Enhancing Phishing Detection in Cloud Environments Using RNN-LSTM in a Deep Learning Framework

Oussama Senouci and Nadjib Benaouda

University of Mohamed El Bachir El Ibrahimi, Bordj Bou Arreridj, Algeria

https://doi.org/10.26636/jtit.2025.1.1916

Abstract — Phishing attacks targeting cloud computing services are more sophisticated and require advanced detection mechanisms to address evolving threats. This study introduces a deep learning approach leveraging recurrent neural networks (RNNs) with long short-term memory (LSTM) to enhance phishing detection. The architecture is designed to capture sequential and temporal patterns in cloud interactions, enabling precise identification of phishing attempts. The model was trained and validated using a dataset of 10,000 samples, adapted from the PhishTank repository. This dataset includes a diverse range of attack vectors and legitimate activities, ensuring comprehensive coverage and adaptability to real-world scenarios. The key contribution of this work includes the development of a high-performance RNN-LSTM-based detection mechanism optimized for cloud-specific phishing patterns that achieve 98.88% accuracy. Additionally, the model incorporates a robust evaluation framework to assess its applicability in dynamic cloud environments. The experimental results demonstrate the effectiveness of the proposed approach, surpassing existing methods in accuracy and adaptability.

Keywords — cloud services, cybersecurity, deep learning, phishing detection, RNN-LSTM

1. Introduction

Phishing attacks have emerged as one of the most persistent cybersecurity threats, exploiting a combination of social engineering and technical manipulation to trick users into revealing sensitive information such as passwords, financial data, and personal identification details [1]. The widespread adoption of cloud computing services for data storage, processing, and communication has made these platforms a target for attackers. Cybercriminals are increasingly designing phishing campaigns that resemble legitimate communications from cloud service providers, leveraging authentic user interfaces to gain user trust and compromise security [2]. This growing threat highlights the need for advanced detection mechanisms tailored to the dynamic nature of the cloud environment.

The sophistication of phishing attacks in cloud ecosystems poses significant challenges to traditional detection methods.

Approaches such as URL blacklisting and heuristic-based systems struggle to keep up with evolving attack techniques [3]. Additionally, the rapid evolution of cloud services, coupled with their diverse applications, complicates the task of identifying malicious activities. Attackers often exploit multiple channels, including email, social networks, and messaging platforms, to deceive users. Furthermore, the rise of phishing-as-a-service (PhaaS) has simplified the process for cybercriminals, enabling large-scale, highly tailored phishing campaigns with minimal effort [4], [5].

To address these challenges, machine learning (ML) and deep learning (DL) techniques have proven to be effective tools for phishing detection. Among these, recurrent neural networks (RNNs) combined with long- and short-term memory units (LSTM) have demonstrated exceptional capability in analyzing sequential data and capturing long-term dependencies. This architecture is particularly well suited to detect behavioral anomalies in cloud-based systems, as it dynamically adapts to temporal patterns in user interactions [6].

This study focuses on leveraging the RNN-LSTM architecture to enhance phishing detection by analyzing irregular access behaviors, suspicious data transfers, and deceptive communications in cloud environments [7]. The presented research uses the publicly available PhishTank dataset, which has been adapted to simulate cloud-specific phishing scenarios. Although the dataset was not created from scratch, it has been tailored to include a diverse range of phishing attacks targeting cloud platforms. Examples include fake login pages, malicious links embedded in cloud communications, and phishing attempts to exploit cloud-based file-sharing services. By refining this dataset to focus on cloud-relevant attack vectors, the study bridges the gap between general phishing detection methods and the challenges posed in cloud environments.

The key contributions of this work are as follows:

1) design and implementation of an RNN-LSTM-based deep learning architecture, optimized to capture sequential and temporal patterns in cloud service interactions. This architecture enhances the detection of subtle anomalies indicative of phishing activities while maintaining computational efficiency.
2) adaptation of the PhishTank data set to reflect specific real-world phishing scenarios for cloud platforms, ensuring comprehensive representation of various attack strategies.
3) development of a robust detection mechanism that integrates RNN-LSTM capabilities with adjustments for

cloud-specific challenges, achieving high accuracy and adaptability across various cloud environments.

4) design of a comprehensive evaluation framework to assess system performance under real-world conditions, focusing on metrics such as detection accuracy, false positives, and adaptability to emerging threats.
5) introduction of enhanced protection strategies aimed at strengthening cloud security by mitigating the risks associated with phishing attacks and reducing potential data breaches and financial losses.

The remainder of this paper is structured as follows: Section 2 reviews related work on phishing detection with a particular focus on cloud-specific challenges and limitations. Section 3 details the proposed RNN-LSTM-based methodology and the adaptation of the dataset. Section 4 presents the experimental results and evaluates the performance of the detection system. Finally, Section 5 concludes the paper by summarizing the contributions and discussing directions for future research.

2. Literature Review

The issue of phishing in cloud environments has gained significant attention from researchers, leading to various approaches aimed at improving detection mechanisms. Existing studies provide a foundation for the methodology adopted in this work by exploring trends, vulnerabilities, and specific detection strategies for phishing attacks in cloud infrastructures. Traditional approaches, such as blacklisting URLs and rule-based systems, have proven effective to some extent, but struggle to keep pace with evolving phishing techniques. The application of deep learning and advanced pattern recognition methods has emerged as a promising alternative, offering improved accuracy and adaptability.

