another important issue is generalisation. In simple terms, this means whether a model that works well on one dataset will also work well on different or newer data. This is particularly important in phishing detection because phishing URLs continue to change. Rashid *et al.* (2024) identified generalisation as an important problem in phishing URL detection research.

This study addresses these issues through the development of PhishingGuard, a web-based phishing detection and website security analysis system. Instead of depending only on a CNN prediction, the system combines three types of analysis:

CNN Analysis + URL/Lexical Analysis + Live Website Security Analysis → Overall Risk Assessment

CNN learns patterns from URL text. Lexical analysis examines understandable characteristics of the URL. Live analysis collects available security information from the website itself. These results are combined to provide a risk assessment and security information that can be understood more easily by the user.

The CNN was developed using the PhiUSIIL Phishing URL dataset (Prasad and Chandra, 2024). After cleaning, 235,370 unique URLs remained. The frozen CNN achieved 99.81% accuracy on a held-out test set of 47,074 URLs. However, practical testing later showed that different textual forms of the same legitimate URL could produce very different CNN predictions.

The main contribution of this study is therefore not simply a high CNN accuracy figure. The research develops a working prototype that connects deep-learning URL classification with lexical indicators, live website information, safe URL processing and user-facing security information. It also examines a practical weakness that became visible after the CNN was moved into the application.

The study focuses on two connected questions: **how effectively can a character-level CNN classify phishing URLs under the evaluated conditions, and what practical problems appear when the trained model is integrated into a web-based phishing analysis system?**

2. MATERIALS AND METHODS

2.1 STUDY DESIGN

The research followed a practical experimental approach. The work was completed in two connected stages. First, a deep-learning model was developed and evaluated for phishing URL classification. Second, the selected model was integrated into PhishingGuard.

The overall development process was:

Dataset → Data Cleaning → Character Vectorisation → CNN Training → Model Evaluation → Frozen CNN → PhishingGuard Integration → Functional Testing → Deployment

This separation allowed the CNN to remain unchanged while the surrounding application was developed and improved.

2.2 DATASET AND DATA PREPARATION

The CNN was developed using the PhiUSIIL Phishing URL dataset, obtained from the UCI Machine Learning Repository (Prasad and Chandra, 2024). The original dataset contained 235,795 records and 54 columns. The URL and corresponding classification label were the main information required for the character-level CNN.

During data preparation, 425 duplicate URL records were removed. No conflicting duplicate labels were identified. This left 235,370 unique URLs, consisting of 134,850 legitimate URLs and 100,520 phishing URLs.

Table 1: Distribution of the cleaned dataset

Class	Number of URLs
Legitimate	134,850
Phishing	100,520
Total	235,370

The original dataset used label **1** for legitimate URLs and **0** for phishing URLs. For this research, the labels were remapped so that **0** represented legitimate and **1** represented phishing. This made the CNN output easier to interpret as a phishing probability.

2.3 TRAINING AND TEST SPLIT

The cleaned dataset was divided using a stratified 80:20 train/test split with: `random_state = 42`.

Stratification was used to maintain a similar class distribution in the training and test sets.

Table 2: Training and held-out test distribution

Dataset	Legitimate	Phishing	Total
Training	107,880	80,416	188,296
Held-out test	26,970	20,104	47,074
Total	134,850	100,520	235,370

The 47,074 test URLs were kept separate from model training and were later used to evaluate the frozen CNN.

2.4 CHARACTER-LEVEL URL REPRESENTATION

CNN analysed URLs at character level. This means that the model looked at the letters, numbers and symbols that make up each web address rather than treating the complete URL as one word.

A TensorFlow `TextVectorization` layer converts the characters into numerical sequences.

The vectorisation process used a maximum setting of 256 tokens, a fixed sequence length of 200 characters, and produced an actual learned vocabulary of 73 character tokens.

The process can be represented as:

Raw URL → Characters → Numerical Sequence → CNN

Character-level URL learning has also been investigated in earlier malicious and phishing URL research (Le *et al.*, 2018; Aljofey *et al.*, 2020).

2.5 CNN ARCHITECTURE

The model was developed using TensorFlow/Keras. After vectorisation, an embedding layer produced a 32-dimensional representation. Two one-dimensional convolutional layers were then used to learn useful patterns from the URL characters.

The architecture was:

URL Input → TextVectorization → Embedding → Conv1D → MaxPooling1D → Conv1D → GlobalMaxPooling1D → Dense → Dropout → Dense → Dropout → Phishing Probability

Both convolutional layers used 64 filters. The dense layers contained 64 and 32 units. The final model contained 31,265 trainable parameters.

