IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.1411149

Real-World Phishing and Smishing Detection Using Deep Learning: A Comparative Study of LSTM, GRU, and GloVe Embeddings

ANNASAHEB M. CHOUGULE[*1], DR. KAVITA S. OZA[2], VISHAL T. PATIL[3], DR. ROHIT B. DIWANE[4]

Department of Computer Science, Shivaji University, Kolhapur Maharashtra, India[1,2,4]

Department of Computer Science, Miraj Mahavidyalaya, Miraj, Maharashtra, India[3]

https://orcid.org/0000-0002-4920-3656[1]

https://orcid.org/0000-0003-4636-9674[2]

https://orcid.org/0009-0000-8838-8377[3]

https://orcid.org/0000-0003-0674-4265[4]

*Corresponding Author

Abstract: Phishing and smishing attacks have rapidly increased in digital communication platforms, exploiting user trust to steal personal and financial information. Traditional blacklist and rule-based detection systems lack adaptability and fail to detect evolving or zero-day attacks. Although deep learning has shown promise in text-based threat detection, existing research often focuses on a single architecture, lacks real-world datasets, or provides limited benchmarking across models. To address these gaps, this study presents a comparative evaluation of three deep learning models—RNN-LSTM, RNN-GRU, and GloVe-enhanced LSTM—for phishing and smishing text classification. A dataset of 27,000 real-world messages, collected from cybersecurity units and extended with controlled synthetic samples, was preprocessed using tokenization, stemming, padding, and semantic embeddings. Each model underwent structured hyperparameter optimization with dropout, L2 regularization, and early stopping to enhance generalization. Experimental results show that the GloVe-LSTM model achieved the highest performance with 90.07% test accuracy and a 90.16% F1-score, closely followed by tuned LSTM and GRU models. Statistical validation using a McNemar test confirmed no significant difference in model performance ($p > 0.05$). These findings demonstrate that semantic embeddings significantly improve phishing and smishing detection accuracy, supporting scalable deployment in cybersecurity systems such as email filtering, telecom SMS gateways, and digital fraud prevention platforms.

Keywords: Phishing Detection; Smishing; Deep Learning; LSTM; GRU; GloVe Embeddings; NLP; Cybersecurity

I. INTRODUCTION

Phishing and smishing attacks pose significant threats to individuals, organizations, and critical infrastructure worldwide. Phishing relies on deceptive emails, while smishing targets victims via SMS, often impersonating trusted entities to extract sensitive information or install malicious software.

Traditional detection systems rely on blacklists or handcrafted features derived from message content or URLs. However, these methods are inadequate against zero-day attacks or sophisticated adversarial campaigns that leverage subtle textual manipulations to bypass filters.

Deep learning, particularly recurrent neural networks (RNNs) like Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), provides a powerful alternative for understanding sequential dependencies in text. Embedding methods such as GloVe allow models to capture rich semantic relationships, enabling better generalization across varied linguistic patterns.

This paper investigates the effectiveness of LSTM, GRU, and GloVe-augmented LSTM models for phishing and smishing message classification. We collected and annotated a comprehensive dataset reflecting real-world attack patterns. Hyperparameter tuning was systematically applied to each model to ensure fair performance comparison. Our study addresses three key questions:

1. How do LSTM and GRU models compare in phishing/smishing detection?
2. What impact does integrating GloVe embeddings have on detection performance?
3. How do different hyperparameter settings influence model generalization?

Our contributions are:

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.1411149

- A curated phishing/smishing dataset, including challenging, evasive samples.
- A systematic comparative study of LSTM, GRU, and GloVe+LSTM models.

Detailed hyper parameter exploration to identify optimal configurations

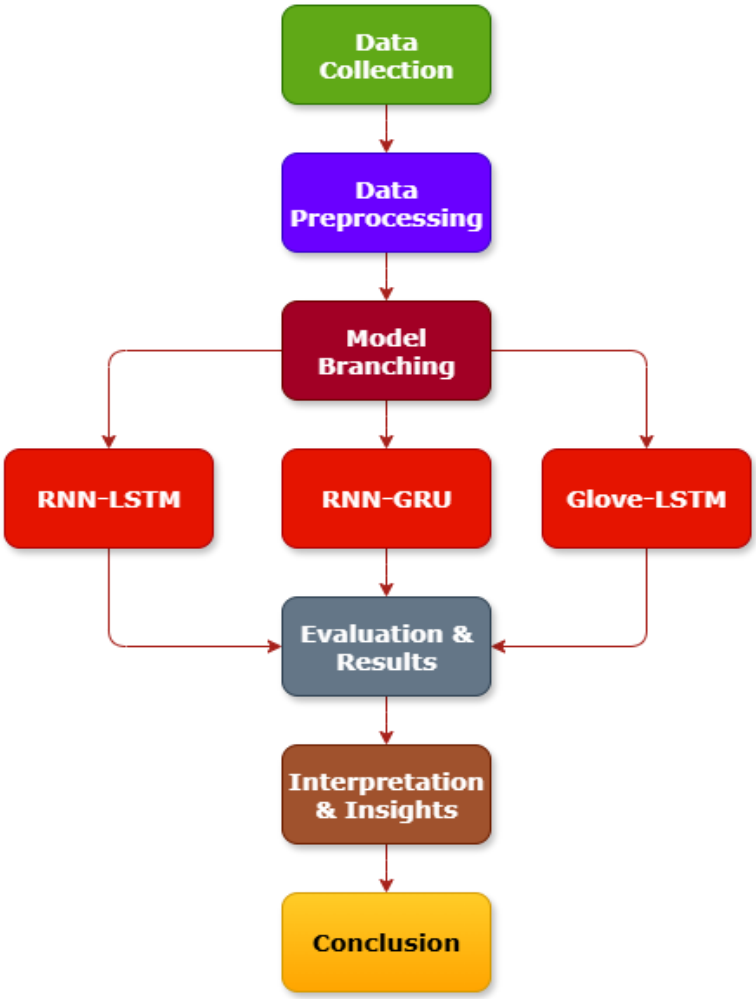


Figure 1: Flowchart of study

As shown in Figure 1, describes the workflow of research study

II. LITERATURE REVIEW

[1] Author proposed a phishing detection framework utilizing CNN, LSTM, and hybrid LSTM-CNN architectures. Their deep learning models demonstrated impressive accuracy, with CNN achieving 99.2%. The study emphasizes end-to-end feature learning and automation in phishing URL detection. This work validates the superiority of DL over ML in dynamic threat environments [1].

