Kammavari Sangham (R) 1952, K.S.Group of Institutions

K. S INSTITUTE OF TECHNOLOGY

No.14, Raghuvanahalli, Kanakapura Road, Bengaluru - 560109
Affiliated to VTU, Belagavi & Approved by AICTE, New Delhi, Accredited by NBA , NAAC & IEI

Department of Computer Science and Engineering
Project Phase – II (18CSP83)
Review – 1

ENHANCING THE PERFORMANCE OF ANTIPHISHING MECHANISM USING MACHINE LEARNING

Group No.: 06

Batch No.: 2021_CSE_08

R Soumya 1KS18CS125
Sri Chandana P 1KS18CS098
Vijetha 1KS18CS117
Sushmitha S 1KS18CS106

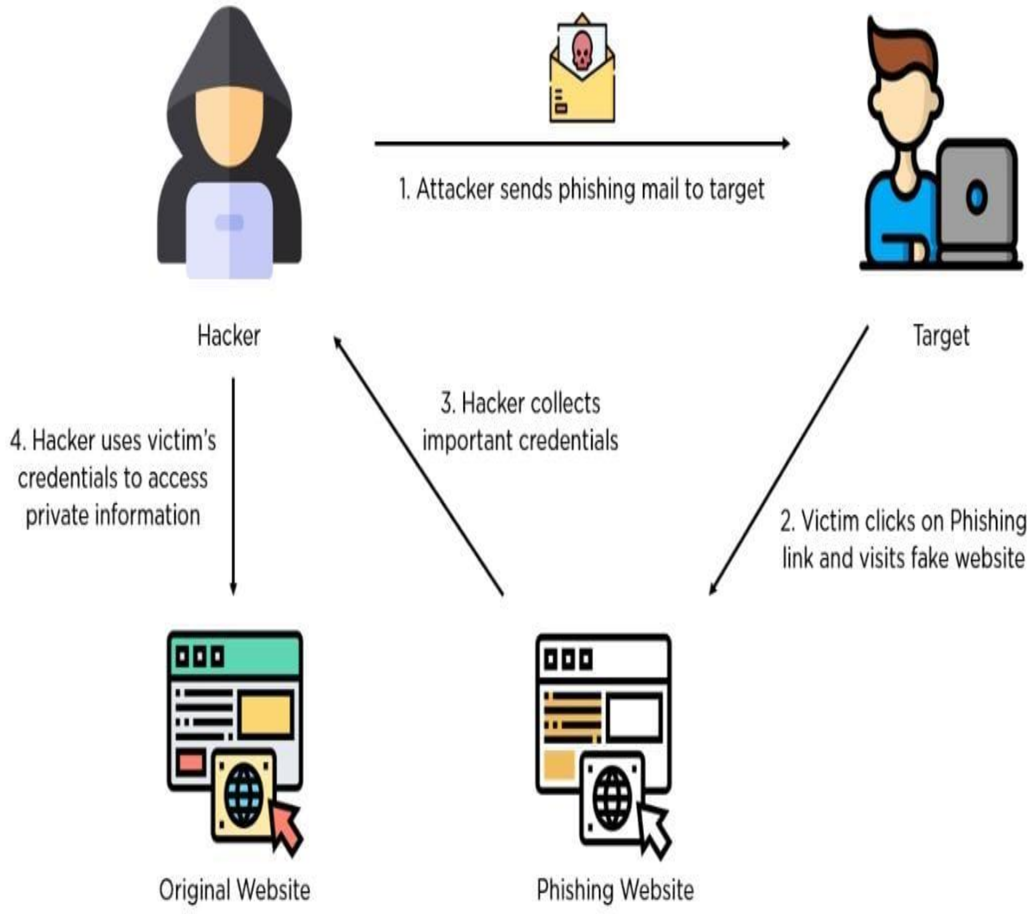
Guided By:
Mr. Roopesh Kumar BN
Asst. Professor Dept of CSE
KSIT Bengaluru

Contents

- Introduction
- Comparison with similar work
- Problem Statement and Objectives
- Methodology
- Technologies and tools used
- Implementation of Modules with codes
- Snapshots
- References

Introduction

- The cyber security problems are increasing nowadays due to the growth of internet world wide one of them is phishing.
- In this attacker creates a replica of existing link or webpage to fool the user to get access to the personal information.
- Phishers use multiple methods, including email, uniform resource locators(URL), instant messages, forum postings, telephone calls, and text messages to steal user information.



Introduction

- The success mainly depends on recognizing phishing websites accurately and within an acceptable timescale.
- The ML based phishing techniques depend on website functionalities to gather information that can help classify websites for detecting phishing sites. Here some common supervised learning techniques are applied to accurately detect phishing websites.