For a non-technical reader, the model can be understood simply: the URL is first changed into numbers, the CNN looks for useful patterns in those numbers, and the final layers use those learned patterns to produce a phishing probability.

model.summary()

... PHISHINGGUARD – FINAL CNN ARCHITECTURE

Model: "functional"

Layer (type)	Output Shape	Param #
url_input (InputLayer)	(None)	0
text_vectorization (TextVectorization)	(None, 200)	0
embedding (Embedding)	(None, 200, 32)	2,336
conv1d (Conv1D)	(None, 196, 64)	10,304
max_pooling1d (MaxPooling1D)	(None, 98, 64)	0
conv1d_1 (Conv1D)	(None, 96, 64)	12,352
global_max_pooling1d (GlobalMaxPooling1D)	(None, 64)	0
dense (Dense)	(None, 64)	4,160
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 32)	2,080
dropout_1 (Dropout)	(None, 32)	0
phishing_probability (Dense)	(None, 1)	33

Total params: 93,797 (366.40 KB)
 Trainable params: 31,265 (122.13 KB)
 Non-trainable params: 0 (0.00 B)

Figure 1. Structure of the character-level CNN used for phishing URL classification.

2.6 MODEL TRAINING

The model was trained in Google Colab. Training used a batch size of 128, a maximum of 10 epochs, and a 20% validation split. Early stopping was used so that training could stop when validation performance was no longer improving.

Training finished after six epochs. The strongest validation loss occurred around epoch 4 and was approximately 0.0113. The selected model was saved as:

`phishing_url_detector.keras` and was then treated as the frozen CNN used for the final evaluation and application.

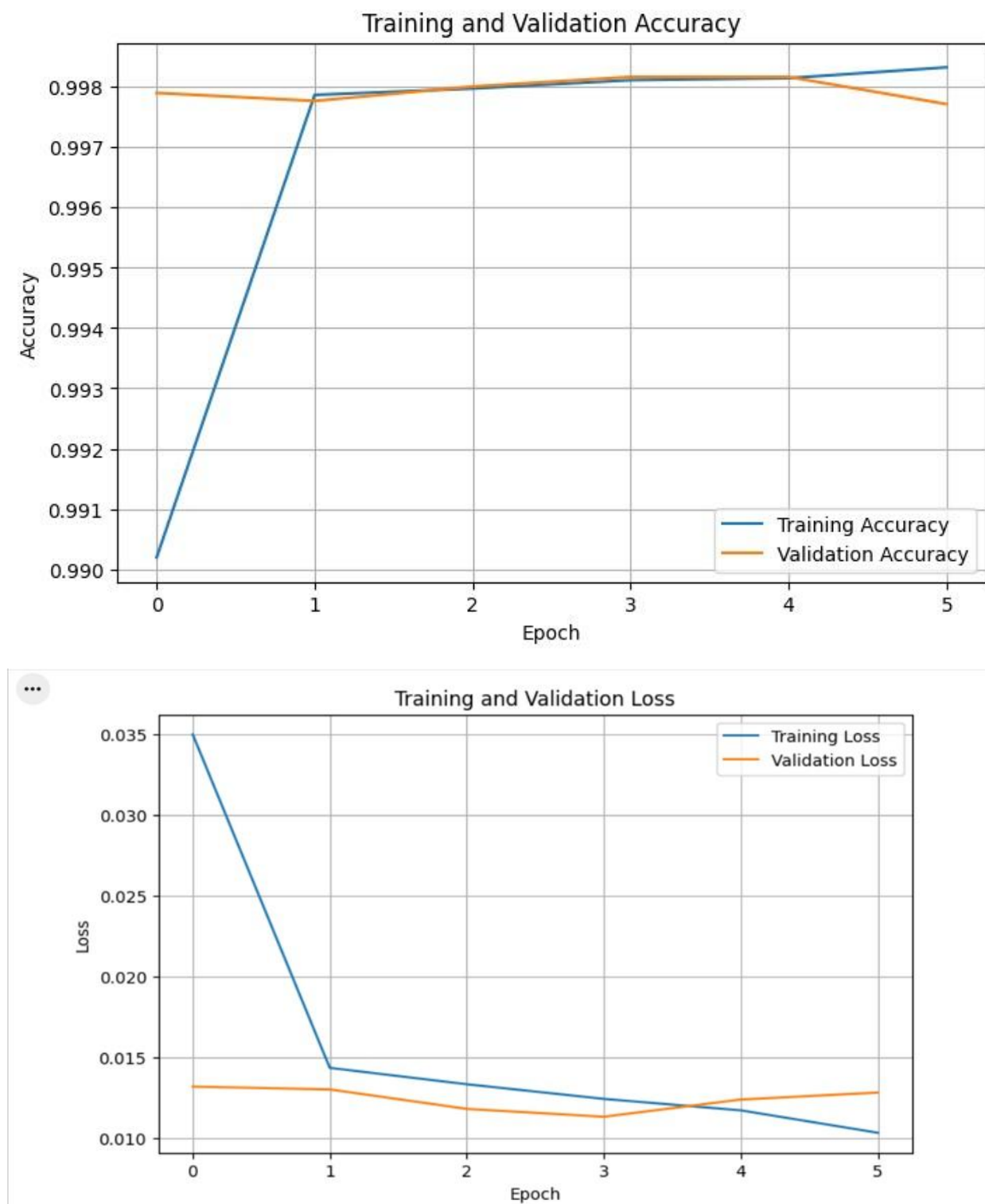


Figure 2. Training and validation behaviour of the character-level CNN.

