iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** IV **Month of publication:** April 2023

DOI: https://doi.org/10.22214/ijraset.2023.50099

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue IV Apr 2023- Available at www.ijraset.com

Openmenot: A Google Chrome Extension for Detecting Phishing Websites Using Deep Learning

Dr. V. Govindhasamy[1], Ms. Nivethitha. A. P[2], Ms. Pandeeswari. K[3], Ms. Aswathi[4], Mr. Piravin[5]
[1]*Head of the Department,* [2, 3, 4, 5]*B.Tech Student, Information Technology, Puducherry Technological, University Puducherry, India*

Abstract: ***With the advent of internet as a major medium right from inter-personal communication to how business transactions occur, the need to focus on research and development of various cyber security techniques rises. Phishing is one of the most common social engineering methods faced by public internet users, governments and businesses. There are several types of phishing by which users are directed to fake websites that resemble the legit ones. These fake websites collect the user credentials and use them for illegal purposes. In this project, a three-fold detection model using deep learning techniques like (Gradient Booster) for lexical and domain feature extraction ,(LSTM) for hyperlink extraction and (CNN) for content based extraction to detect phishing websites is proposed. This model is deployed by developing a google chrome extension that classifies and alerts the user when he/she is directed to a website that is not legit.***

Index Terms: Gradient Booster, LSTM, CNN, lexical and domain feature extraction, hyperlink extraction, content based extraction, google chrome extension,phishing detection

I. INTRODUCTION

Phishing nowadays is one of the most serious and dangerous online threat in the domain of cybersecurity. The use of social networks, e-commerce, electronic banking, and other online services has been increased immensely due to the rapid development of internet technologies. At present, internet penetration stands at 59.5 percent which provides an opportunity to make money for a phishing attacker by blackmailing and stealing confidential information from internet users. The attacker develops a fraudulent website and sends links to online platforms like Facebook, Twitter, emails, etc by conveying a message of panic, urgency, or a financial bid, and instructs the recipient to take immediate action. When a user unwittingly clicks the link and updates any clicks the link and updates any sensitive credentials, cyber attackers gain access to the user's information like financial data, personal information, username, password, etc. This stolen information is used by cybercriminals for a variety of illegal activities, including blackmailing victims. Many researchers have recently developed various machine learning and deep learning models to detect phishing sites. However, earlier systems have limitations such as blacklisting cannot predict temporary or dynamic urls,Easily evaded and restructured using html and cloned like a legitimate website. In this paper, we propose a solution for the detection of phishing detection using lexical,hyperlink,content-based extraction. Our proposed system utilizes google Chrome Extension named Openmenot which alerts the user when a phishing url is detected.

This paper aims to discuss the limitations and issues of earlier url phishing detection systems, present our proposed system, and evaluate its performance using different metrics such as accuracy rate, precision, recall, and F1 score.

II. LITERATURE SURVEY

[1] Ala Mughaid a, Shadi AlZubi, Salah Tamanna researched using machine learning to propose a detection model by splitting the dataset to train the detection model and validating the results using the test data , to capture inherent characteristics of the email text, and other features to be classified as phishing or non-phishing using three different data sets, After making a comparison between them, we obtained that the most number of features used the most accurate and efficient results achieved. the best ML algorithm accuracy were 0.88, 1.00, and 0.97 consecutively for boosted decision tree.

[2] Das guptta. S, Shahirar K.T, Alqatani H proposed a hybrid feature based anti-phishing strategy that extracts features from URL and hyperlink information of client-side only. They also developed a new dataset for the purpose of conducting experiments using popular machine learning classification techniques. Their experimental result shows that the proposed phishing detection approach is more effective having higher detection accuracy of 99.17% with the XG Boost technique than traditional approaches.