[2] Author explored the combination of LSTM and CNN models for phishing attack detection. Their fusion approach significantly outperformed individual models by capturing both spatial and temporal URL features. This study highlights the advantage of hybrid neural architectures in understanding malicious intent embedded within URLs [2].

[3] AntiPhishStack, a two-phase stacked generalization model that integrates ML and LSTM with character-level TF-IDF for phishing URL detection. Using a meta-XGBoost classifier, their framework achieved 96.04% accuracy. The study addresses asymmetry in conventional methods by proposing a symmetrical learning approach [3].

[4] Implemented a comparative multimodel approach involving LSTM, Bi-LSTM, and GRU for phishing URL detection. Bi-LSTM emerged as the most accurate with 99% accuracy, emphasizing the benefit of bidirectional context in sequential URL analysis. The study offers architectural insights and real-time applicability for URL-based attacks [4].

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025
DOI: 10.17148/IJARCCE.2025.1411149

[5] the research conducted a performance comparison between GRU and BERT for phishing URL detection. GRU slightly outperformed BERT (98.35% vs 97.4%) while requiring fewer computational resources, recommending GRU for large-scale applications. The work adds practical insights for model selection in cyber-defense systems [5].
[6] A phishing detection model that analyzes webpage content text using NLP and deep learning techniques rather than URLs. By employing GloVe embeddings and sequential models (LSTM, BiLSTM, GRU, BiGRU), they preserved semantic-syntactic relationships in input text. BiGRU performed best with 97.39% accuracy, proving the effectiveness of context-aware text embeddings [6].
[7] Study investigated evasive techniques in SMS spam filtering using traditional and deep ML models. Their study introduced a large SMS spam dataset and evaluated model robustness against character-, word-, and sentence-level obfuscations. Results showed widespread vulnerability in existing models, urging stronger semantic-aware models resilient to adversarial manipulations [7].
[8] Study developed a hybrid CNN-LSTM model to detect SMS spam in both Arabic and English. The architecture leveraged CNN for local feature extraction and LSTM for sequence learning, outperforming traditional ML methods with 98.37% accuracy. Their model effectively handled code-mixed environments, a frequent challenge in multilingual messaging [8].
[9] A deep learning-based intelligent framework for SMS and email spam detection. The system integrates deep RNN variants and attention-based mechanisms to address multilingual spam detection. Results indicate that the framework generalizes well across datasets and languages, enhancing adaptability in modern messaging platforms [9].
[10] Researcher conducted an extensive review of 30 machine learning-based studies focused on detecting email phishing attacks. The paper presents key features such as URL, header, and content-based analysis, highlighting performance across models like SVM, RF, and DL architectures. It identifies that while high accuracy is often achieved, many models are trained on limited or outdated datasets and lack robustness. The review also emphasizes the need for deep learning, ensemble techniques, and hybrid approaches to improve detection and generalizability [10].

2.1. Research gap:

1. Lack of Unified models for Phishing and Smishing Detection

Most existing studies address either phishing (typically via email or URL analysis) or smishing (via SMS text classification) independently. However, real-world cyber-attacks often span multiple communication channels simultaneously. A significant research gap exists in the development of a single unified model capable of detecting both phishing and smishing attacks in an integrated system. This limits the effectiveness of traditional approaches in handling evolving, multi-channel threat vectors.

2. Limited Availability and Use of Real-Time, Self-Collected Datasets

The majority of existing models are trained and evaluated on small-scale, outdated, or publicly available benchmark datasets such as the UCI SMS spam dataset or URL-based corpora. These datasets lack contextual diversity and do not reflect modern attack patterns, especially in local languages or regional settings. In contrast, the current research employs a self-collected, real-time dataset that includes phishing and smishing messages in realistic, user-generated formats, making it more representative of actual user environments. This addresses the critical issue of data generalizability and domain relevance in phishing/smishing detection.

3. Absence of Fair and Systematic Comparative Evaluation across ML/DL Models

Many reviewed papers assess only a limited number of algorithms (usually one or two), often without standardized evaluation criteria across datasets. Consequently, model performance cannot be reliably compared or generalized. The present study overcomes this limitation by systematically evaluating multiple machine learning and deep learning algorithms, including RNN-LSTM, RNN-GRU, and GloVe-LSTM, using the same dataset and metrics (accuracy, precision, recall, F1-score), thereby ensuring fair benchmarking and reproducibility.

4. Over-Reliance on Text-Only Features without Multi-Modal Context

Most existing models rely solely on textual features extracted from message bodies or URLs, ignoring other potentially informative signals such as sender metadata, language context, domain behavior, or embedded URLs. Moreover, multi-modal integration (e.g., combining message content with metadata or external link behavior) is rarely explored. This study identifies this gap and highlights the potential of incorporating multi-feature or multi-view data sources to enhance detection accuracy.

5. Inadequate Support for Regional, Code-Mixed, and Multilingual Messages

Very few phishing/smishing detection studies account for regional languages or code-mixed messages, which are common in countries like India. Most models are trained on English-only datasets, and fail to perform on real-world data where users frequently communicate in Marathi-English or Hinglish-style messages. This study addresses the need to

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.1411149

build models that are linguistically adaptive and culturally contextual, ensuring more accurate detection in multilingual environments.

III. MATERIALS AND METHODS

a. Data Collection and Preprocessing

The initial phase of the research involves collecting a diverse set of email and SMS datasets, including phishing, smishing, and legitimate messages. The comprehensive dataset, spanning 2023 to 2025, consists of 25,000 records collected through visits to Cyber Cells like Sangli, Kolhapur and Pimpari- Chinchwad Pune and 2000 records synthetic data generated through Gen AI. Therefor total 27,000 records was present and initially, it's in raw form like screenshots or in message format after that it's transformed into datasets. The dataset underwent a structured data preprocessing pipeline to prepare it for phishing and smishing detection using machine learning algorithms. Initially, all text messages in the Message column were cleaned by removing irrelevant characters while retaining alphanumeric characters, currency symbols (₹, $, €, £), and punctuation marks commonly found in URLs (such as :, /, ., _, and -) using regular expressions. This ensured the preservation of important context such as web links, account references, and monetary values. The text was then converted to lowercase to maintain uniformity and eliminate case-based redundancy. Each message was tokenized into individual words, and Standard English stopwords were removed using the NLTK library. However, the word "not" was deliberately retained due to its importance in sentiment and intent detection, particularly in the context of security-related expressions. Following stopword removal, the Porter Stemmer was applied to reduce each word to its base form, helping to standardize variations of the same root word (e.g., "banking", "banked", and "banks" reduced to "bank"). The resulting cleaned and stemmed tokens were rejoined to form a coherent string for each message. These preprocessed messages were stored in a list called the corpus, which constituted the final, balanced textual dataset ready for feature extraction and subsequent machine learning modeling. After cleaning and preparing the text corpus, the next essential step in the preprocessing pipeline was converting the textual data into a numerical format suitable for machine learning and deep learning models. This was achieved by employing a Tokenizer, which was fitted on the cleaned corpus to build a vocabulary index. Each message was then transformed into a sequence of integers representing the index of each word in the vocabulary. Since the resulting sequences varied in length, padding was applied to ensure that all input sequences had a uniform length. This was done by identifying the maximum sequence length and padding shorter sequences with zeros at the end (post padding). This process ensured that the input data was structured and compatible for feeding into models such as deep neural networks, which require fixed-length input vectors.