Comparison with similar work

1. Google Safe Browsing:

- It uses blacklist anti-phishing technique to detect phishing .The suspicious URL is checked in the blacklist for its presence.
- The limitation in this approach is that phishing sites which are not listed in blacklist are not detected.

2. Based on visual similarity:

- This technique classifies the suspicious websites based on image similarity.
- It can auto update the whitelist with addition of suspicious websites that are classified as neither legitimate nor phishing.
- The limitation of this approach is very high false positive and high false negative rate and image comparison at client side leads to delay in browser’s experience.

Comparison with similar work

3. PhishNet:

- This technique takes blacklist as input and predicts variations of each URL.
- This technique also has the limitation of not detecting zero day phishing attacks.

4. SpoofGaurd:

- This technique takes some phishing symptoms of suspicious website and assigned some weights to classified as phishing website otherwise as legitimate.
- It has an advantage of detecting zero day phishing attack but has a limitation of high false positive rate.

Problem Statement and Objectives

"To design the machine learning model that enhances the performance of anti-phishing mechanism".

OBJECTIVES

- To design a model that detects the phishing websites accurately.
- To prevent the identity theft of the online users.

Methodology

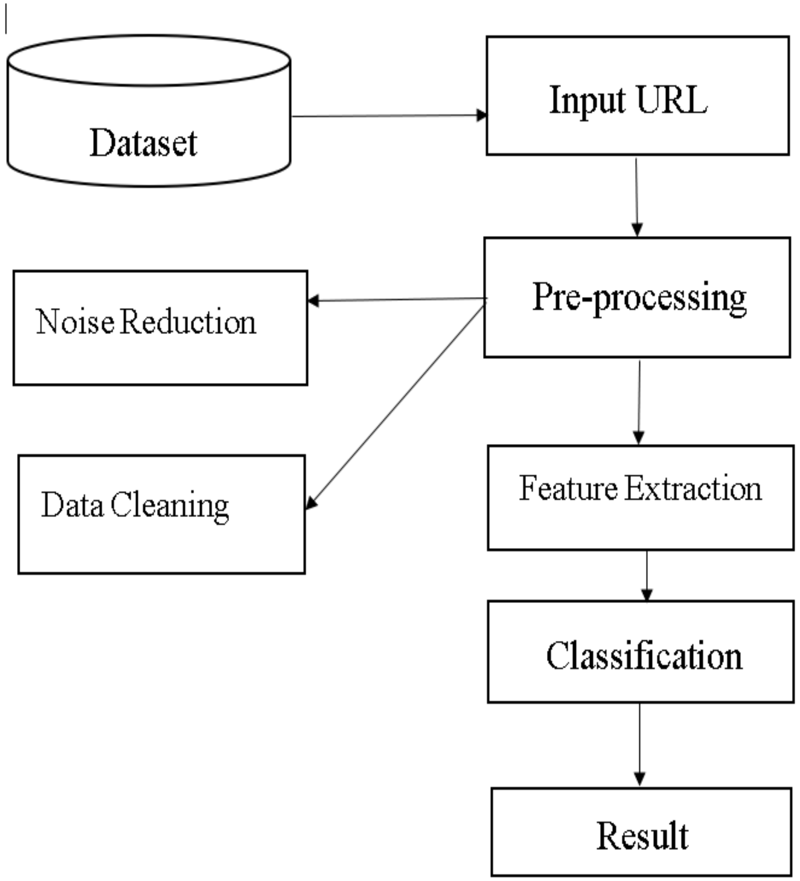
Data Acquisition

- Dataset is collected from UCI and Kaggle websites.
- No. of records= 11056
- Training dataset size= 75% (7738)
- Test dataset size= 25% (3318)

Methodology

CNN-Model

1. This model involves feature extraction and classification.
2. In feature extraction, CNN is used to extract local features.
3. The CNNs are used to learn sequential information from the
4. Extracted features are converted to matrix form which is don
   by the hidden layers.
5. We aim to get a matrix representation as
   u → G ∈ R^(L×k) where gi ∈ Rk.
6. An instance can be depicted in the concatenation of L comp
   G = G1:L = g1 ⊕ g2 ⊕ ... ⊕ gL



Methodology

7. It forms a pooling layer, these features are grouped and passed to fully connected(FC) layers for classification purposes.
8. To get a more accurate result, it is combined with the LSTM model.

Accuracy and Prediction

- Result of CNN-LSTM model is compared with XG-Boost.
- CNN-LSTM and XG-Boost will give an accuracy rate and predicts whether the input URL is phished or genuine separately.