In previous studies, a variety of datasets containing legitimate and phishing-related data have been utilized in previous studies. For example, the authors of [8] proposed a phishing detection system that uses email data and employs algorithms such as support vector machines (SVM), naive Bayes (NB) and LSTM. Their method involves extracting features from emails and creating labeled datasets for classification. Although effective in identifying phishing patterns, this approach relies heavily on predefined features, making it less adaptable to emerging attack strategies.

In study [9], the authors developed a phishing email detection model based on deep learning techniques, such as graph convolutional networks (GCNs) combined with natural language processing. By focusing on the textual content of emails, the system demonstrated improved detection accuracy. However, the reliance on annotated training data presents challenges in scalability, as the manual labeling of large datasets can be time and resource consuming.

A different approach was taken in [10], where a data-driven model was proposed to detect phishing web pages using a multilayer perceptron (MLP). This study utilized a Kaggle dataset comprising 10 features and 10,000 websites, achieving training and testing accuracies of 95% and 93%, respectively. Despite these promising results, the model's dependence on limited features restricts its ability to generalize to phishing web pages employing novel evasion techniques.

The work [11] introduced PhishNot, a system for detecting phishing URLs using machine learning. By reducing the input features to 14, the authors optimized the system for fast processing and demonstrated a high detection accuracy of 97.5% using a random forest algorithm. Although this streamlined approach is computationally efficient, it may fail to capture complex patterns in sophisticated phishing URLs, potentially limiting its effectiveness in more advanced scenarios.

In article [12], a hybrid CNN-LSTM model was proposed to detect phishing attacks through image-based encoding. The system incorporated advanced techniques such as SMOTE and auto-encoder-based GANs to balance datasets and extract meaningful features. Grayscale images were used for the analysis and the approach achieved superior performance on multiple benchmarks. However, the computational complexity of this method may pose challenges for real-time detection or deployment in resource-constrained environments.

Collectively, these studies highlight the potential and limitations of existing phishing detection techniques. Based on their findings, this research aims to address key gaps by developing a more adaptable and scalable system to detect phishing activities in cloud-hosted environments.

2.1. Comparison and Limitations of Existing Methods

Table 1 presents a comparison of the phishing detection approaches reviewed in this study. It describes the type of model, training data used, accuracy achieved, complexity of the model, robustness to evolving phishing tactics, processing time, and key limitations associated with each method. This comparative analysis highlights the strengths and weaknesses of the existing approaches, offering valuable insights into their real-world applicability. By evaluating these characteristics, one can identify the most suitable phishing detection strategy for their specific needs.

Despite notable progress in phishing detection, the following limitations are evident in current methods:

1) Dependence on predefined features. The approach [8] is based on fixed set of characteristics, which may not effectively capture the nuances of emerging phishing strategies. Frequent updates are necessary to maintain its relevance in detecting advanced attacks.
2) Requirement for annotated training data. The method presented in [9] uses supervised learning, which requires large amounts of annotated data. This dependency increases the cost and effort required, limiting scalability and adaptability to novel phishing techniques.
3) Narrow scope and limited features. The framework [10] employs an MLP trained on a dataset with only ten attributes. This restricts the ability to handle phishing web pages using innovative tactics or evasion methods.
4) Simplified feature sets at the expense of complexity. The PhishNot system [11] reduces the input features to only 14

Tab. 1. Comparison of phishing detection approaches.

Approach	Model type	Training data	Accuracy	Complexity	Robustness	Processing time
[8]	SVM, NB, LSTM	Various datasets	95% (estimated)	Medium	Low	Fast
[9]	GCN	Annotated email data	93% (test)	High	Medium	Moderate
[10]	MLP	10,000 webpages	95% (training)	High	Low	Moderate
[11]	Random forest	Representative data	97.5%	Medium	Low	Fast
[12]	CNN-LSTM	3 benchmark datasets	Superior to others	High	Low	Slow

to achieve faster processing. However, this simplification may exclude intricate phishing URL patterns, reducing its effectiveness against sophisticated attacks.

5) High computational overhead. The CNN-LSTM approach shown in [12] involves advanced techniques such as SMOTE, GANs, and swarm intelligence, significantly increasing computational requirements. This complexity makes it unsuitable for real-time detection or resource-constrained environments.

3. Proposed Approach

In this section, we introduce a deep learning-based approach for detecting phishing in cloud environments leveraging an RNN-LSTM model. The proposed system is specifically designed to address the dynamic and evolving nature of phishing attacks in cloud settings by capturing sequential and temporal patterns in user interactions. The methodology consists of four phases: data acquisition and preprocessing, feature representation, model training, and performance evaluation.

The first phase involves gathering a comprehensive dataset comprising legitimate and phishing interactions in cloud services. The PhishTank[1]. dataset is curated to ensure its relevance to real-world scenarios. To prepare the dataset for analysis, it undergoes pre-processing steps, including cleaning to remove inconsistencies, normalization to standardize data attributes, and the extraction of relevant features such as URL length, IP address presence, and HTTPS usage. This ensures that the data set is high quality and suitable for deep learning.

The second phase focuses on transforming the data into a format compatible with the RNN-LSTM architecture. Textual input, such as URLs, are converted to numerical representations using character-level embeddings. These embeddings preserve contextual relationships within the data, enabling the model to analyze sequential patterns effectively and detect phishing behaviors.