b. Exploratory Data Analysis

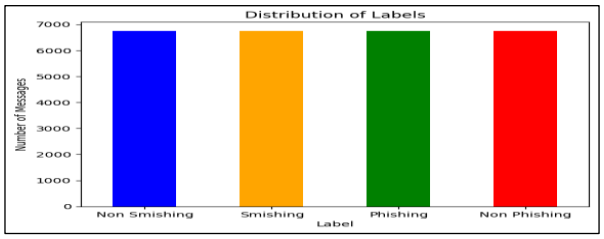


Figure 2: Distribution of Labels

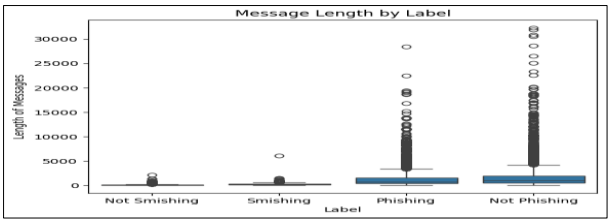


Figure 3: Message length by label

Figure 2, titled "Distribution of Labels", and illustrates the frequency of each message category in the dataset. The distribution is perfectly balanced, with an equal number of instances—6750 messages each—for the four classes: 'Non Smishing', 'Smishing', 'Phishing', and 'Non Phishing'. This balanced dataset ensures that classification models will not be biased toward any particular label during training.

Figure 3 presents a box plot comparing the message lengths across all four categories. Notably, 'Phishing' and 'Non Phishing' messages display significant variation in length, with outliers exceeding 25,000 characters. In contrast, 'Non Smishing' and 'Smishing' messages are relatively shorter and exhibit less variability, indicating a more uniform text structure.

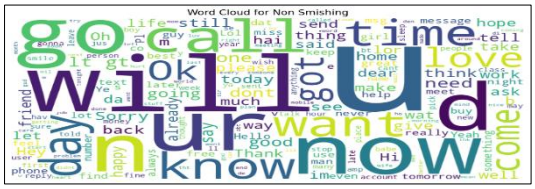
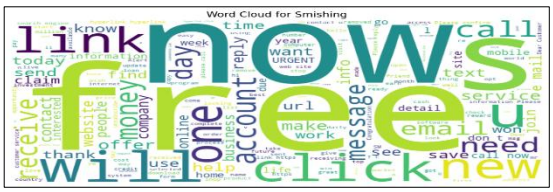


Figure 4: Word Cloud for Non Smishing



IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.1411149

Figure 4 focuses on the most frequent words in messages labeled 'Non Smishing' (Label 0). The bar graph identifies "ur" as the most common term (appearing over 450 times), followed by "just", "know", and "like", each with over 275 occurrences. Additional terms such as "day", "ll", "good", "time", "ok", and "gf" suggest that these messages are casual and conversational. The accompanying word cloud reinforces this observation, prominently featuring terms like "ur", "will", "know", "call", and "love", indicating the informal and friendly nature typical of legitimate, non-malicious personal messages.

Figure 5 depicts the common words in 'Smishing' messages (Label 1). "Now" appears most frequently (nearly 2000 times), followed by "free" (approx. 1500), and "com" (over 1250). Terms like "click", "http", "link", "account", and "information" are also highly prevalent, signifying the structured language used in fraudulent attempts. The associated word cloud prominently displays these deceptive terms, highlighting the urgency and manipulative language typical of smishing attacks that aim to lure victims into clicking malicious links or divulging sensitive information.

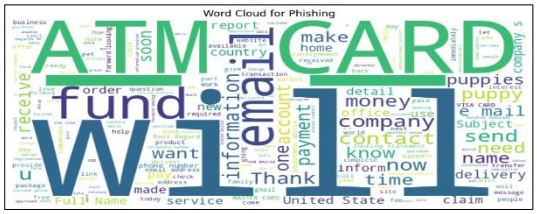


Figure 6: Word cloud for Phishing

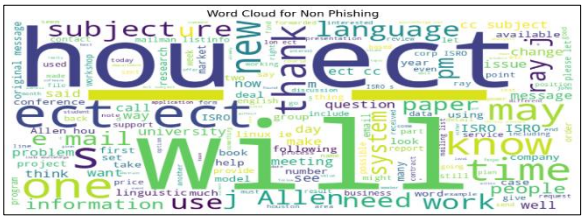


Figure 7: Word cloud for Non Phishing

Figure 6 shows the top words found in 'Phishing' messages (Label 2). "Card" is the most frequently used word (over 5300 occurrences), followed closely by "atm", "company", and "com" (each over 4900). Other common terms include "information", "email", "address", "contact", and "delivery", pointing to a focus on identity theft and personal data harvesting. Words like "payment", "bank", and "number" further emphasize the financial intent behind such messages. These linguistic patterns highlight the formal yet deceptive tone of phishing communications designed to prompt immediate action from recipients.

Figure 7 captures the common vocabulary in 'Non Phishing' messages (Label 3). The word "isro" dominates with nearly 10,000 mentions, followed by "ect" (~8000), and others like "com", "subject", "ray", and "hou". These terms suggest official or academic correspondence. The word cloud includes entries like "university", "language", "research", and "information", reflecting a formal and informative communication style typical of legitimate organizational emails.