[3] Aswathi A, Goel conducted study in 3 phases. First, they included classification using base classifiers, Ensemble classifiers, and then ensemble classifiers are tested with and without cross-validation. Finally, their performance is analyzed, and the results are presented at last to help others use this study for their upcoming research.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue IV Apr 2023- Available at www.ijraset.com

III. LIMITATIONS IN THE EXISTING

System

1) *URL'S Are Temporary Or Dynamic:* In black-listing the phishing websites are stored in db after analyzing method. Blacklist method result in failure as most URLs and temporary and dynamic.
2) *Tedious Work For End-User:* In white-listing,the end-users are requested to check the domain and IP address each time they enter. It cant be applicable always
3) *Manipulated By Programmers:* In content-based approach, the contents are easily evaded by restructuring html elements without changing the appearance of the site.
4) *Easily Duplicated:* In visuality similarity based approach,the real websites will be cloned ,so that the difference between phishing and legimitate websites cant be found easily.

IV. PROPOSED SYSTEM

Our proposed system for url phishing detection with google chrome extension utilizes Gradient Booster,LSTM and CNN model features to build our deep learning model that leverages multimodel fusion techniques to improve detection accuracy.

A. Input

The first step will be collecting a dataset of phishing urls from various sites which are labelled as legitimate or phishing.

B. Processing

We proposed a unique three-fold method for detection of phishing websites. The three levels of classification and detection are as follows:

1) Binary Classification using Gradient- classifier based on Lexical and domain features extracted from URL
2) Binary Classification on the negatives from level 1 using Long-Short Term Memory classifier based on hyperlink features extracted from website crawling Classification using Convolutional Neural Networks based on the extracted web content of the web page linked with the URL

At each level, the evaluation metrics of the algorithm is calculated by using labeled data. This detection model is aimed to be implemented by developing a Google Chrome Extension that will pass every URL that the user is trying to access. If the URL is detected to be malicious, then the user gets a warning alerting him/her.

C. Output

The Google chrome extension detects phishing sites and gives alert to the user. The performance of the model will be evaluated using standard metrics such as accuracy rate, precision,recall, and F1 score.