2.7 PHISHINGGUARD ARCHITECTURE

The frozen CNN was integrated into PhishingGuard. The complete system uses three analysis layers:

1. **CNN analysis** – learns patterns from URL characters.
2. **Lexical analysis** – examines understandable characteristics of the URL.
3. **Live website analysis** – examines available information from the website.

The complete workflow is:

User Enters URL → URL Validation → Safe Processing → CNN + Lexical + Live Analysis → Combined Risk Assessment → Security Report

The final implemented system follows this general process from user input through validation, analysis, combined decision logic and a user-facing result.

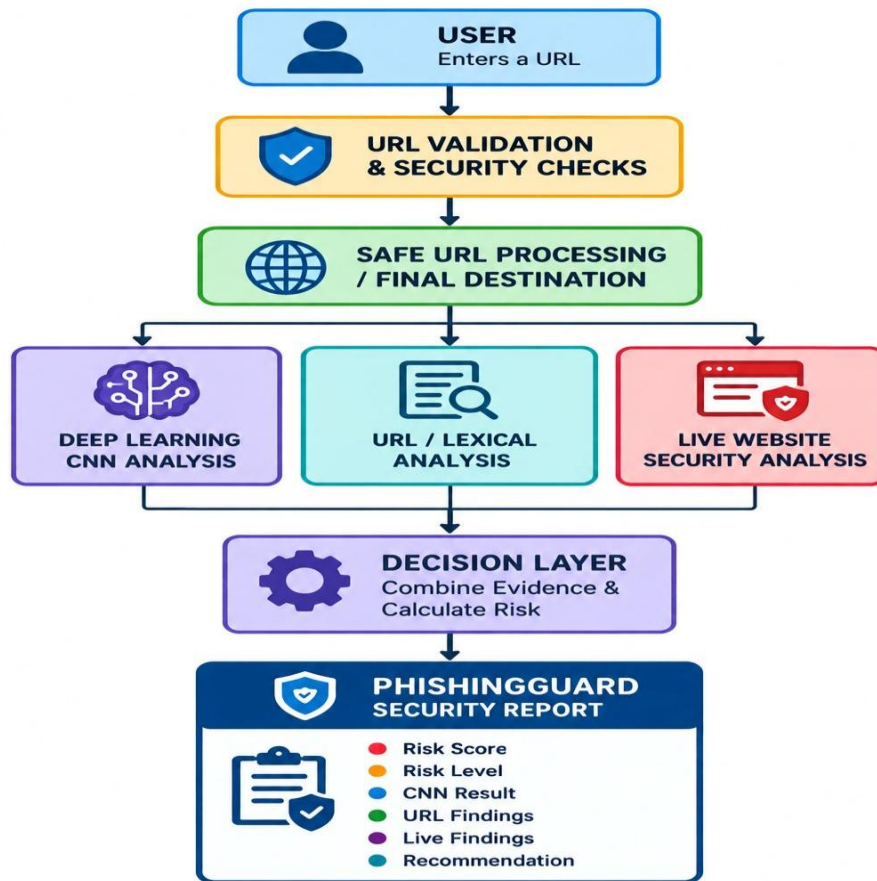


Figure 3: Overall architecture and processing flow of the PhishingGuard system.

2.8 LEXICAL AND LIVE WEBSITE ANALYSIS

The lexical layer examines features such as raw IP addresses, punycode, the @ symbol, shortening services, URL length, subdomains, hyphens, digits, encoded content, suspicious terms, entropy and path depth.

These features are treated as indicators rather than proof of phishing. For example, a long URL can appear suspicious, but a legitimate website may also use a long URL. This is one reason why the system does not depend on a single lexical feature. Heuristic URL-based methods and their limitations have previously been discussed by Melo e Silva, Feitosa and Garcia (2020).

The live-analysis layer examines available information including HTTPS/TLS, redirects, forms, password or email fields, hidden fields, iframes, external or insecure form actions, links and suspicious phrases.

HTTPS or a valid certificate is not treated as proof that a website is safe, because phishing websites can also use encrypted connections.

2.9 SAFE URL PROCESSING

Because users can submit URLs to the application, PhishingGuard applies security checks before attempting live analysis. The implementation blocks unsupported schemes, embedded credentials, local/internal addresses, private or reserved network addresses and unsupported ports. Redirect, timeout and response-size controls are also used.

These controls help prevent the application itself from making unsafe requests.

2.10 COMBINED RISK ASSESSMENT

The prototype combines the three analysis layers using:

$$R=(0.62\times C)+(0.20\times L)+(0.18\times W)$$

where C is the CNN score, LL is lexical risk and WW is live website risk.