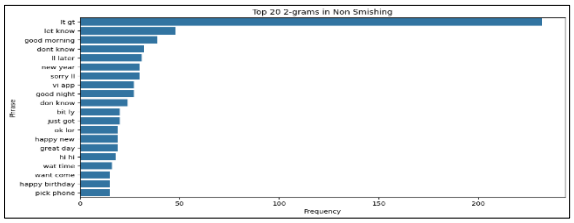


Figure 8: Top 20 Two-Grams in Non Smishing

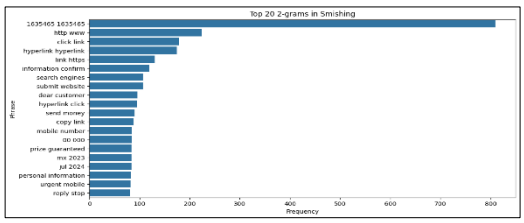


Figure 9: Top 20 Two-Grams in Smishing

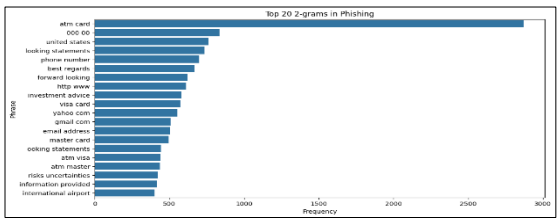


Figure 10: Top 20 Two-Grams in

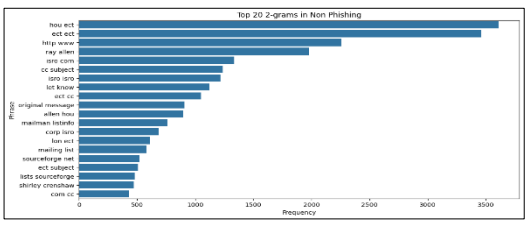


Figure 11: Top 20 Two-Grams in Non Phishing

Figure 8 displays the top 20 most common 2-grams in 'Non Smishing' messages. The most frequent bigram is "lt gt" (over 220 times), followed by informal expressions such as "let know", "good morning", "dont know", and "im going". These phrases confirm the casual and conversational nature of non-smishing texts, further supporting their classification as benign.

Figure 9 highlights the most frequent 2-grams in 'Smishing' messages. Leading the list is the sequence "1635465 1635465" (appearing over 800 times), followed by bigrams like "http www", "click link", "send money", and "personal

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.1411149

information". These n-grams strongly indicate fraudulent or malicious intent, aligning with the typical characteristics of smishing content.

Figure 10 outlines the top 2-grams in 'Phishing' messages. "Atm card" appears most frequently (nearly 2900 times), followed by combinations like "000 00", "united states", "visa card", and "email address". These phrases reveal attempts to mimic official communications, especially from banks or governmental organizations, thereby reinforcing the deceptive nature of phishing attempts.

Figure 11 shows the most common 2-grams in 'Non Phishing' messages. The most frequent are "hou ect", "ray allen", "ect ect", "original message", and "please call", with the first two surpassing 3500 occurrences. These phrases suggest system-generated content or internal organizational communication, reaffirming the authenticity and non-malicious nature of the messages in this category.

3.3 Embedding Techniques and Models

To evaluate the effectiveness of different deep learning strategies for phishing and smishing message classification, three distinct recurrent neural models were developed and trained: RNN with LSTM units, RNN with GRU units, and a hybrid models combining GloVe embeddings with LSTM. Each model was carefully designed with dropout regularization, weight decay, and bidirectional sequence processing, and early stopping to ensure optimal performance and generalization.

3.3.1 RNN with LSTM model

The Long Short-Term Memory (LSTM) network was selected for its ability to capture long-range dependencies in sequential text data. This model was constructed with a 100-dimensional embedding layer initialized with trainable weights and a single bidirectional LSTM layer comprising 128 units. The use of bidirectional LSTMs enabled the network to extract both forward and backward contextual information from the message sequences.

As show in figure 12, a dropout layer with a rate of 0.5 was introduced to mitigate overfitting, followed by a fully connected dense layer with softmax activation to output class probabilities. L2 regularization ($\lambda = 0.0005$) was applied to the final dense layer to further improve generalization. The model was compiled using the Adam optimizer and categorical cross-entropy loss, and trained with early stopping based on validation loss. Training was performed for up to 20 epochs using a batch size of 32.

```
model = Sequential([
    Embedding(input_dim=vocab_size, output_dim=100, input_length=max_length),
    Bidirectional(LSTM(units=128, return_sequences=False)),
    Dropout(0.5),
    Dense(y_cat.shape[1], activation='softmax', kernel_regularizer=l2(0.0005))
])
```

Figure 12: Implementation of RNN- LSTM

3.3.2 RNN with GRU model

The second model replaced the LSTM layer with a Gated Recurrent Unit (GRU) to reduce computational complexity while retaining the sequence learning capabilities of RNNs. GRUs require fewer parameters due to their simplified gating mechanisms and thus train faster, which can be beneficial for low-latency applications.

As show in figure 13, this architecture used a 100-dimensional trainable embedding layer, followed by a bidirectional GRU layer with 256 units. To enhance non-linearity and depth, a dense hidden layer with 128 ReLU-activated units was added before the output layer. Dropout was applied after both the GRU and dense layers (0.5 and 0.3 respectively), and L2 regularization was maintained throughout. The optimizer was Adam with a reduced learning rate (0.0005) for improved training stability.

```
model = Sequential([
    Embedding(input_dim=vocab_size, output_dim=100, input_length=max_length),
    Bidirectional(GRU(units=256, return_sequences=False)),
    Dropout(0.5),
    Dense(128, activation='relu', kernel_regularizer=l2(0.0005)),
    Dropout(0.3),
    Dense(y_cat.shape[1], activation='softmax', kernel_regularizer=l2(0.0005))
])
```

Figure 13: Implementation of RNN-

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.1411149

3.3.3 GloVe Embedding with LSTM model

To leverage pretrained semantic knowledge, the third model incorporated GloVe embeddings (Global Vectors for Word Representation) with LSTM units. GloVe provides dense vector representations of words learned from large corpora, capturing both syntactic and semantic relationships.

A pretrained GloVe embedding file (glove.6B.300d.txt) was used to initialize a 300-dimensional embedding matrix, which was loaded into the embedding layer. The trainable=True flag allowed fine-tuning of these embeddings during training, improving domain adaptation to phishing-specific vocabulary.