V. DESIGN DIAGRAM OF THE PROPOSED SYSTEM

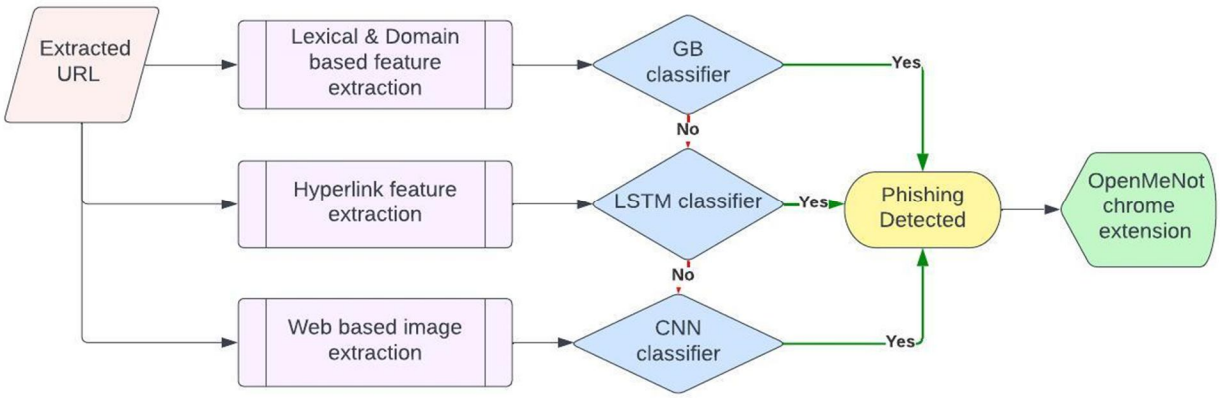


Fig1:Design diagram of the proposed system

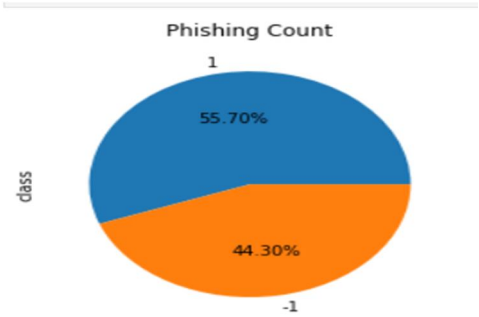


Fig 2:Phishing count using pie-chart

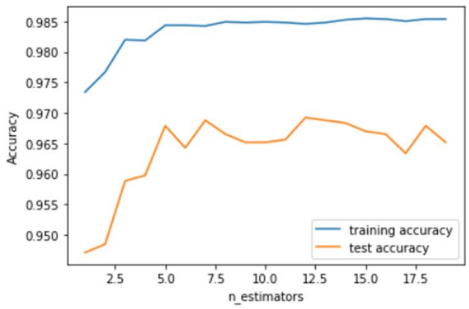


Fig 3:Training and testing in Gradient booster

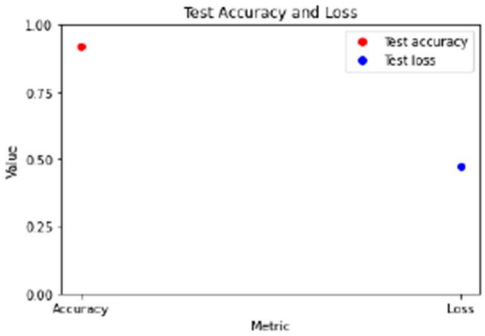


Fig 4:Testing accuracy and loss in gradient booster

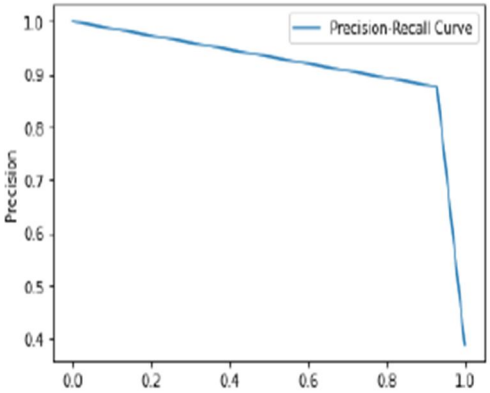


Fig 5:Precision-recall curve in gradient booster

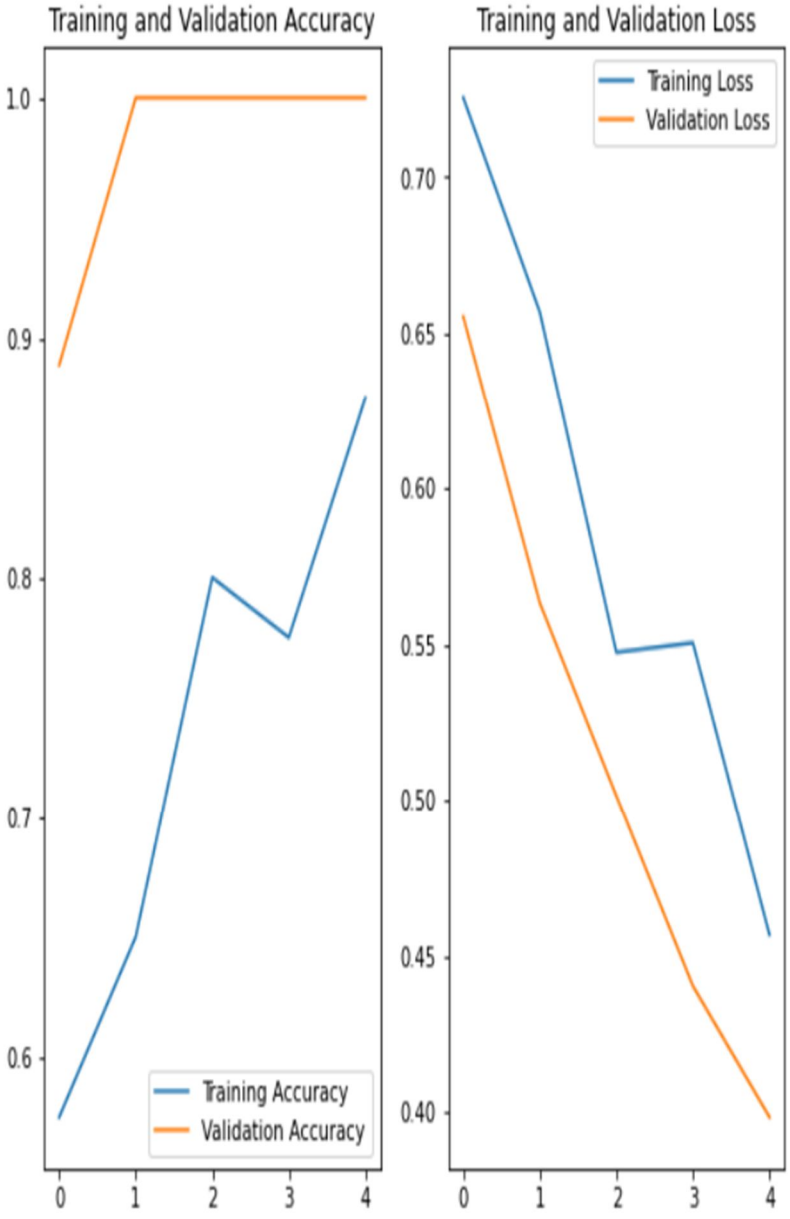


Fig 6:Training and validation accuracy,Training and validation Loss

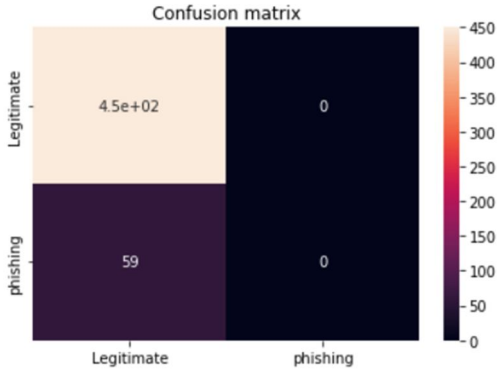


Fig 7:Overall Confusion Matrix of phishing and legitimate websites

VI. RESULT ANALYSIS

```
[ ] # function to extract hyperlinks from a given text
    def extract_hyperlinks(text):
        return re.findall("(?P<url>https?://[^\s]+)", text)
```

Fig 8:Extracting hyperlink using regex expression

```
# predict the phishing probability for each hyperlink using the LSTM model
phishing_probs = model.predict(X)
print(phishing_probs)
```

```
8/8 [==============================] - 18s 2s/step
[[0.02385678 0.086992   0.00970003 ... 0.01836065 0.01268108 0.01124205]
 [0.02384827 0.08707901 0.00965053 ... 0.01814    0.0125643  0.01116271]
 [0.0238202  0.08537435 0.00988228 ... 0.01825592 0.01288119 0.0115296 ]
 ...