Technologies and tools used

Software Requirements

- Windows 7
- Python
- Anaconda

Libraries Used:

- Numpy
- Pandas
- Keras
- Tensorflow
- Matplotlib

Implementation of modules with codes

Data Pre-processing :

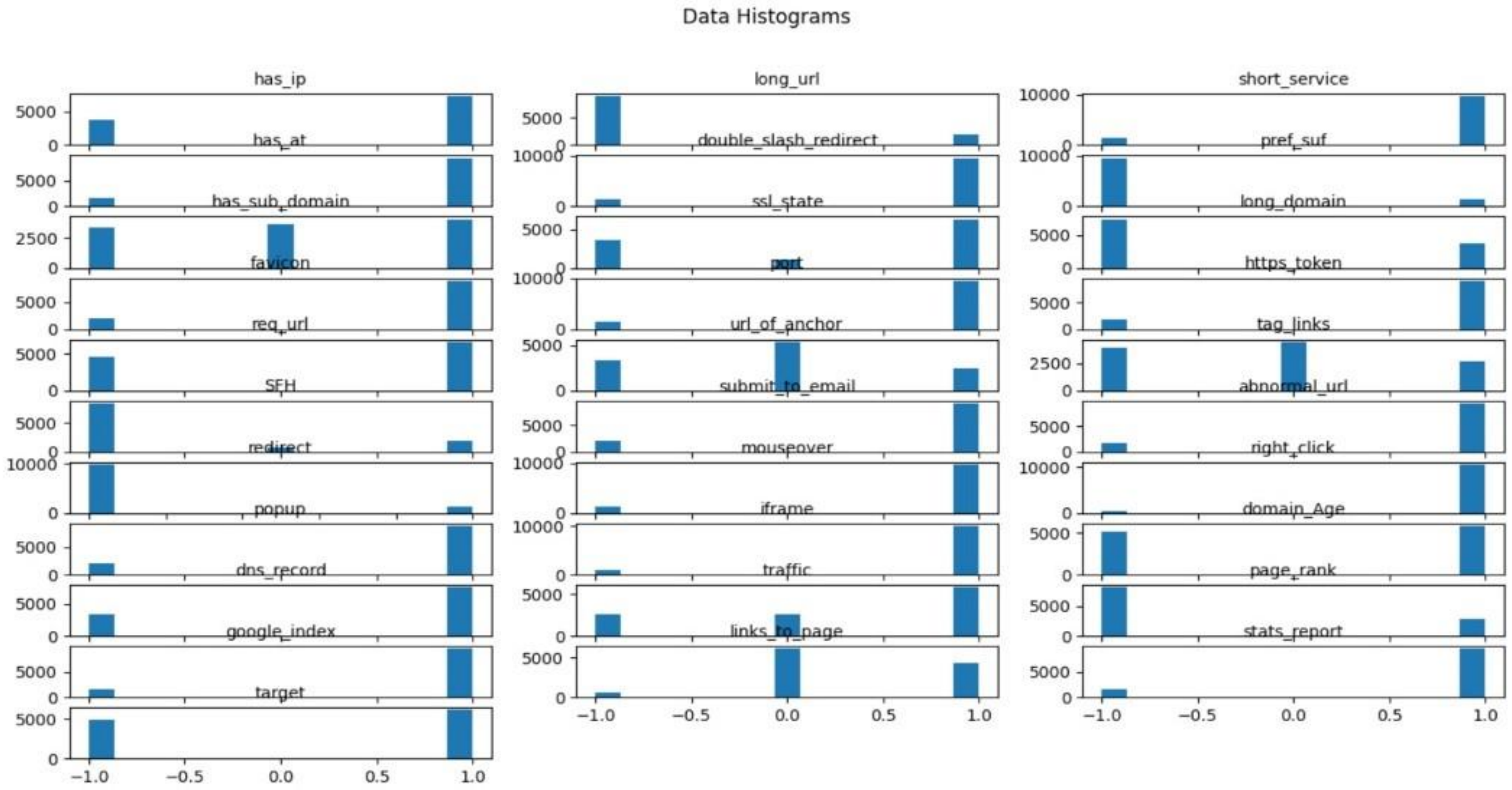
```
plt.subplot(nrows, ncols, i+1)
plt.hist(data[var].values, bins=15)
plt.title(var, fontsize=10)
plt.tick_params(labelbottom='off', labelleft='off’)
plt.tight_layout()
plt.subplots_adjust(top=0.88)
fig.savefig('results/DataHistograms.png’)
```