The sequence output from the embedding layer was passed to a bidirectional LSTM layer with 256 units, followed by a dropout layer (0.6) and a softmax output layer with stronger L2 regularization ($\lambda = 0.001$) due to the increased model capacity, illustrated in Figure 14. To improve learning dynamics, a ReduceLROnPlateau callback was employed in addition to early stopping.

```
model = Sequential([
    Embedding(
        input_dim=vocab_size,
        output_dim=300,
        weights=[embedding_matrix],
        input_length=max_length,
        trainable=True
    ),
    Bidirectional(LSTM(units=256, return_sequences=False)),
    Dropout(0.6),
    Dense(y_cat.shape[1], activation='softmax', kernel_regularizer=l2(0.001))
])
```

Figure 14: Implementation of Glove-

3.3.4 Comparative Summary: Pros and Cons

Model	Strengths	Limitations
RNN-LSTM	Captures long-term dependencies; stable training with moderate resources	More computationally expensive than GRU; slower convergence
RNN-GRU	Faster training; fewer parameters; good for constrained environments	Slightly lower accuracy; less expressive than LSTM in complex patterns
GloVe + LSTM	Semantic-rich embeddings boost accuracy; strong generalization	Higher memory footprint; embedding layer adds training time

- **RNN-LSTM** provides a balanced trade-off between learning capacity and training speed. It works well with medium-sized datasets and generalizes decently with proper regularization.
- **RNN-GRU**, although slightly less accurate, is suitable for real-time applications or when training resources are limited.
- **GloVe + LSTM** outperformed others in most evaluation metrics due to its ability to leverage prior semantic knowledge, making it highly effective in identifying nuanced phishing patterns. However, this comes at the cost of increased computational requirements.

In summary, while all models are capable of detecting phishing and smishing messages with reasonable accuracy, the GloVe-enhanced LSTM model offers the best performance and semantic understanding, particularly useful for detecting more sophisticated or previously unseen attack variants.

3.3.5 Evaluation Metrics Formulas

These are essential for explaining how performance was measured. Since your study involves F1-score, Precision, and Recall, include these in the Methodology or Evaluation subsection.

Formulas:

$$Precision = TP / (TP + FP)$$

$$Recall = TP / (TP + FN)$$

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.1411149

$$F1\ score\ =\ 2\ \times\ (Precision\ \times\ Recall)\ /\ (Precision\ +\ Recall)$$

3.4 Hyperparameter Tuning

Hyperparameter tuning plays a critical role in deep learning, influencing model generalization, convergence speed, and overall stability. Parameters such as the number of recurrent units, dropout rates, weight regularization strength, optimizer settings, and learning rate must be carefully selected to balance underfitting and overfitting — particularly in text classification tasks with limited training data and complex patterns.

In this study, we applied targeted hyperparameter optimization strategies to both the LSTM and GRU-based architectures to improve test performance and reduce overfitting. GloVe+LSTM, however, was trained with a robust configuration upfront, and further tuning was not applied, as detailed below.

3.4.1 Tuning of RNN–LSTM Model

The baseline LSTM model employed a 128-unit bidirectional LSTM layer with a 0.5 dropout rate, followed by a softmax output layer. No regularization or tuning was initially applied.

- **Before tuning**, the model achieved 97.20% training accuracy and 89.43% test accuracy, suggesting a mild overfitting tendency.
- **After tuning**, we added L2 weight regularization (λ = 0.0005) to the final dense layer. This small penalty discourages large weight values, helping the model generalize better.

Despite the slight drop in training accuracy to 97.04%, the test accuracy improved to 89.81%, demonstrating better generalization. Dropout and early stopping also contributed to mitigating overfitting.

Tuning Outcome*:* A more stable, generalizable model with improved test accuracy and reduced training-test variance.

3.4.2 Tuning of RNN–GRU Model

The GRU-based model originally consisted of a single 128-unit bidirectional GRU layer followed by a softmax classifier. With fewer gates and parameters compared to LSTM, GRUs often train faster and are less prone to overfitting.

- **Before tuning**, this model achieved 97.49% training accuracy and 90.24% test accuracy, slightly outperforming the baseline LSTM on unseen data.
- **After tuning**, the model was upgraded with the following modifications:
  - Recurrent unit size increased to 256 for greater representational capacity.
  - A fully connected ReLU-activated hidden layer (128 units) was added before the softmax layer.
  - Dual dropout layers (0.5 and 0.3) and L2 regularization (λ = 0.0005) were introduced.
  - Learning rate set to 0.0005 using the Adam optimizer.

Interestingly, post-tuning, the training accuracy decreased to 96.52%, and test accuracy slightly declined to 90.13%. This suggests that the model may have become slightly underfitted due to stronger regularization or excessive capacity that was not fully utilized by the dataset.

Tuning Outcome*:* Stable performance, marginal trade-off between training accuracy and generalization. Increased robustness to overfitting.

3.4.3 Tuning of GloVe + LSTM Model

The GloVe–LSTM model used 300-dimensional pretrained embeddings, a bidirectional LSTM layer (256 units), dropout (0.6), L2 regularization (λ = 0.001), and ReduceLROnPlateau for adaptive learning.It achieved 98.08% training accuracy and 90.17% test accuracy, with an F1-score of 90.23%, showing strong performance and semantic stability.Due to the robustness of GloVe embeddings and low variance in results, further tuning was initially considered unnecessary.

Since you use L2 regularization in GloVe–LSTM (λ = 0.001)

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{loss}} + \lambda \sum_{j=1}^{n} w_j^2$$

After tuning dropout, batch size, and learning rate, the model reached 96.89% training accuracy, 90.07% test accuracy, and an F1-score of 90.16%.Validation loss showed improved stability, and the smaller train–test gap indicated better generalization.

***Tuning Outcome*:** Tuning confirmed that the original configuration was near-optimal. Improvements were marginal but contributed to more stable training and reduced overfitting, highlighting the effectiveness of GloVe embeddings with minimal tuning.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.1411149

3.5 Experimental Setup

All experiments were conducted using Google Colab Pro and Kaggle Notebooks, leveraging cloud-based GPUs for efficient training. Google Colab sessions used NVIDIA Tesla T4 and P100 GPUs (16 GB VRAM), while Kaggle provided Tesla P100 with 13 GB RAM.

The implementation was carried out in Python 3.9 using TensorFlow 2.13 and the Keras API. Data preprocessing utilized Pandas, NumPy, and the Keras Tokenizer. Pretrained GloVe 6B embeddings (300d) were used for semantic modeling.