The overall score is interpreted as:

Score	Risk level
Below 35	LOW
35 to below 60	MEDIUM
60 to below 80	HIGH
80–100	CRITICAL

The 62%, 20% and 18% weights are prototype design choices. They were not statistically optimised using a large independent validation dataset.

2.11 EVALUATION

Evaluation was carried out at different levels. First, the frozen CNN was tested on the 47,074-URL held-out set using accuracy, precision, recall, F1-score, confusion matrix and ROC-AUC.

Second, an additional balanced 10,000-URL evaluation was conducted. Because these URLs came from the same underlying PhiUSIIL source, this test is treated as same-source evaluation, not independent external validation.

Finally, the complete PhishingGuard application underwent functional and regression testing. Practical testing also examined URL representation after different forms of the same legitimate website were found to produce different CNN predictions.

3. RESULTS

3.1 HELD-OUT CNN EVALUATION

The frozen CNN achieved strong results on the 47,074-URL held-out test set.

Table 3: CNN performance on the held-out test set

Metric	Result
Accuracy	99.81%
Precision	99.99%
Recall	99.58%
F1-score	99.78%
ROC-AUC	0.9986

These results show strong separation between legitimate and phishing URLs within the evaluated dataset. However, they represent the performance of the CNN, not the measured accuracy of the complete PhishingGuard application.

3.2 CONFUSION MATRIX

The confusion matrix contained:

Classification outcome	Number
True Negative	26,967
False Positive	3
False Negative	85
True Positive	20,019
Total	47,074

The model correctly classified 26,967 legitimate URLs and 20,019 phishing URLs. Only three legitimate URLs were incorrectly classified as phishing. However, 85 phishing URLs were incorrectly classified as legitimate.

The false negatives remain important because they represent phishing URLs that were missed by CNN. Therefore, even very high accuracy does not mean that the model is perfect.

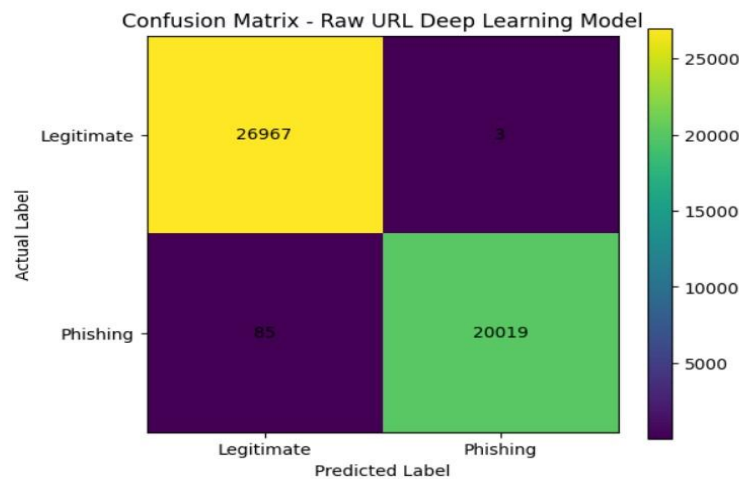


Figure 4: Confusion matrix for the frozen CNN on the 47,074-URL held-out test set.

3.3 ROC-AUC

CNN achieved a ROC-AUC of 0.9986. A value close to 1 indicates strong separation between the two classes across different classification thresholds. This result supports the strong performance shown by the other evaluation measures.

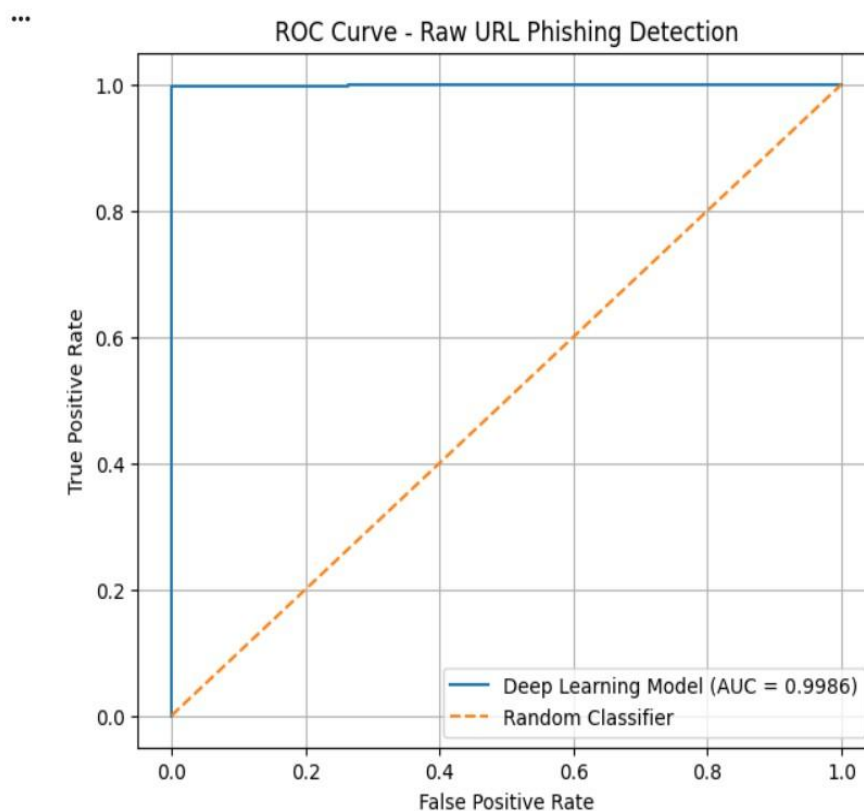


Figure 5: ROC curve of the frozen CNN on the held-out test set.