Implementation of modules with codes

Feature extraction and evaluation :

```
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size = 0.25, random_state =0 )

x_train = np.reshape(x_train, (x_train.shape[0], x_train.shape[1],1))

x_test = np.reshape(x_test, (x_test.shape[0], x_test.shape[1],1)

model = build_model([x_train.shape[0], x_train.shape[1],1])

print(model.summary())

start = timer()
hist=model.fit(x_train,y_train,batch_size=128,epochs=25,validation_split=0.2,verbose=2)

model.save('results/CNN.h5’);
```

Snapshots



Data Histograms
has_ip
long_url
short_service
has_at
double_slash_redirect
pref_suf
has_sub_domain
ssl_state
long_domain
favicon
port
https_token
req_url
url_of_anchor
tag_links
SFH
submit_to_email
abnormal_url
redirect
mouseover
right_click
popup
iframe
domain_Age
dns_record
traffic
page_rank
google_index
links_to_page
stats_report
target

Snapshots

```
Train on 6632 samples, validate on 1659 samples
Epoch 1/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 2/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 3/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 4/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 5/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 6/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 7/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 8/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 9/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 10/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 11/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 12/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 13/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 14/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 15/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 16/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 17/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 18/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 19/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 20/25
 - 1s - loss: 1.0000 - accuracy: 0.0000e+00 - val_loss: 1.0000 - val_accuracy: 0.0000e+00
Epoch 21/25
```

Snapshots

Anaconda Prompt (Anaconda3) - python Main.py

```
 - 1s - loss: 0.5673 - accuracy: 0.4735 - val_loss: 0.5876 - val_accuracy: 0.4557
Epoch 13/25
 - 1s - loss: 0.5660 - accuracy: 0.4691 - val_loss: 0.5820 - val_accuracy: 0.4659
Epoch 14/25
 - 1s - loss: 0.5609 - accuracy: 0.4742 - val_loss: 0.5828 - val_accuracy: 0.4334
Epoch 15/25
 - 1s - loss: 0.5579 - accuracy: 0.4762 - val_loss: 0.5776 - val_accuracy: 0.4684
Epoch 16/25
 - 1s - loss: 0.5515 - accuracy: 0.4837 - val_loss: 0.5751 - val_accuracy: 0.4497
Epoch 17/25
 - 1s - loss: 0.5640 - accuracy: 0.4694 - val_loss: 0.5964 - val_accuracy: 0.4219
Epoch 18/25
 - 1s - loss: 0.5539 - accuracy: 0.4812 - val_loss: 0.5716 - val_accuracy: 0.4665
Epoch 19/25
 - 1s - loss: 0.5450 - accuracy: 0.4824 - val_loss: 0.5714 - val_accuracy: 0.4461
Epoch 20/25
 - 1s - loss: 0.5490 - accuracy: 0.4818 - val_loss: 0.5959 - val_accuracy: 0.5057
Epoch 21/25
 - 1s - loss: 0.5466 - accuracy: 0.4821 - val_loss: 0.5709 - val_accuracy: 0.4858
Epoch 22/25
 - 1s - loss: 0.5437 - accuracy: 0.4887 - val_loss: 0.5707 - val_accuracy: 0.4840
Epoch 23/25
 - 1s - loss: 0.5441 - accuracy: 0.4851 - val_loss: 0.5609 - val_accuracy: 0.4750
Epoch 24/25
 - 1s - loss: 0.5365 - accuracy: 0.4872 - val_loss: 0.5578 - val_accuracy: 0.4671
Epoch 25/25
 - 1s - loss: 0.5358 - accuracy: 0.4907 - val_loss: 0.5569 - val_accuracy: 0.4828
2764/2764 [==============================] - 0s 43us/step
0.5586618362735908 0.48697540163993835
```

Type here to search

ENG 20:48 27-04-2022

References

[1] A S S V Lakshmi Pooja, Sridhar.M, "Analysis of Phishing Website Detection Using CNN and Bidirectional LSTM ", in Fourth International Conference on Electronics, Communication and Aerospace Technology (ICECA-2020) IEEE Xplore Part Number: CFP20J88-ART; ISBN: 978-1-7281-6387-1, 2020.

[2] Ali Aljofey, Qingshan Jiang, Qiang Qu, Mingqing Huang, Jean-Pierre Niyigena, "An Effective Phishing Detection Model Based on Character Level Convolutional Neural Network from URL ", in MDPI Journal, September 2020.

[3] M. A. Adebowale, K. T. Lwin, M. A. Hossain, "Deep Learning with Convolutional Neural Network and Long Short-Term Memory for Phishing Detection ", in Conference paper IEEE, August 2019.