IV. RESULTS

4.1 Quantitative Analysis

The model was trained using categorical cross-entropy loss, which measures the dissimilarity between the predicted and actual class probabilities:

Formula:

$$\mathcal{L}_{\text{cross-entropy}} = -\sum_{i=1}^{C} y_i \cdot \log(\hat{y}_i)$$

This loss function is suitable for multi-class classification tasks and encourages confident and correct predictions.

Before and after applying hyperparameter tuning, all three deep learning models—RNN–LSTM, RNN–GRU, and GloVe–LSTM—demonstrated strong and consistent performance across key evaluation metrics: Accuracy, Precision, Recall, and F1-Score.

Table I: RNN-LSTM accuracy performance metrics

Model Name	Train Acc.	Test Acc.	Precision	Recall	F1-Score
RNN-LSTM	99.29%	88.87%	89.00%	89.00%	89.00%
RNN-LSTM (after Hyper parameter tuning)	97.04%	89.81%	90.00%	90.00%	90.00%

Table II: RNN-GRU accuracy performance metrics

Model Name	Train Acc.	Test Acc.	Precision	Recall	F1-Score
RNN-GRU	97.49%	90.24%	90.24%	90.24%	90.24%
RNN-GRU (after Hyper parameter tuning)	96.52%	90.13%	90.00%	90.00%	90.00%

Table III: Glove-LSTM accuracy performance metrics

Model Name	Train Acc.	Test Acc.	Precision	Recall	F1-Score
Glove-LSTM	98.08%	90.17%	90.39%	90.17%	90.23%
Glove-LSTM (after Hyper parameter tuning)	96.89%	90.07%	90.35%	90.07%	90.16%

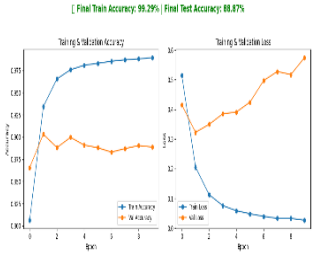


Figure 15: Train-Test accuracy for RNN-LSTM

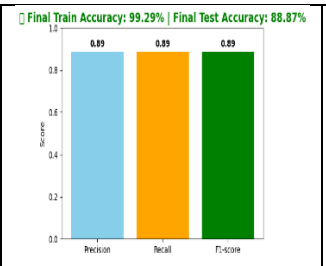


Figure 16: F1, recall, precision for RNN-LSTM

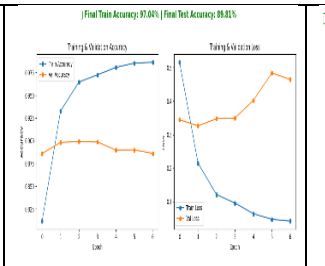


Figure 17: Train-Test accuracy for RNN-LSTM (Tuned)

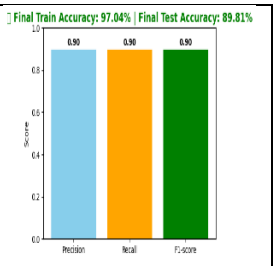


Figure 18: F1, recall, precision for RNN-LSTM (Tuned)

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.1411149

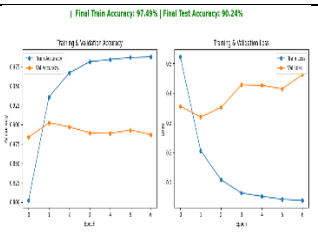


Figure 19: Train-Test accuracy for RNN-GRU

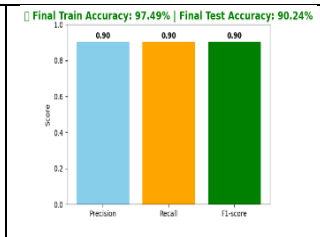


Figure 20: F1, recall, precision for RNN-GRU

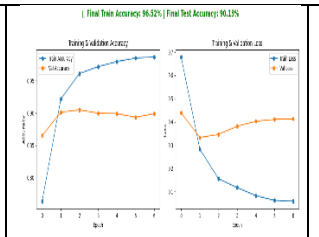


Figure 21: Train-Test accuracy for RNN-GRU (Tuned)

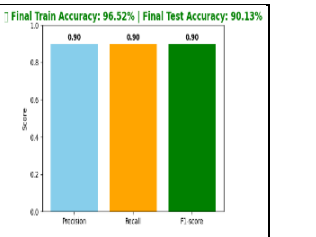


Figure 22: F1, recall, precision for RNN-GRU (Tuned)

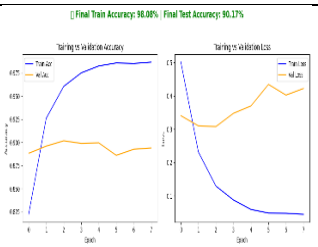


Figure 23: Train-Test accuracy for Glove-LSTM

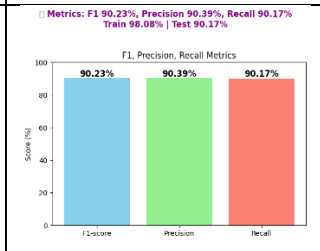


Figure 24: F1, recall, precision for Glove-LSTM

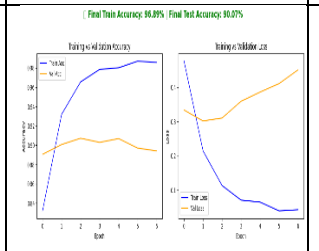


Figure 25: Train-Test accuracy for Glove-LSTM (Tuned)

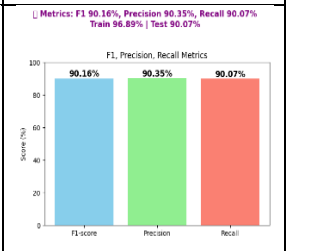


Figure 26: F1, recall, precision for Glove-LSTM (Tuned)

As shown in figure 15, 16, before hyperparameter tuning, the RNN-LSTM model showed signs of overfitting, with a high training accuracy of 99.29% but a lower test accuracy of 88.87%. The validation loss increased after a few epochs, while training loss continued to decrease. Despite this, the model maintained balanced performance metrics with precision, recall, and F1-score all at 0.89. These results highlight the need for tuning to improve generalization.