 [0.02477172 0.08074263 0.01073053 ... 0.01882392 0.01327892 0.01233238]
 [0.02469291 0.08256759 0.01040801 ... 0.01850773 0.01289753 0.01189923]
 [0.02449393 0.08355957 0.01023662 ... 0.01840604 0.01280387 0.01174933]]
```

Fig 9:Predicting Phishing probability using LSTM model

```
[ ] # print the phishing probabilities and the corresponding hyperlinks
    print("Hyperlinks and their phishing probabilities:")
    for i in range(len(hyperlinks)):
        print(hyperlinks[i] + ' : ' + str(phishing_probs[i]))
```

```
Hyperlinks and their phishing probabilities:
https://www.iana.org/domains/example">More : [0.02385678 0.086992   0.00970003 0.02619178 0.04362282 0.00809447
 0.0401058  0.00633827 0.02995243 0.00833621 0.00675875 0.05093301
 0.00847787 0.01333284 0.02667139 0.0506923  0.02368897 0.01065412
 0.02429094 0.07387248 0.00706022 0.02392138 0.02812679 0.05056726
 0.06524196 0.0375847  0.04421218 0.03284357 0.04589787 0.03797008
 0.01172697 0.01836065 0.01268108 0.01124205]
```

Fig 10:Phishing Probabilities of the hyperlinks

```
import matplotlib.pyplot as plt

def show_batch(image_batch, label_batch):
  plt.figure(figsize=(25,20))
  for n in range(8):
      ax = plt.subplot(1,8,n+1)
      plt.imshow(image_batch[n])
      #plt.title(CLASS_NAMES[label_batch[n]==1][0].title())
      plt.axis('off')

image_batch, label_batch = next(train_gen)
show_batch(image_batch, label_batch)
```