3.4 ADDITIONAL BALANCED EVALUATION

The frozen CNN was also evaluated using a balanced sample of 10,000 URLs, consisting of 5,000 legitimate and 5,000 phishing URLs.

The CNN achieved 99.85% accuracy, 100% precision, 99.70% recall and 99.85% F1-score. The confusion matrix contained 5,000 true negatives, no false positives, 15 false negatives and 4,985 true positives.

These results provide additional evidence of strong CNN performance. However, the sample came from the same underlying PhiUSIIL source. It therefore does not provide independent external validation. The source paper explicitly retains this limitation rather than presenting the experiment as evidence of external generalisation.

3.5 URL REPRESENTATION SENSITIVITY

One of the most important findings appeared during practical testing.

The frozen CNN produced very different predictions for several textual forms associated with the same legitimate Google website:

URL representation	CNN phishing probability
google	0.33%
google/	100%
https://google.com	100%
google.com	20.02%

For example, adding a slash to **google** changed the CNN prediction from 0.33% to 100%.

This finding showed that the character-level CNN was sensitive to the exact form of the URL it received. The problem was important because users may enter the same website in different ways.

The final backend therefore validates the submitted URL, follows the legitimate destination where appropriate and prepares a more consistent URL representation for CNN analysis. Importantly, this processing did not retrain or alter the frozen CNN.

3.6 CROSS-LAYER DECISION LOGIC

Practical testing also identified situations in which the CNN prediction strongly disagreed with lexical and live website evidence.

An application-level corroboration rule was introduced for cases where the CNN score is at least 80, lexical risk is below 10, live risk is below 10, live analysis is available, HTTPS and valid TLS are present, and no major form warning is found.

When these conditions are met, the overall application-level risk can be capped at 24.99 before later safety overrides.

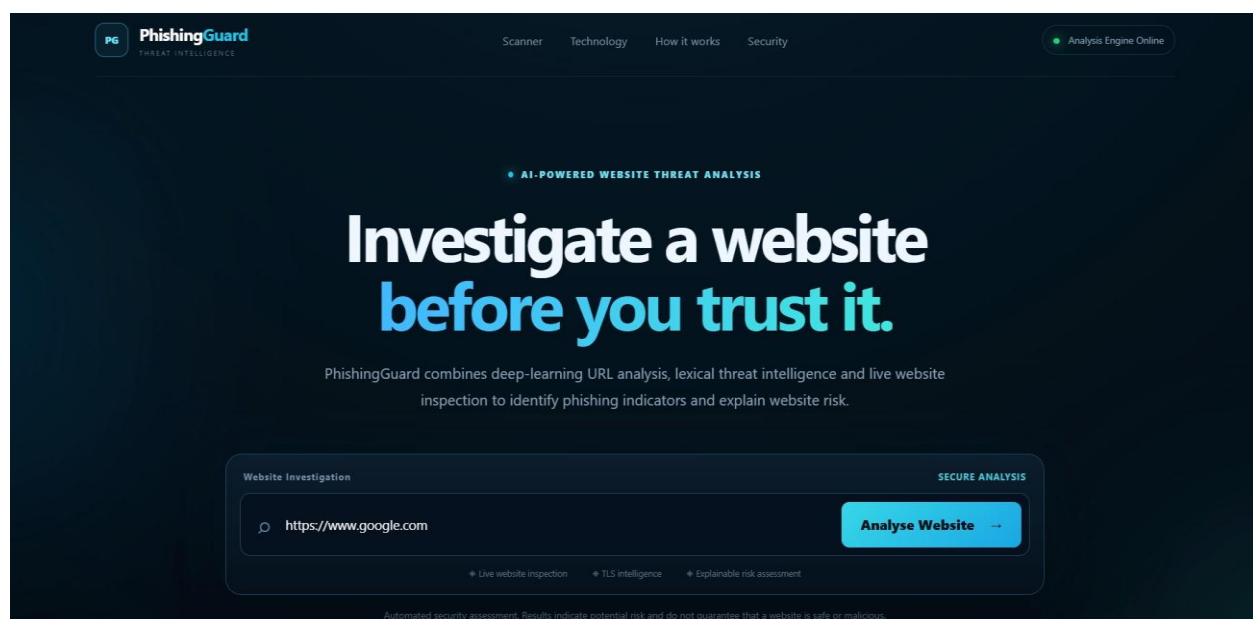
The value 24.99 is not a CNN probability. It is an application-level result created by the decision layer. The CNN itself remains unchanged.