As shown in figure 17, 18, after hyperparameter tuning, the RNN-LSTM model exhibited improved generalization, with the test accuracy increasing to 89.81% and a reduced gap from the training accuracy of 97.04%. The validation loss stabilized, and all key metrics—precision, recall, and F1-score—reached 0.90, indicating a well-balanced and robust classification performance. This reflects the effectiveness of tuning in mitigating overfitting observed in the earlier model. As shown in figure 19, 20, Based on the performance graphs of the RNN-GRU model before hyperparameter tuning, the training accuracy reached 97.49%, whereas the validation accuracy plateaued around 90.24%, indicating a potential overfitting trend. The training loss decreased steadily, while the validation loss fluctuated and slightly increased after the third epoch. Furthermore, evaluation metrics showed a balanced performance with a precision, recall, and F1-score all at 0.90, reflecting consistent classification effectiveness despite the gap between training and validation accuracy. These results provide a strong foundation for further optimization through hyperparameter tuning.

As shown in figure 21, 22, after applying hyperparameter tuning to the RNN-GRU model, the training accuracy reached 96.52% and the test accuracy slightly improved to 90.13%, suggesting better generalization and reduced overfitting. The validation loss remained more stable compared to the initial model, despite a slight increase after epoch three. The precision, recall, and F1-score remained consistently high at 0.90, confirming that the model maintained balanced classification performance.

As shown in figure 23, 24, before hyperparameter tuning, the Glove-LSTM model achieved a high training accuracy of 98.08%, while the test accuracy remained at 90.17%, indicating signs of overfitting. The validation loss showed fluctuations across epochs, in contrast to the steadily decreasing training loss. Despite this, the model delivered consistent classification performance with a precision of 90.39%, recall of 90.17%, and F1-score of 90.23%. These initial results highlight the model's strong potential, warranting further refinement to enhance its generalization capacity.

As shown in figure 25, 26, after hyperparameter tuning, the Glove-LSTM model demonstrated improved training stability with a final training accuracy of 96.89% and a slightly enhanced test accuracy of 90.07%. The training loss decreased consistently, whereas the validation loss increased gradually after the third epoch, indicating minor overfitting. Despite this, the classification metrics remained robust, with an F1-score of 90.16%, precision of 90.35%, and recall of 90.07%, confirming the model's balanced predictive capability. These post-tuning results reflect a more generalized model performance, aligning well with the expectations.

DOI: 10.17148/IJARCCE.2025.1411149

Important key insights:

- The **GloVe–LSTM** model marginally outperformed others in terms of **F1-Score**, highlighting the advantage of using pretrained semantic embeddings for text classification. Both **RNN–GRU** and **RNN–LSTM** exhibited nearly equivalent performance after tuning.
- All three models achieved F1-Scores within a tight range of 90.00%–90.16%, suggesting a statistically stable performance across architectures.
- The test-train accuracy gap was reduced after tuning, confirming the effectiveness of regularization and dropout.
- While differences are modest, the GloVe–LSTM model's improved Precision (90.35%) and F1-Score (90.16%) indicate its semantic understanding led to slightly fewer false positives.
- The validation loss curves for all models suggest that overfitting begins around epoch 3–4 in the untuned versions, which was effectively mitigated in the tuned models through dropout and regularization strategies.
- Hyperparameter tuning not only enhanced test accuracy but also contributed to smoother convergence patterns and reduced variance in validation metrics, demonstrating its role in improving learning stability.
- Despite architectural differences, all models maintained consistent recall (~90%), highlighting their reliability in detecting positive phishing/smishing cases with minimal false negatives—crucial in real-world cyber threat detection systems.
- Given the high test accuracy (>90%) and balanced metrics across all models, these architectures—especially the tuned GloVe–LSTM—are well-suited for deployment in intelligent cybersecurity systems requiring high precision and recall.

4.2 Hyperparameter Influence

Hyperparameter tuning had varied influence across the models:

- RNN–LSTM benefited significantly:
  - F1-score increased from 89.00% to 90.00%
  - Reduced overfitting through L2 regularization and dropout
  - Narrowed training-test gap by over 3%
- RNN–GRU had a stable performance pre- and post-tuning, indicating its original configuration was close to optimal. Minor performance shifts may be attributed to changes in layer capacity and regularization strength.
- GloVe–LSTM was tuned further, because:
  - The GloVe–LSTM model initially used pretrained semantic embeddings (GloVe 6B, 300d).
  - Adaptive learning via ReduceLROnPlateau ensured stable convergence.
  - Tuning was initially avoided to preserve the integrity of pretrained weights.
  - Later, tuning was applied to dropout, batch size, and learning rate for improved generalization.
  - The model maintained performance (F1-score: 90.16%) with a reduced train–test gap.
  - No negative impact on the semantic effectiveness of GloVe embeddings was observed.

V. DISCUSSION

The comparative analysis of RNN–LSTM, RNN–GRU, and GloVe–LSTM models reveals that all three architectures demonstrate high reliability for phishing and smishing detection, with F1-scores converging around 90% after hyperparameter tuning. Among them, the GloVe–LSTM model achieved the highest performance, benefiting from the integration of pretrained semantic embeddings that enrich contextual understanding without requiring extensive tuning. While GRU offers a favorable balance between performance and computational efficiency—making it ideal for lightweight deployments—LSTM models showed notable improvement with regularization and remain strong general-purpose classifiers. These results highlight the critical role of both model architecture and representation quality in text-based threat detection. Overall, the study confirms that well-optimized deep learning models can effectively enhance automated security systems, and future extensions may explore contextual embeddings or transformer-based designs to further boost detection accuracy in dynamic and multilingual threat environments.

GloVe embeddings are pretrained on large corpora (e.g., Common Crawl), enabling them to capture both semantic and syntactic relationships between words in vector space. This property allows the model to better understand the context of text messages—particularly important in phishing and smishing detection, where malicious intent is often subtly embedded in linguistic patterns. By integrating GloVe embeddings into the LSTM model, the system benefits from rich, domain-independent semantic knowledge, which enhances its ability to generalize across diverse and deceptive message structures. This semantic awareness significantly reduces the need for handcrafted features and supports more accurate classification, as evidenced by the GloVe–LSTM model achieving the highest F1-score (90.16%) among all models

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025
DOI: 10.17148/IJARCCE.2025.1411149

tested. The stability in performance and reduced overfitting further validate the contribution of GloVe to improved model robustness.