The third phase is dedicated to training the RNN-LSTM model. The architecture is designed to capture temporal dependencies in the data, using LSTM layers to retain essential information over time. Regularization techniques, such as dropout, are applied during training to prevent overfitting and improve the generalization of the model. Hyperparameters, including the learning rate, batch size, and the number of LSTM units, are optimized through systematic grid search and cross-validation.

Finally, the model performance is evaluated using metrics such as accuracy, precision, recall, and F1 score. These metrics assess the system's effectiveness in distinguishing phishing attempts from legitimate interactions. The analysis also includes an evaluation of the importance of individual characteristics in the detection process, providing information on the decision-making and the patterns it identifies as indicative of phishing behavior.

3.1. Data Exploration and Analysis

PhishTank, a community-curated repository of validated phishing websites, serves as the main data source for this study. This database aggregates reports of suspicious websites from users and validates their authenticity through expert review. Each entry includes critical information such as the URL of the phishing site's URL, submission time, verification status, and operational state. PhishTank provides these data in both API and CSV formats, making them highly accessible for machine learning research [13]. For this investigation, we used 10,000 samples, evenly categorized as authentic or phishing based on 18 distinct attributes that focus on URL structure and activity patterns. These attributes enable a detailed analysis of phishing activity, including detection patterns, response times, and the lifecycle of malicious URLs.

A preliminary exploration of the dataset revealed patterns in phishing behavior and response efficiency. Key observations include:

- **Phishing lifecycle**. The dataset records both submission and verification timestamps, allowing for analysis of the time taken to confirm a phishing report. This provides insight into response efficiency.
- **URL characteristics**. Phishing URLs often feature irregular structures, including excessive length, presence of IP addresses, or frequent use of special characters, which differentiate them from legitimate URLs.

[1]The PhishTank dataset is accessible at `https://phishtank.org`

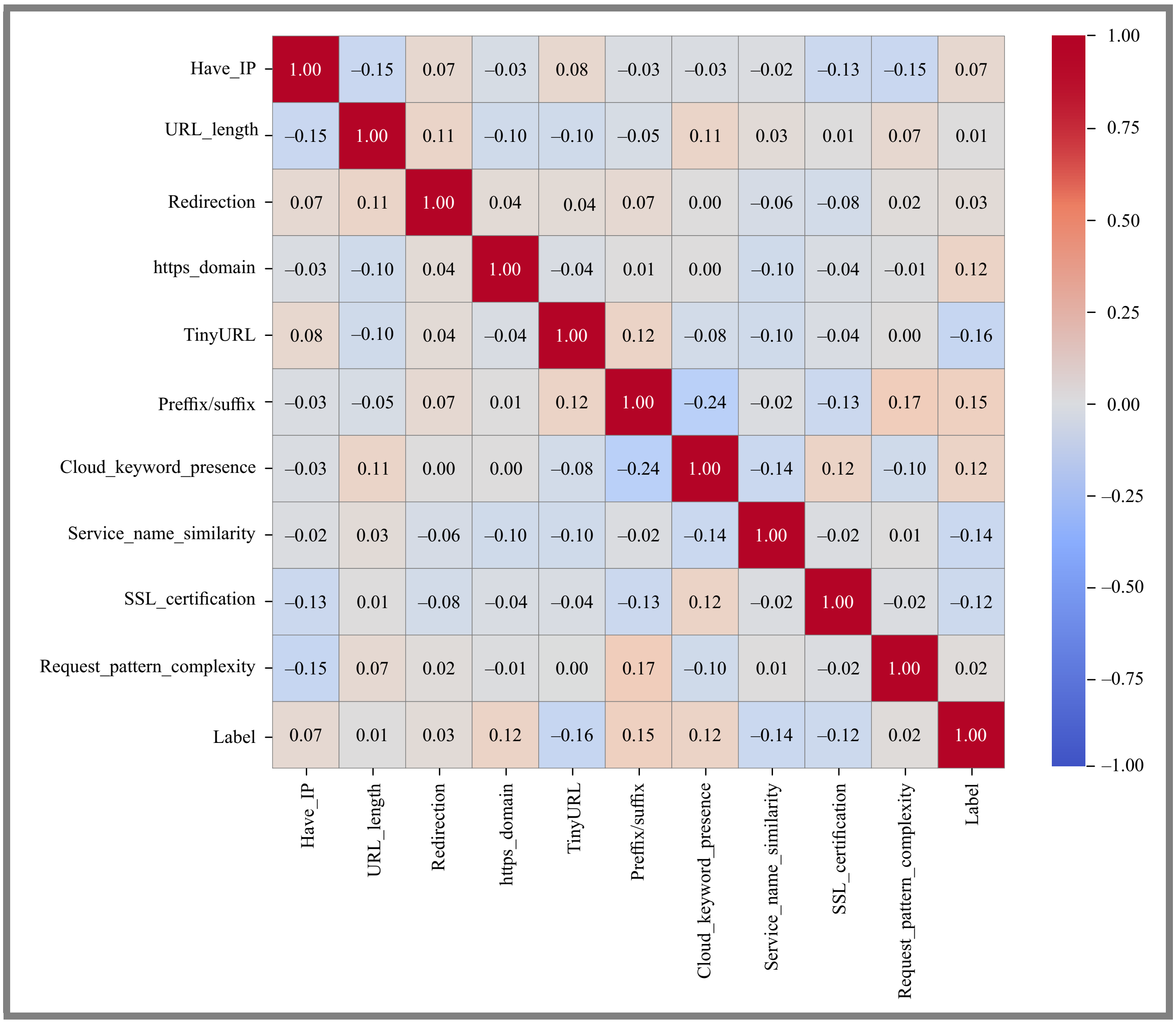

Fig. 1. Correlation matrix of features.

- **Verification patterns**. Most verified phishing sites are confirmed within a short time frame, highlighting the effectiveness of the PhishTanks validation process.

To ensure the quality of the dataset for training and evaluation, the following steps were undertaken:

- removal of incomplete or duplicate entries to maintain data integrity,
- normalization of URL features to ensure consistency across samples,
- label encoding of target variables, with phishing URLs labeled as "1" and legitimate URLs as "0".

These steps ensure that the dataset is both clean and ready for robust feature extraction and model training.

3.2. Feature Extraction

To ensure robust detection, we extracted significant features relevant to identifying phishing attempts in cloud environments. The following key attributes, aligned with our study and correlation analysis, are particularly essential:

– Having_IP – indicates the presence of an IP address in the URL, a tactic commonly used in phishing attacks to bypass domain-based detection.

– URL_Length – represents the total length of the URL. Phishing URLs often have unusually long paths to obfuscate their content.

– Redirection – counts the number of redirects in a URL. Phishing websites frequently use multiple redirections to conceal their malicious intent.

– Https_Domain – verifies the presence of HTTPS in the domain. Phishing attackers often exploit this to create a false sense of security.

– TinyURL – identifies the use of URL shortening services, which are frequently leveraged in phishing campaigns to obscure malicious destinations.

- Prefix/Suffix – checks for the presence of prefix or suffix patterns in domain names, as these can mimic legitimate domains.
- Cloud_Keyword_Presence – detects cloud-related keywords (e.g., cloud, aws, azure) in the URL, which phishers may use to impersonate trusted cloud providers.
- Service_Name_Similarity – analyzes the similarity between the URL content and popular cloud service names to deceive users into assuming legitimacy.
- SSL_Certification – determines the validity of SSL certificates as phishing websites may use SSL to appear credible.
- Request_Pattern_Complexity – evaluates the complexity of the URL's request pattern, as phishing sites often include irregular or unusually structured paths.
- Label – the target variable, classifying URLs as phishing or legitimate.

These features are essential for identifying phishing websites by analyzing both structural and behavioral patterns. Their correlations, as shown in Fig. 1, highlight their significance in cloud phishing detection models in the form of a correlation matrix.

Each dataset feature is labeled 1 for phishing websites and 0 for legitimate ones. Due to the size (20,000 entries: 10,000 phishing URLs and 10,000 authentic), seven significant features were extracted. These qualities help identify legitimate and fraudulent websites – see Fig. 2.

3.3. Proposed Phishing Detection Model

This subsection introduces the architecture and functionality of the proposed phishing detection model. The model leverages an RNN framework augmented with LSTM layers to effectively capture sequential and temporal patterns in cloud service interactions. By analyzing these behavioral patterns over time, the model can differentiate between phishing and legitimate activities with high precision.

The proposed architecture is designed to process input sequences while preserving contextual and temporal dependencies critical for identifying phishing attempts. The model is made up of multiple layers, each serving a specific role in the detection pipeline.

In the embedding layer, input tokens are first converted into dense vectors that represent their semantic meaning. This layer is crucial for the model to understand the contextual links. The expression of embedding transformation is as follows:

$$\mathbf{E}(X) = \{e(x_1), e(x_2), \ldots, e(x_T)\} \,, \tag{1}$$

where $e(x_i)$ is the embedding vector for token x_i.

Embedding tokens into dense representations gives the model a numerical representation of text that improves interpretability for future layers, helping it uncover phishing patterns. The quality of the embeddings can be improved using pre-trained models like Word2Vec or GloVe, which can be integrated into the embedding layer in such a way:

$$e(x_i) = \mathbf{W} \cdot v_i \,, \tag{2}$$

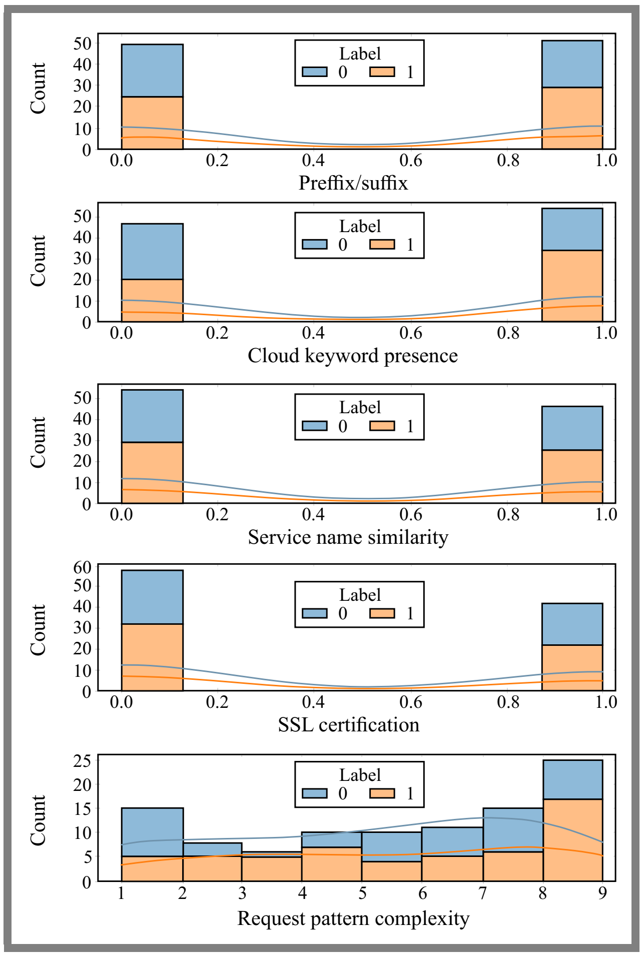

Fig. 2. Histogram of feature visualization.

where $\mathbf{W}$ is the weight matrix and v_i is the vector representation of the pre-trained model.

The LSTM layers capture embedded data sequence dependencies. LSTMs can keep crucial information and discard less important information, making them suitable for detecting phishing patterns that depend on earlier or later sequences. Each LSTM layer computes hidden states h_t and cell states C_t at each time step t, as defined by:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad \text{(forget gate)} \,, \tag{3}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad \text{(input gate)} \,, \tag{4}$$

$$\tilde{C}_t = \text{tgh}(W_C \cdot [h_{t-1}, x_t] + b_C) \quad \text{(cell candidate)} \,, \tag{5}$$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \quad \text{(cell state)} \,, \tag{6}$$

$$h_t = o_t \cdot \text{tgh}(C_t) \quad \text{(hidden state)} \,. \tag{7}$$

Eqs. (3) – (7), f_t, i_t, C_t, $\tilde{C}_t$, h_t, and o_t represent the forget gate, the input gate, cell candidate, the cell state the hidden state, and output gate, respectively.

The LSTM technique selectively updates and preserves cell states through these gates to keep the model's phishing detection knowledge. To enhance the model's capacity to capture long-range dependencies, we can also implement a bidirectional LSTM, which processes the input sequence in both forward and backward directions.

$$h_t = \text{LSTM}(x_t, h_{t-1}, C_{t-1}) + \text{LSTM}(x_t^{\text{rev}}, h_{t-1}^{\text{rev}}, C_{t-1}^{\text{rev}}) \ . \quad (8)$$

A fully connected layer and output layer evaluate the final hidden state h_T after LSTM layers, converting the learned features into a probability score for phishing attempts. The prediction formula is:

$$\hat{y} = \sigma(W_O \cdot h_T + b_O) \ , \quad (9)$$

where σ denotes sigmoid activation.

The probability of the sequence being phishing is between 0 and 1 with this function. These layers decide by mapping the RNN's learned properties into a binary outcome. Next, the model is trained by minimizing a binary cross-entropy loss function that compares true labels and predicted probability. The loss function encourages accurate model predictions by reducing the error between true labels and predictions. The definition of a loss function is as follows:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} \left(y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)\right) \ , \quad (10)$$

where, y_i represents the true label, while $\hat{y}_i$ denotes the predicted probability for each instance.

Regularization methods such as dropout can be added after LSTM layers to enhance generalization and reduce overfitting. The dropout layer allows the model to avoid over-reliance on any single feature by randomly setting a percentage p of input units to zero during training. A mathematical expression is:

$$h_t^{\text{drop}} = h_t \cdot r \ , \quad (11)$$

where r is a random variable drawn from a Bernoulli distribution with parameter p.

As evaluation metrics, precision, recall, and F1 score measures are used to assess the model's performance. The F1 score is calculated as follows:

$$\text{F1 score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \ . \quad (12)$$

Building a robust phishing detection system for cloud environments requires thorough training and meticulous tuning to adapt to the constantly evolving nature of phishing threats.

The performance of the model is optimized through careful selection of hyper parameters, including the learning rate, batch size, number of LSTM layers, and dropout rate. Hyperparameter tuning is performed systematically using grid search and cross-validation to identify the best configuration. Regularization techniques are applied to mitigate overfitting, ensuring that the model maintains strong predictive capabilities when exposed to unseen data.

The following parameters are assumed during model training:

- Embedding dimension = 256, enhancing the representation of tokens to capture more intricate semantic relationships while still being manageable in terms of computational resources.
- LSTM units = 256 for each LSTM layer, increasing the model's ability to effectively learn and retain complex sequential dependencies in the data.
- Dropout rate = 0.3, striking a balance between reducing overfitting and preserving enough model capacity to ensure robust learning.
- Batch size = 64, allowing for more stable gradient updates while also accommodating larger data chunks, which can lead to better convergence.
- Epochs = 50, with an emphasis on using early stopping based on validation loss to ensure that the model does not overtrain while still having sufficient training time to achieve high accuracy.

The computationally effective embedding dimension of 256 captures more intricate semantic links and improves token representation. Each 256-unit LSTM layer helps the model learn and retain complicated sequential data dependencies. A dropout rate of 0.3 balances overfitting with learning model robustness. With a batch size of 64, larger training data chunks improve gradient updates and convergence. To avoid overtraining and ensure accuracy, the model is trained for more than 50 epochs and stopped when validation loss occurs.

4. Performance Evaluation

The proposed model was built and tested using Google Colab, a powerful cloud-based machine learning model deployment and testing tool. Free GPU resources make Google Colab ideal for testing complex algorithms and datasets.

The metrics considered during the evaluation of the proposed approach include:

- **Accuracy**, comparing predicted labels to actual labels gives a clear picture of the model performance.
- **Precision, recall and F1 score**. Precision assesses a model's ability to correctly identify affirmative phishing attempts, while recall measures its ability to catch all actual incidents. The F1 score is a harmonic mean of precision and recall that accounts for false positives and negatives, which is essential to analyze model performance in imbalanced datasets.
- **Confusion matrix**. This tool provides a detailed breakdown of the model's classification performance by showing the true positives, true negatives, the false positives, and false negatives. Analyzing the confusion matrix allows for a deeper understanding of specific weaknesses in the model, such as misclassifying legitimate URLs as phishing attempts, which can be particularly detrimental in practical applications.
- **Loss function**. In the context of RNNs, monitoring the loss function during training is vital for assessing model performance. A decreasing loss value indicates that the model is learning effectively, while fluctuations can signal over- or underfitting issues.

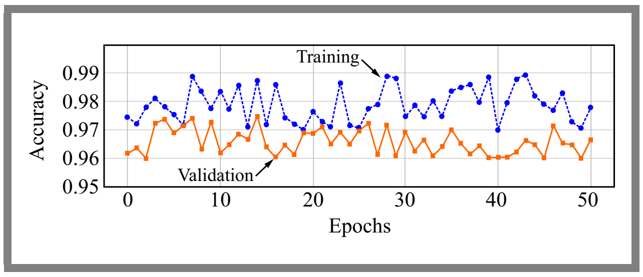

Fig. 3. Model training and validation accuracy performance.

4.1. Results Analysis

Figure 3 illustrates the performance metrics of the RNN deep learning model used to detect phishing attempts in cloud-based applications. The algorithm consistently identifies complex traits that separate phishing interactions from real ones, with training accuracy between 97% and 99%. The model validation accuracy varies from 96% to 97.5%, demonstrating its ability to generalize to unknown data, crucial for real-world applications.

Training loss of optimization targets, the difference between predicted outputs and targets. Low validation loss indicates learning efficacy, while low training loss indicates that model predictions meet goals. Data pattern capture issues may cause high validation loss, while good learning indicates low validation loss.

The model may overfit or underfit if it memorizes the training data or is too basic to detect patterns. Tracking training loss is essential. The robust model has 98.885% training accuracy and 97.75% validation accuracy, indicating that there are no overfits or underfitting. The model recognizes substantial training data patterns and generalizes well to novel samples. Real-world phishing detection requires balanced models with low false positives. Consistent RNN performance improves cloud phishing security.

Figure 4 shows the RNN deep learning model loss performance metrics to identify phishing attempts in cloud-based applications. The model minimizes the difference between expected outputs and targets with a training loss of 0.025 to 0.038. Reduced training loss shows that the model can handle complicated data patterns. The validation loss is 0.053 to 0.068, indicating a strong new data generalization. The model finds fundamental patterns without overfitting the training dataset, since validation loss is always less than training loss.

Training and validation loss might show overfitting and underfitting, when the model memorizes training data but misses data patterns. The resilience balances detail and generalization with 0.031 training loss and 0.060 validation loss. Phishing detection models with low false positives must balance performance. Loss performance measurements show RNN architecture stability, securing cloud from phishing.

Figure 5 shows the confusion matrix for the proposed model, assessing its classification performance. This matrix indicates the distinction and proposes improvements. High false negatives signal that the model must be modified to detect more positives. Overall, the balanced true positives and true negatives of the matrix show that the program could detect

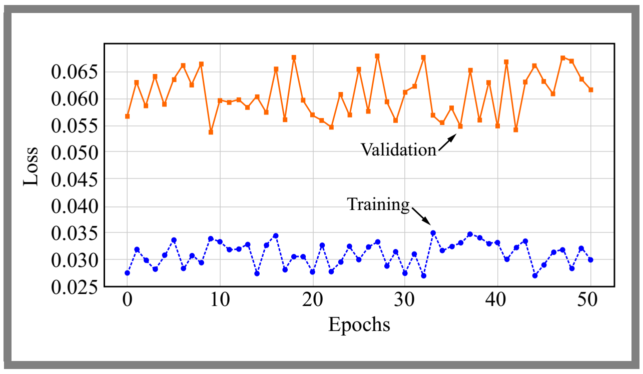

Fig. 4. Performance of the training and validation loss of proposed model.

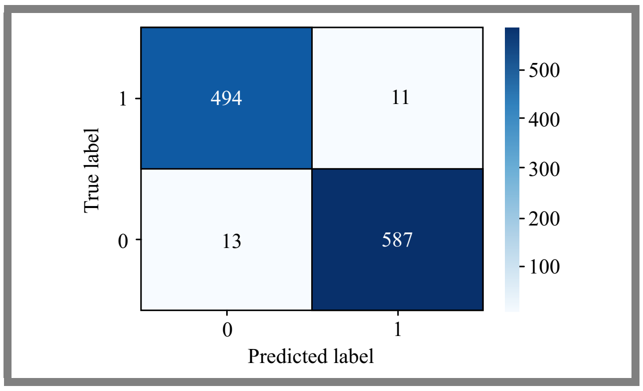

Fig. 5. Confusion matrix for the proposed model.

phishing attempts. This robust result shows that the proposed model is acceptable for real-world applications with precision enhancement potential.

Table 2 summarizes model's evaluation metrics. The model classifies the dataset samples well with 0.988 accuracy. The macro- and micro-averaged precision and recall indicate positive sample detection with low false positives. The recall scores and macro- and microaveraged precision of 0.979 and 0.982 demonstrate its ability to recover relevant samples. The methodology works because macro and micro F1 scores average precision and recall into a harmonic mean. The macro-averaged F1 score of 0.977 and micro-averaged F1 score of 0.984 indicate balanced and trustworthy performance. These results show that the proposed approach can generalize and reduce classification errors for real-world applications.

Table 3 shows the accuracy and loss metrics for model training and validation. The high training accuracy of 0.988 and the validation accuracy of 0.977 show the model ability to capture patterns in the training data and generalize to new data. A low training loss of 0.0318 shows minimal inaccuracy in predicting training samples, while a validation loss of 0.0537 indicates consistent performance on unseen data without over- or underfitting. These indicators demonstrate the stability and learning potential, promising applications in the real world.

Table 4 analyzes the accuracy of the model using PhishTank dataset, proving its effectiveness in phishing detection. The RNN-LSTM model achieves maximum accuracy at 98.885%, surpassing other methods. Traditional approaches like KNN (87.98%) and random forest classifier (94.262%) were less

Tab. 2. Evaluation metrics of the proposed model.

Evaluation parameter	Value
Accuracy score	0.988
Macro averaged precision	0.979
Micro averaged precision	0.982
Macro averaged recall	0.972
Micro averaged recall	0.984
Macro averaged F1 score	0.977
Micro averaged F1 score	0.984

Tab. 3. Accuracy and loss performance of the proposed model.

Evaluation metric	Performance value
Training accuracy	0.988
Validation accuracy	0.977
Training loss	0.0318
Validation loss	0.0537

Tab. 4. Comparison of accuracy for different models in the PhishTank dataset.

Ref.	Approach	Accuracy	Dataset
[5]	Multi-model	97.81%	PhishTank
[7]	KNN	87.980%	PhishTank
[14]	NLP	97.981%	PhishTank
[15]	RFC	94.262%	PhishTank
[16]	SVM	94.130%	PhishTank
Proposed	RNN-LSTM	98.885%	PhishTank

accurate than multi-model (97.81%). With a precision of 97.981%, the NLP model is useful but not as accurate as the RNN-LSTM model. The SVM and the random forest classifier have a precision of 94.130% and 94.262%, respectively, indicating that traditional techniques are less effective than deep learning in detecting phishing URLs. The comparison indicates that the RNN-LSTM model improves phishing detection and is acceptable for accurate real-world applications.

4.2. Related Work Comparison

Table 4 compares the phishing detection methods applied to the PhishTank dataset, showcasing both traditional models like KNN, RFC, and SVM, and advanced techniques such as multimodel and NLP approaches. The proposed RNN-LSTM model achieves the highest accuracy of 98.885%, highlighting its ability to effectively leverage sequential and temporal patterns for superior phishing detection in cloud environments.

4.3. Integration in Cloud Computing Environments

The proposed phishing detection model integrates seamlessly into cloud environments, serving as a vital component of the security framework. Positioned at the network entry point, it enables real-time monitoring of incoming traffic, including URLs in emails, shared cloud storage links, and web requests, effectively preventing malicious traffic from reaching users or critical services.

Designed as a scalable microservice, the system leverages technologies like Docker and Kubernetes to dynamically adjust resources based on traffic volume. Its API-based design allows easy integration with existing security tools, such as firewalls and intrusion detection systems. Continuous learning is supported through updates from cloud-based threat intelligence feeds, ensuring adaptability to emerging phishing tactics and maintaining high detection accuracy.

Key use cases for this integration include:

- email security – analyzing and detecting phishing links embedded in emails hosted on cloud platforms,
- web gateway protection – filtering malicious URLs in real-time to block access to phishing websites,
- cloud storage monitoring – scanning shared links and files for potential phishing attempts,
- application-level protection – safeguarding cloud-based applications by monitoring API calls and user interactions for anomalies.

5. Conclusions

This paper presents a robust approach to phishing detection in cloud environments by combining RNNs with LSTM units. Leveraging sequential data processing and the ability to retain long-term dependencies, the RNN-LSTM model efficiently captured temporal patterns within cloud service interactions. This capability significantly improved accuracy and resilience in detecting phishing attempts. The model was trained on a comprehensive dataset of 10,000 cloud-based interactions, which encompassed a diverse range of applications, storage solutions, and email services. This data set facilitated robust training and ensured adaptability in various cloud scenarios.

Achieving an accuracy of 98.88%, the proposed model demonstrated the potential of deep learning to significantly enhance phishing detection in cloud computing, thereby reinforcing cybersecurity by distinguishing legitimate from malicious interactions.