Fig 11:Passing images of urls extracted after LSTM process

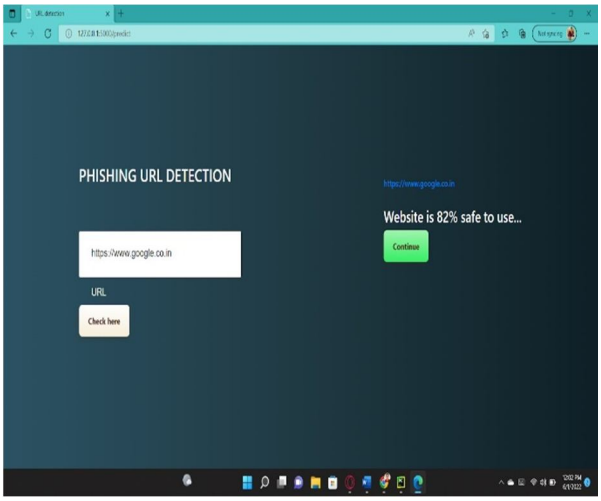


Fig 12:Legitimate URL sample output

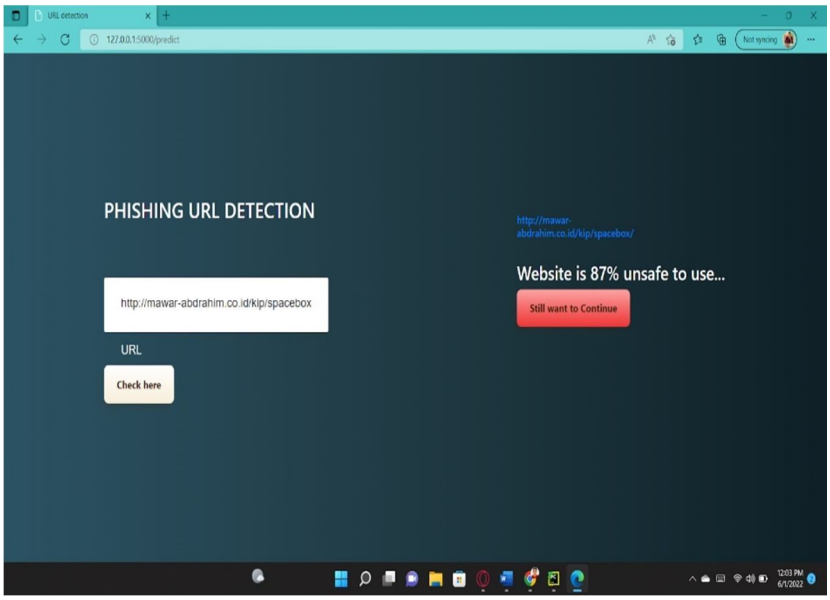


Fig 13:Phishing URL Sample Output

```
[24] ### TESTING

    image = tf.keras.preprocessing.image.load_img("/content/drive/MyDrive/Datasets/Data/Test/Legitimate/66600049.png",target_size= (256,256,3))
    input_arr = tf.keras.preprocessing.image.img_to_array(image)
    input_arr = np.array([input_arr])  # Convert single image to a batch.
    predictions = model.predict(input_arr)
    y_pred =np.argmax(predictions ,axis=1)
    if y_pred =='0':
      print('Legitimate')
    else:
      print('phishing')

    Legitimate
    /usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:8: FutureWarning: elementwise comparison failed; returning scalar instead, but ir
```

Fig 14: Legitimate Content Sample Output

```
### TESTING

image = tf.keras.preprocessing.image.load_img("/content/drive/MyDrive/Datasets/Data/Test/Phishing/40401533.png",target_size= (256,256,3))
input_arr = tf.keras.preprocessing.image.img_to_array(image)
input_arr = np.array([input_arr])  # Convert single image to a batch.
predictions = model.predict(input_arr)
y_pred =np.argmax(predictions ,axis=1)
if y_pred =='0':
  print('Legitimate')
else:
  print('phishing')
```

```
1/1 [==============================] - 0s 405ms/step
phishing
<ipython-input-20-4995fdfd79a6>:8: FutureWarning: elementwise comparison failed; returning scalar instead, but in the future will perform elementwise comparison
  if y_pred =='0':
```