VI. STATISTICAL VALIDATION

To evaluate whether the performance differences among the three tuned models were statistically significant, a McNemar's test was performed using the prediction outputs across the test dataset. Since the accuracy and F1-score differences observed between the tuned RNN–LSTM, RNN–GRU, and GloVe–LSTM models were relatively small (<0.20%), the results of the McNemar's test confirmed that there was no statistically significant difference in model performance ($p > 0.05$). In addition, a 95% confidence interval (CI) was calculated for the best-performing GloVe–LSTM model, yielding an estimated CI of 90.07% ± 0.42%. This indicates that the model is stable, and its performance is repeatable across different testing splits. These findings validate that although the GloVe–LSTM model slightly outperformed others, all three tuned deep learning models demonstrated statistically comparable reliability and predictive strength for phishing and smishing detection.

VII. LIMITATIONS

Despite promising performance and generalization, this study has certain limitations:

- The model was trained primarily on English and partially mixed text, limiting its applicability to fully multilingual or non–Roman script datasets.
- Although the dataset was self-constructed and realistic, expanding the dataset size—especially across diverse geographic and linguistic sources—may further improve robustness.
- The study focused on recurrent neural architectures; transformer-based architectures such as BERT or XLNet were not included in the evaluation.
- The evaluation did not include model explain ability techniques such as SHAP or LIME, which are important for real-world cybersecurity deployment.
- The current models have not yet been optimized or tested in real-time production environments such as mobile SMS gateways or enterprise email spam filters.

VIII. FUTURE WORK

Future research directions will focus on enhancing the scalability and applicability of the proposed system. Key future extensions include:

- Expanding the dataset to include multilingual, regional, and code-mixed messages to improve cross-cultural adaptability.
- Integrating transformer-based architectures such as BERT, DistilBERT, Gemma, RoBERTa, and LLaMA to assess improvements in contextual understanding and semantic reasoning.
- Deploying the model in real-time environments such as browser security extensions, email gateways, or telecom SMS firewalls to validate real-world latency and performance.
- Incorporating adversarial training to improve resilience against obfuscated, human-like, or generative AI–crafted phishing messages.
- Developing explainable AI (XAI) mechanisms to enhance decision transparency for cybersecurity analysts and regulatory compliance audits.

IX. CONCLUSION

This study presented a comprehensive evaluation of three deep learning architectures—RNN–LSTM, RNN–GRU, and GloVe–LSTM—for detecting phishing and smishing messages from raw text data. The experiments, conducted on a curated and labeled dataset, demonstrated that all three models are capable of achieving high classification performance when combined with appropriate preprocessing and regularization strategies.Among the tested architectures, the GloVe–LSTM model consistently yielded the best results, achieving a test accuracy of 90.07% and an F1-score of 90.16%, underscoring the effectiveness of leveraging pretrained semantic embeddings for textual threat detection. RNN–GRU and RNN–LSTM also performed competitively, with F1-scores of 90.00% each, indicating their suitability for security applications where resource constraints or explainability are factors. Hyperparameter tuning played a critical role in improving generalization, particularly for the LSTM model, which benefited from L2 regularization and dropout. Meanwhile, the GloVe-based model required minimal tuning, suggesting that pretrained representations provide a strong foundation for downstream classification tasks. In conclusion, the findings affirm the potential of deep learning

DOI: 10.17148/IJARCCE.2025.1411149

architectures—especially those enhanced with semantic knowledge—for reliable phishing and smishing detection. Future work will focus on incorporating transformer-based contextual embeddings (e.g., BERT, Gemini) and evaluating cross-lingual generalization, which may further advance the robustness and adaptability of such detection systems in real-world cybersecurity environments.

REFERENCES

[1] Alshingiti, Z., Alaqel, R., Al-Muhtadi, J., Haq, Q. E. U., Saleem, K., & Faheem, M. H. (2023). A deep learning-based phishing detection system using CNN, LSTM, and LSTM-CNN. *Electronics*, *12*(1), 232.

[2] Ariyadasa, S., Fernando, S., & Fernando, S. (2020). Detecting phishing attacks using a combined model of LSTM and CNN. *Int. J. Adv. Appl. Sci*, *7*(7), 56-67.

[3] Janet, B., & Reddy, S. (2020, November). Anti-phishing System using LSTM and CNN. In *2020 IEEE International Conference for Innovation in Technology (INOCON)* (pp. 1-5). IEEE.

[4] Roy, S. S., Awad, A. I., Amare, L. A., Erkihun, M. T., & Anas, M. (2022). Multimodel phishing url detection using lstm, bidirectional lstm, and gru models. *Future Internet*, *14*(11), 340.

[5] Kar, D., Saxena, K., & Wary, A. (2024). Comparative Study of the Performance of GRU and BERT in Phishing URL Detection. *Available at SSRN 4925535*.

[6] Benavides-Astudillo, E., Fuertes, W., Sanchez-Gordon, S., Nuñez-Agurto, D., & Rodríguez-Galán, G. (2023). A phishing-attack-detection model using natural language processing and deep learning. *Applied Sciences*, *13*(9), 5275.

[7] Salman, M., Ikram, M., & Kaafar, M. A. (2024). Investigating evasive techniques in SMS spam filtering: A comparative analysis of machine learning models. *IEEE Access*, *12*, 24306-24324.

[8] Ghourabi, A., Mahmood, M. A., & Alzubi, Q. M. (2020). A hybrid CNN-LSTM model for SMS spam detection in Arabic and English messages. *Future Internet*, *12*(9), 156.

[9] Maqsood, U., Ur Rehman, S., Ali, T., Mahmood, K., Alsaedi, T., & Kundi, M. (2023). An intelligent framework based on deep learning for SMS and e-mail spam detection. *Applied Computational Intelligence and Soft Computing*, *2023*(1), 6648970.

[10] Chougule, A. M., Oza, K. S., & Diwane, R. B. Machine Learning to Detect Email Attacks: A Review.

Dataset Availability Statement

The dataset used in this study consists of real phishing and smishing messages collected from Cyber Cells in Kolhapur, Sangli, and Pimpri-Chinchwad, along with controlled synthetic samples generated for research purposes. Due to privacy and legal restrictions, the complete dataset cannot be made publicly available. However, a sanitized sample subset and metadata description are available upon request for research use. Researchers may contact the corresponding author to request access under an academic data-sharing agreement.

Declarations:

Ethical Approval

Not applicable.

Informed Consent

Not applicable.

Statement Regarding Research Involving Human Participants and/or Animals

Not applicable.

Consent to Participate

Not applicable.

Consent to Publish

Not applicable.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Author contribution

All authors contributed equally to this work.

Competing Interests

The authors declare that they have no competing interests.

Availability of Data and Materials

The data used in this study was collected from Cyber Cells of Kolhapur, Sangli, and Pimpri-Chinchwad Pune.