Although the results are promising, there is significant potential for further research to improve and extend this work. Future research could focus on integrating advanced architectures, such as transformers or hybrid deep learning models, to enhance accuracy and efficiency. Adding real-time data streams and multilingual phishing datasets would increase the applicability and address the global nature of phishing threats. Furthermore, testing the system in real-world cloud environments would provide important insights into its scalability, reliability, and practical deployment, helping to close the gap between research and real-world applications.

References

[1] C. Sharma and C. Sharma, "Cloud Computing Security: Threats and Mitigation Strategies", *2024 International Conference on Signal Processing and Advance Research in Computing (SPARC)*, vol. 1, Lucknow, India, 2024.

[2] M. Dawood *et al.*, "Cyberattacks and Security of Cloud Computing: A Complete Guideline", *Symmetry*, vol. 15, no. 11, 2023 (https://doi.org/10.3390/sym15111981).

[3] P. Prajapati *et al.*, "Phishing E-mail Detection Using Machine Learning", *Smart Innovation, Systems and Technologies*, vol. 392, 2023 (https://doi.org/10.1007/978-981-97-3690-4_32).

[4] J.K. Samriya *et al.*, "Blockchain and Reinforcement Neural Network for Trusted Cloud Enabled IoT Network", *IEEE Transactions on Consumer Electronics*, vol. 70, no. 1, pp. 2311–2322, 2024 (https://doi.org/10.1109/TCE.2023.3347690).

[5] B. Jha, M. Atre, and A. Rao, "Detecting Cloud-based Phishing Attacks by Combining Deep Learning Models", *2022 IEEE 4th International Conference on Trust, Privacy and Security in Intelligent Systems, and Applications (TPS-ISA)*, Atlanta, USA, 2023 (https://doi.org/10.1109/TPS-ISA56441.2022.00026).

[6] S.R. Alotaibi *et al.*, "Explainable Artificial Intelligence in Web Phishing Classification on Secure IoT with Cloud-based Cyber-physical Systems", *Alexandria Engineering Journal*, vol. 110, pp. 490–505, 2024 (https://doi.org/10.1016/j.aej.2024.09.115).

[7] P. Ramadevi *et al.*, "Analysis of Phishing Attack in Distributed Cloud Systems Using Machine Learning", *2023 Second International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT)*, Trichirappalli, India, 2023 (https://doi.org/10.1109/ICEEICT56924.2023.10157447).

[8] U.A. Butt *et al.*, "Cloud-based Email Phishing Attack Using Machine and Deep Learning Algorithm", *Complex& Intelligent Systems*, vol. 9, pp. 3043–3070, 2023 (https://doi.org/10.1007/s40747-022-00760-3).

[9] A. Alhogail and A. Alsabih, "Applying Machine Learning and Natural Language Processing to Detect Phishing Email", *Computers and Security*, vol. 110, art. no. 102414, 2021 (https://doi.org/10.1016/j.cose.2021.102414).

[10] I. Saha *et al.*, "Phishing Attacks Detection Using Deep Learning Approach", *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, Tirunelveli, India, 2020 (https://doi.org/10.1109/ICSSIT48917.2020.9214132).

[11] M.M. Alani and H. Tawfik, "PhishNot: A Cloud-based Machine Learning Approach to Phishing URL Detection", *Computer Networks*, vol. 218, art. no. 109407, 2022 (https://doi.org/10.1016/j.comnet.2022.109407).

[12] M. Elberri, U. Tokeser, J. Rahebi, and J. Lopez-Guede, "A Cyber Defense System Against Phishing Attacks with Deep Learning Game Theory and LSTM-CNN with African Vulture Optimization Algorithm (AVOA)", *International Journal of Information Security*, vol. 23, pp. 2583–2606, 2024 (https://doi.org/10.1007/s10207-024-00851-x).

[13] E.A. Aldakheel *et al.*, "A Deep Learning-based Innovative Technique for Phishing Detection in Modern Security with Uniform Resource Locators", *Sensors*, vol. 23, no. 9, 2023 (https://doi.org/10.3390/s23094403).

[14] F. Sadique, R. Kaul, S. Badsha, and S. Sengupta, "An Automated Framework for Real-time Phishing URL Detection", *Proc. of the 10th Annual Computing and Communication Workshop and Conference (CCWC)*, pp. 335–341, 2020 (https://doi.org/10.1109/CCWC47524.2020.9031269).

[15] O. Sahingoz, E. Buber, O. Demir, and B. Diri, "Machine Learning Based Phishing Detection from URLs", *Expert Systems with Applications*, vol. 117, pp. 345–357, 2018 (https://doi.org/10.1016/j.eswa.2018.09.029).

[16] R. Rao, T. Vaishnavi, and A. Pais, "CatchPhish: Detection of Phishing Websites by Inspecting URLs", *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, pp. 813–825, 2019 (https://doi.org/10.1007/s12652-019-01311-4).

Oussama Senouci, Ph.D., Associate Professor
Computer Science Department
https://orcid.org/0000-0002-5345-0713
E-mail: oussama.senouci@univ-bba.dz
University of Mohamed El Bachir El Ibrahimi, Bordj Bou Arreridj, Algeria
https://www.univ-bba.dz

Nadjib Benaouda, Ph.D., Associate Professor
Computer Science Department
https://orcid.org/0000-0002-4361-9597
E-mail: nadjib.benaouda@univ-bba.dz
University of Mohamed El Bachir El Ibrahimi, Bordj Bou Arreridj, Algeria
https://www.univ-bba.dz