This rule improved some tested cases, but it also creates a possible trade-off. A sophisticated phishing website may use HTTPS and a valid certificate and may appear normal to some security checks. The rule therefore requires wider independent evaluation before it can be considered fully validated. The source paper itself identifies this limitation.

3.7 FINAL PHISHINGGUARD PROTOTYPE

The final system presents more than a single model probability. A user enters a URL and receives an overall risk category together with supporting security information.

The deployed workflow combines validation, CNN analysis, lexical analysis, live analysis, weighted risk calculation, decision logic and a security report. The system was deployed as a web-accessible prototype rather than remaining only as a Google Colab experiment.



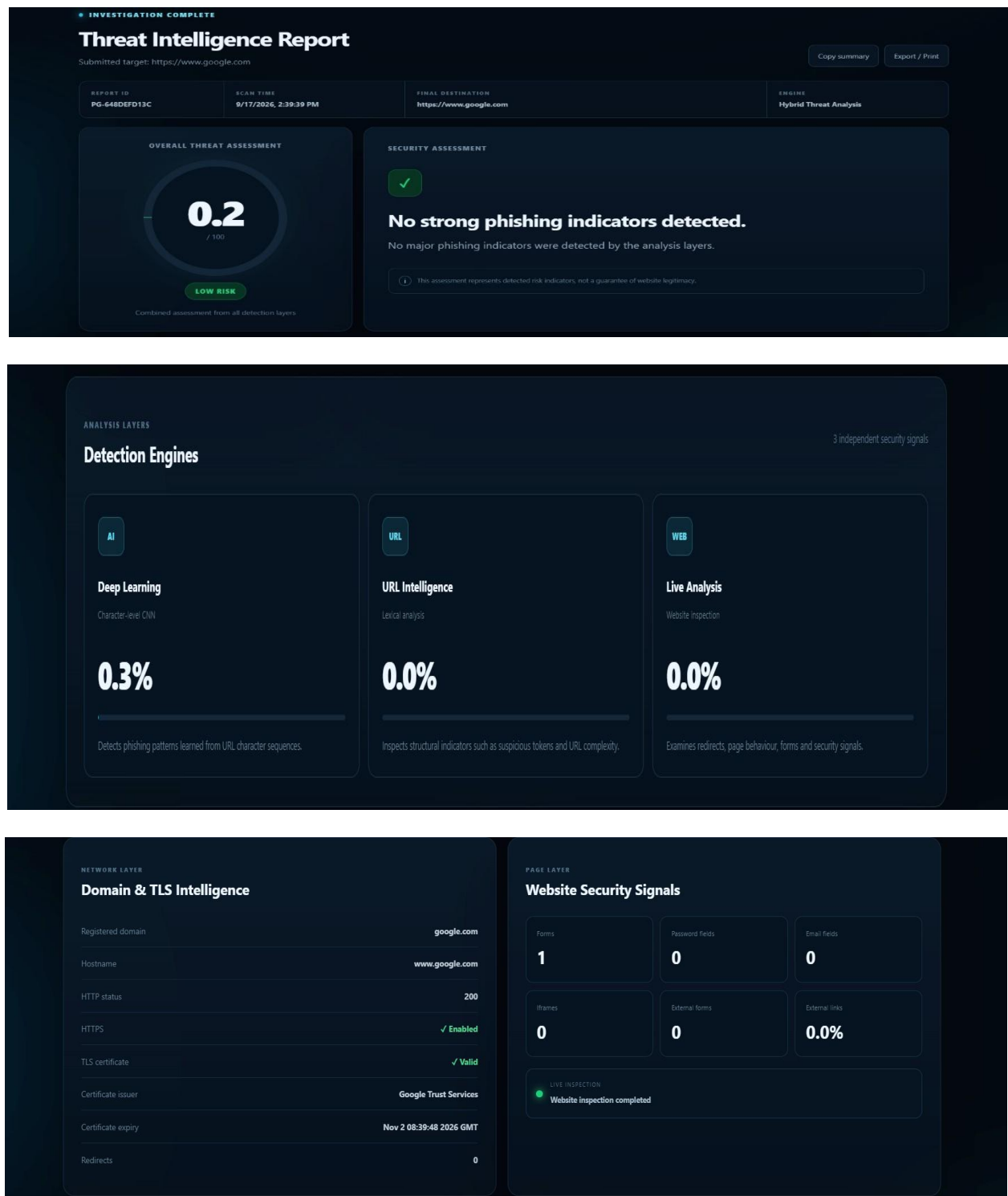


Figure 6: PhishingGuard user interface showing the security analysis and user-facing risk result.

4. DISCUSSION

The results show that the character-level CNN performed strongly on the evaluated PhiUSILL data. However, practical testing also showed that a high numerical score alone is not enough to judge whether a phishing detector will behave reliably in every practical situation.