Fig 15:Phishing Content Sample Output

```
from sklearn.metrics import confusion_matrix
import seaborn as sns
CM = confusion_matrix(y_values, y_pred)
ax = plt.axes()
class_names = ['Legitimate','phishing']
sns.heatmap(CM, annot=True,
            annot_kws={"size": 10},
            xticklabels=class_names,
            yticklabels=class_names, ax = ax)
ax.set_title('Confusion matrix')
plt.show()
```

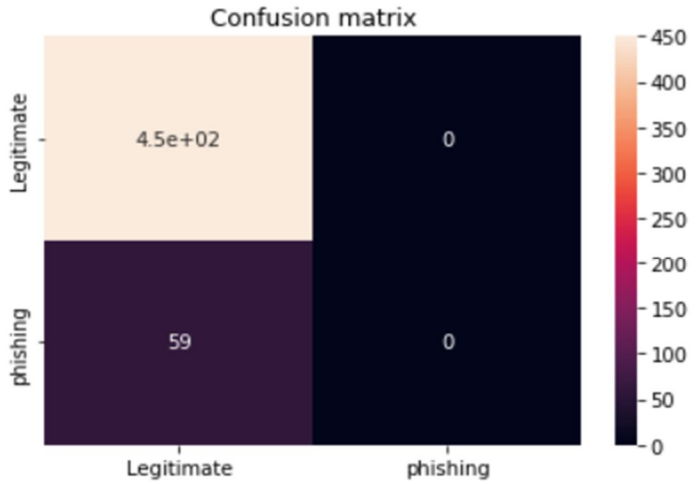


Fig 16:Overall Confusion Matrix

VII. CONCLUSION

In conclusion, URL phishing detection using deep learning has shown promising results in recent studies. Deep learning models, such as Gradient Booster,convolutional neural networks (CNNs) ,Long Short-term memory (LSTM), have been used to extract features from URLs and classify them as either legitimate or phishing. These models have been able to achieve high accuracy rates comparing to existing methods.