The CNN achieved 99.81% accuracy and 0.9986 ROC-AUC on the held-out test set. These results support the usefulness of character-level URL analysis. Earlier work by Le et al. (2018) and Aljofey et al. (2020) also demonstrated the value of learning directly from URL information using deep-learning methods.

The very high precision means that few legitimate URLs were incorrectly classified as phishing in the held-out test. However, 85 phishing URLs were missed. In cybersecurity, these false negatives remain important because a missed phishing URL may expose a user to a malicious website.

The additional 10,000-URL evaluation also produced strong results. However, it is important not to overstate this experiment. The URLs came from the same underlying PhiUSIIL source. A model that performs strongly on data from one source may behave differently on newer URLs or data collected from another source. Generalisation therefore remains an important issue (Rashid et al., 2024).

The URL representation experiment is particularly important. A headline accuracy of 99.81% could suggest that the model is almost completely reliable. Practical testing provided a more complicated picture. The same legitimate website produced probabilities ranging from 0.33% to 100% depending on how its address was written.

This does not make the held-out evaluation incorrect. Instead, it shows that dataset evaluation and practical system testing measure different things. The held-out experiment measures classification performance under the defined dataset conditions. Practical testing examines what happens when users interact with the system in less controlled ways.

The application-level URL processing was introduced to reduce this problem for the tested cases. The original CNN was deliberately kept frozen. This is important because changing the model after seeing difficult examples and continuing to report the original performance could make the evaluation unclear.

The practical findings also support the use of multiple analysis layers. A CNN provides a learned prediction from URL text, but it does not explain every characteristic of the website. Lexical analysis provides understandable URL indicators, while live analysis adds information about the website itself.

The three layers therefore provide different forms of evidence. This does not mean that the final 62/20/18 weighting is optimal. Those values remain prototype design choices and require independent evaluation.

The cross-layer rule should be interpreted in the same careful way. It can reduce the overall risk when the CNN strongly disagrees with several low-risk indicators, but HTTPS and a valid TLS certificate cannot prove that a website is legitimate. A phishing site can also have these characteristics. The rule may therefore reduce some false positives while creating a possible false-negative risk.

Previous research has shown that machine learning and deep learning can be useful for phishing and malicious URL detection (Sahingoz et al., 2019; Aljofey et al., 2020; Le et al., 2018). Wider reviews also show continued research interest in these approaches (Catal et al., 2022; Safi and Singh, 2023).

Direct accuracy comparisons between different studies should nevertheless be treated carefully. Different studies may use different datasets, preprocessing methods, class distributions and evaluation procedures. A larger accuracy percentage in one experiment does not automatically prove that one system is better in every situation.

The contribution of this study is therefore not based on claiming that PhishingGuard is better than every existing phishing detector. Its contribution is the development of a working prototype that combines a high-performing character-level CNN with lexical analysis, live website analysis, safe URL processing and user-facing security information. The research also reports a practical model weakness rather than presenting only successful results.

From a Cyber Threat Intelligence (CTI) perspective, this approach is useful because it combines several pieces of evidence about a potentially suspicious URL. Instead of giving the user only a binary label, PhishingGuard can present the CNN result, URL characteristics, available website information and an overall risk category.

PhishingGuard should therefore be viewed as a phishing risk analysis and decision-support prototype, not as a guarantee that a website is safe. A LOW result cannot prove that a website is harmless, and a HIGH result should be treated as evidence requiring caution rather than automatic proof of malicious activity.

The overall research path can be summarised as:

Strong CNN Performance → Practical Testing → URL-Sensitivity Finding → Investigation → Application-Level Mitigation → Further Evaluation

This is an important outcome because it shows both what the developed approach does well and where further work is required.

5. LIMITATIONS AND FUTURE DIRECTIONS

The study has several limitations that need to be clearly reported.

First, the main CNN evaluation and the additional balanced evaluation both used data originating from the PhiUSIIL source. Although the 47,074 test URLs were held out from training, the study does not provide completely independent external validation. The source paper explicitly notes that the results therefore do not demonstrate external generalisation.

Second, the frozen CNN showed URL representation sensitivity. Application-level URL processing reduced this problem for the tested examples, but the underlying CNN was not retrained. The current approach should therefore be described as a mitigation around the model rather than a complete solution to the model's sensitivity.

Third, the 62% CNN, 20% lexical and 18% live-analysis weights were selected for the prototype and were not statistically optimised using a large independent dataset. The same limitation applies to the 24.99 cross-layer risk cap.

Fourth, live website analysis has practical limitations. A phishing website can use HTTPS and possess a valid TLS certificate. Website content can change, websites can block automated requests, and network conditions can prevent some live information from being collected. Live analysis therefore provides supporting evidence rather than permanent proof of website safety.

Finally, the complete PhishingGuard system was functionally tested on a smaller practical set than the CNN dataset evaluation. Therefore, 99.81% is the accuracy of the frozen CNN on the held-out PhiUSIIL test set, not the measured accuracy of the complete deployed PhishingGuard system.

Future work should first evaluate the model using completely independent and more recent datasets. This would provide stronger evidence about whether the CNN generalises to phishing URLs from different sources and time periods.

URL representation should also be addressed during future model development. Different valid representations of the same website could be included during training, together with more consistent URL canonicalisation and data augmentation. A future CNN V2 could then be trained and evaluated separately from the current frozen model.

The hybrid risk weights and decision thresholds should also be evaluated using a large labelled validation dataset. This would allow the contribution of CNN, lexical and live-analysis evidence to be determined using experimental evidence rather than prototype choices.

Finally, the complete PhishingGuard pipeline should be tested on a much larger independent collection of legitimate and phishing websites. A future user study could also examine whether technical and non-technical users understand the risk categories, supporting information and recommendations.

6. CONCLUSION

This research investigated the use of deep learning for phishing URL detection and developed a web-based security prototype called PhishingGuard.

A character-level CNN was developed using the PhiUSIIL dataset. After duplicate removal, 235,370 unique URLs remained. A stratified 80:20 split produced 188,296 training URLs and a held-out test set of 47,074 URLs.

The frozen CNN achieved 99.81% accuracy, 99.99% precision, 99.58% recall, 99.78% F1-score and 0.9986 ROC-AUC on the held-out test set. An additional balanced 10,000-URL same-source evaluation achieved 99.85% accuracy. These results show strong CNN classification performance under the evaluated conditions.

However, practical testing identified an important weakness. Small changes in URL representation could cause large differences in CNN predictions. This demonstrated that high dataset-level accuracy does not automatically guarantee stable behaviour after a model is placed inside a real application.

PhishingGuard therefore combines the frozen CNN with lexical analysis, live website security analysis, safe URL processing and application-level decision logic. The system presents an overall risk assessment together with supporting security information rather than relying only on a model probability.

The research therefore provides two connected outcomes. First, it demonstrates strong character-level CNN performance under the tested conditions. Second, it shows the practical challenges that can appear when a trained phishing classifier is moved from an experimental environment into a working web application.

The study does not claim that PhishingGuard can identify every phishing website. Independent external validation, improved URL handling, larger complete-system testing and empirical optimisation of the hybrid decision process are still required.

Overall, the findings show that deep learning can be a strong component of phishing URL detection, but model accuracy should not be considered in isolation. Practical phishing analysis also requires careful URL processing, additional security evidence, testing of unexpected behaviour and clear communication of the result to the user.

REFERENCES

1. Aljofey, A., Jiang, Q., Qu, Q., Huang, M. and Niyigena, J.-P. (2020) 'An effective phishing detection model based on character level convolutional neural network from URL', *Electronics*, 9(9), Article 1514. doi: 10.3390/electronics9091514.
2. Alkhalil, Z., Hewage, C., Nawaf, L. and Khan, I. (2021) 'Phishing attacks: A recent comprehensive study and a new anatomy', *Frontiers in Computer Science*, 3, Article 563060. doi: 10.3389/fcomp.2021.563060.
3. Catal, C., Giray, G., Tekinerdogan, B., Kumar, S. and Shukla, S. (2022) 'Applications of deep learning for phishing detection: A systematic literature review', *Knowledge and Information Systems*, 64(6), pp. 1457–1500. doi: 10.1007/s10115-022-01672-x.
4. Le, H., Pham, Q., Sahoo, D. and Hoi, S.C.H. (2018) 'URLNet: Learning a URL representation with deep learning for malicious URL detection', *arXiv preprint, arXiv:1802.03162*.
5. Melo e Silva, C.M.R., Feitosa, E.L. and Garcia, V.C. (2020) 'Heuristic-based strategy for phishing prediction: A survey of URL-based approach', *Computers & Security*, 88, Article 101613. doi: 10.1016/j.cose.2019.101613.

6. Prasad, A. and Chandra, S. (2024) PhiUSIIL Phishing URL (Website) [Dataset]. UCI Machine Learning Repository. doi: 10.1016/j.cose.2023.103545.
7. Rashid, F., Doyle, B., Han, S.C. and Seneviratne, S. (2024) 'Phishing URL detection generalisation using unsupervised domain adaptation', Computer Networks, 245, Article 110398. doi: 10.1016/j.comnet.2024.110398.
8. Reyes-Dorta, N., Caballero-Gil, P. and Rosa-Remedios, C. (2024) 'Detection of malicious URLs using machine learning', Wireless Networks, 30, pp. 7543–7560. doi: 10.1007/s11276-024-03700-w.
9. Safi, A. and Singh, S. (2023) 'A systematic literature review on phishing website detection techniques', Journal of King Saud University – Computer and Information Sciences, 35(2), pp. 590–611. doi: 10.1016/j.jksuci.2023.01.004.
11. Sahingoz, O.K., Buber, E., Demir, O. and Diri, B. (2019) 'Machine learning based phishing detection from URLs', Expert Systems with Applications, 117, pp. 345–357. doi: 10.1016/j.eswa.2018.09.029.