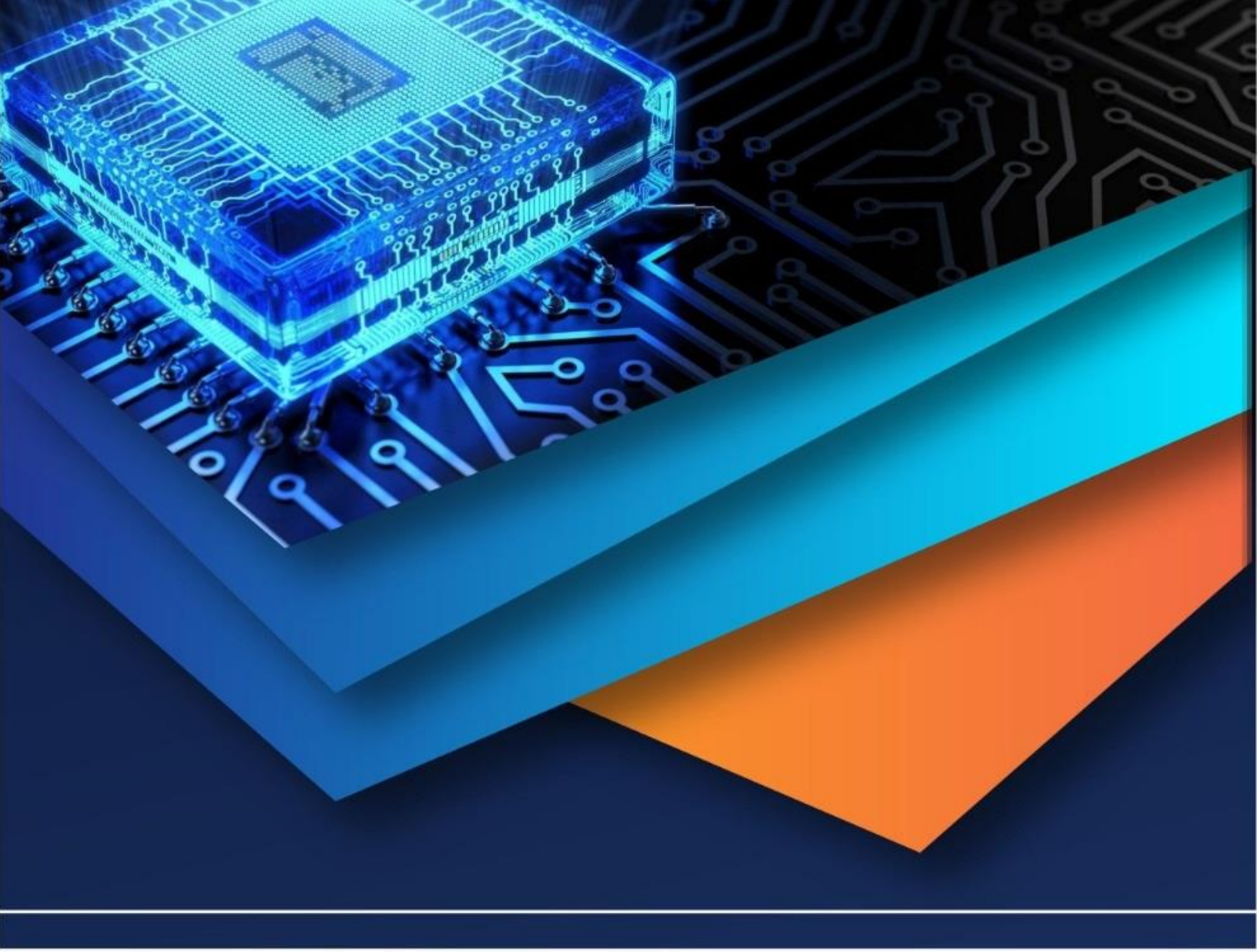
1) After Training and testing of model using Gradient Booster obtain the accuracy of 0.981.
2) After Training and testing of model using LSTM obtain the accuracy of 0.969.
3) After Training and testing of model using CNN model obtain the accuracy of 0.973.

VIII. ACKNOWLEDGEMENT

We are deeply indebted to Dr.V.Govindhasamy, Head of the Department, Department of Information Technology, Puducherry Technological University, Puducherry, India.

REFERENCES

[1] Erzhou Zhu,Yugang chen."An Effective Phishing Websites Detection Model Based On Optimal Feature Selection and Neural Network. IEEE ACCESS,7,2019
[2] Elmahgiubi, M., Ennajar, M., Drawil, N., & Elbuni, M. S. (2015). "Sign language translator and gesture recognition". 201Global Summit on Computer & Information Technology (GSCIT). doi:10.1109/gscit.2015.7353332
[3] Estrada Jiménez, L. A., Benalcázar, M. E., & Sotomayor, N. (2017). "Gesture Recognition and Machine Learning Applied to Sign Language Translation". IFMBE Proceedings, 233–236. doi:10.1007/978-981-10-4086-3_59
[4] Lean Karlo S. Tolentino, Ronnie O. Serfa Juan, August C. Thio-ac, Maria Abigail B. Pamahoy, Joni Rose R. Forteza,"Sign Language Recognition System" and International Journal of Machine Learning and Computing, Vol. 9, No. 6, December 2019
[5] Mohammad Elham Walizad, Mehreen Hurroo "Sign Language Recognition System using Convolutional Neural Network and Computer Vision"International Journal of Engineering Research & Technology (IJERT)-2020
[6] R.S. Sabeenian, S. Sai Bharathwaj, M. Mohamed Aadhil, "Sign Language Recognition Using Deep Learning and Computer Vision" Journal of Advanced research in Dynamical and Control System(Elsevier)-2020
[7] Bantupalli, Kshitij and Xie, Ying, "American Sign Language Recognition Using Machine Learning and Computer Vision" (2019)
[8] Kshitij Bantupalli, Ying Xie, "American Sign Language Recognition using Deep Learning and Computer Vision"IEEE International Conference on Big Data (Big Data)- 2020



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY