

**A Project Report**

**On**

# Malicious URL Detection using

# Machine Learning in Python

*Submitted In Partial Fulfillment of the Requirement for the Award of* **Post Graduate Diploma in Artificial Intelligence (PG-DAI)** Under the Guidance of

**MR. NIMESH DAGUR**

**(Project Guide)**



Submitted By

**NIKHIL DAHITULE PRIYA KARVANDE JAYESH PATIL**

**PRN: 220920528027 PRN: 220920528032 PRN: 220920528016**

# CONTENTS

|  |  |  |
| --- | --- | --- |
| **INDEX** | **TITLE** | **PAGE NUMBER** |
| I | Certificate | **2** |
| II | Acknowledgement | **3** |
| III | Abstract | **4** |
| IV | Introduction to the problem statement and the possible solution | **5** |
| V | Data Preprocessing | **6** |
| VI | Coding | **7-17** |
| VII | Results | **18** |
| VIII | Conclusion & Future Scope | **19-20** |
| IX | Reference & Bibliography | **21** |

CDAC, B-30, Institutional Area, Sector-62 Noida (Uttar Pradesh)-201307

# CERTIFICATE CDAC, NOIDA

This is to certify that Report entitled **“A Malicious URL Detection using**

**Machine learning”** which is submitted by Nikhil Dahitule, Priya Karvande and Jayesh Patil in partial fulfillment of the requirement for the award of **Post Graduate Diploma in Artificial Intelligence** (PG-DAI) to **CDAC, Noida** is a record of the candidates own work carried out by them under my supervision.

The documentation embodies results of original work, and studies are carried out by the student themselves and the contents of the report do not from the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

**MR. NIMESH DAGUR**

**(Project Guide)**

# ACKNOWLEDGEMENT

We would like to express our best sense of gratitude & endeavor with respect to **Mr. Nimesh Dagur (Project Guide)** CDAC, Noida for suggesting the problems scholarly guidance and expert supervision during the course of this project. Special thanks to **Mr. Ravi Payal (Program Coordinator).**

We are very thankful to **Mr. Nimesh Dagur (Project Guide)** the project guide for constant simulated discussion, encouraging new ideas about this project.

**ABSTRACT**

Malicious URL detection is an important task in cyber-security aimed at identifying and blocking URLs that lead to malicious content, such as phishing sites, malware downloads, and fraudulent pages. This task is crucial in protecting users from cyber attacks and ensuring the security of their online activity. Malicious URL detection can be achieved through various techniques, including blacklisting, Machine Learning, and behavior-based analysis. In recent years, deep learning models have shown promising results in this area, allowing for more accurate and efficient detection of malicious URLs. Despite the advances in this field, the detection of malicious URLs remains a challenging problem due to the constantly evolving nature of cyber threats. Therefore, ongoing research is needed to develop robust and adaptive solutions to address this problem.

# INTRODUCTION TO THE PROBLEM STATEMENT

* Internet has dominated the world by dragging half of the world’s population exponentially into the cyber world. With the booming of internet transactions, cybercrimes rapidly increased and with anonymity presented by the internet, Hackers attempt to trap the end-users through various forms such as phishing, malware, man-in-the-middle, and so on. Among all these attacks, phishing reports to be the most deceiving attack.
* Our main aim of this project is classification of a phishing website with the aid of various machine learning & deep learning techniques to achieve maximum accuracy and concise model.
* We will train machine learning models extracting past data to make decision or prediction on future data. Using this technique, algorithm will analyze various blacklisted and legitimate URLs and their features to accurately detect phishing URLs as well as narrow down to best machine learning algorithm by comparing accuracy rate, false positive and false negative rate of each algorithm

# Data Pre-processing

In this case study, we will be using a [Malicious URLs dataset](https://www.kaggle.com/datasets/sid321axn/malicious-urls-dataset) of **6,51,191**URLs, out of which **4,28,103** benign or safe URLs, **96,457** defacement URLs, **94,111** phishing URLs, and **32,520** malware URLs.

Now, let’s discuss different types of URLs in our dataset i.e., Benign, Malware, Phishing, and Defacement URLs.

**Benign URLs:**These are safe to browse URLs. Some of the examples of benign URLs are as follows:

**mp3raid.com/music/krizz\_kaliko.html**

**infinitysw.com**

**google.co.in**

**myspace.com**

**Malware URLs:**These type of URLs inject malware into the victim’s system once he/she visit such URLs. Some of the examples of malware URLs are as follows:

**proplast.co.nz**

**http://103.112.226.142:36308/Mozi.m**

**microencapsulation.readmyweather.com**

**xo3fhvm5lcvzy92q.download**

**Defacement URLs**: Defacement URLs are generally created by hackers with the intention of breaking into a **web server** and replacing the **hosted** **website** with one of their own, using techniques such as **code injection**, **cross-site scripting**, etc. Common targets of **defacement** URLs are religious websites, government websites, bank websites, and corporate websites. Some of the examples of defacement URLs are as follows:

**http://www.vnic.co/khach-hang.html**

**http://www.raci.it/component/user/reset.html**

**http://www.approvi.com.br/ck.htm**

**http://www.juventudelirica.com.br/index.html**

**Phishing URLs:** By creating phishing URLs, hackers try to steal sensitive personal or financial information such as login credentials, credit card numbers, internet banking details, etc. Some of the examples of phishing URLs are shown below:

**roverslands.net**

**corporacionrossenditotours.com**

**http://drive-google-com.fanalav.com/6a7ec96d6a**

**citiprepaid-salarysea-at.tk**

Next, we will plot the word cloud of different types of URLs.

**Wordcloud of URLs**

The word cloud helps in understanding the pattern of words/tokens in particular target labels.

It is one of the most appealing techniques of natural language processing for understanding the pattern of word distribution.

As we can see in the below figure word cloud of benign URLs is pretty obvious having frequent tokens such as html, com, org, wiki etc. Phishing URLs have frequent tokens as tools, **ietf, www, index, battle, net** whereas **html, org, html** are higher frequency tokens as these URLs try to mimick original URLs for deceiving the users.

The word cloud of malware URLs has higher frequency tokens of **exe, E7, BB, MOZI**. These tokens are also obvious as malware URLs try to install trojans in the form of executable files over the users’ system once the user visits those URLs.

The defacement URLs’ intention is to modify the original website’s code and this is the reason that tokens in its word cloud are more common development terms such as **index, php, itemid, https, option,** etc.

**Training**

# CODING

7-17 coding remaining

from \_\_future import print\_function import keras

from keras.preprocessing.image import ImageDataGenerator from keras.models import Sequential

from keras.layers import Dense,Dropout,Activation,Flatten,BatchNormalization

from keras.layers import Conv2D,MaxPooling2D import os

num\_classes = 2

img\_rows,img\_cols = 48,48

batch\_size = 32

train\_data\_dir = r'D:\Dixant\CDAC\Project\face-expression-recognition- dataset\train'

validation\_data\_dir = r'D:\Dixant\CDAC\Project\face-expression-recognition- dataset\validation'

train\_datagen = ImageDataGenerator(rescale=1./255,

rotation\_range=30, shear\_range=0.3, zoom\_range=0.3, width\_shift\_range=0.4,

height\_shift\_range=0.4, horizontal\_flip=True, fill\_mode='nearest')

validation\_datagen = ImageDataGenerator(rescale=1./255)

train\_generator = train\_datagen.flow\_from\_directory(train\_data\_dir,

color\_mode='grayscale', target\_size=(img\_rows,img\_cols), batch\_size=batch\_size, class\_mode='categorical', shuffle=True)

validation\_generator = validation\_datagen.flow\_from\_directory(validation\_data\_dir,

color\_mode='grayscale', target\_size=(img\_rows,img\_cols), batch\_size=batch\_size, class\_mode='categorical', shuffle=True)

model = Sequential()

# Block-1

model.add(Conv2D(32,(3,3),padding='same',kernel\_initializer='he\_normal',inp ut\_shape=(img\_rows,img\_cols,1)))

model.add(Activation('elu')) model.add(BatchNormalization())

model.add(Conv2D(32,(3,3),padding='same',kernel\_initializer='he\_normal',inp ut\_shape=(img\_rows,img\_cols,1)))

model.add(Activation('elu')) model.add(BatchNormalization()) model.add(MaxPooling2D(pool\_size=(2,2))) model.add(Dropout(0.2))

# Block-2

model.add(Conv2D(64,(3,3),padding='same',kernel\_initializer='he\_normal')) model.add(Activation('elu'))

model.add(BatchNormalization()) model.add(Conv2D(64,(3,3),padding='same',kernel\_initializer='he\_normal')) model.add(Activation('elu'))

model.add(BatchNormalization()) model.add(MaxPooling2D(pool\_size=(2,2))) model.add(Dropout(0.2))

# Block-3

model.add(Conv2D(128,(3,3),padding='same',kernel\_initializer='he\_normal')) model.add(Activation('elu'))

model.add(BatchNormalization()) model.add(Conv2D(128,(3,3),padding='same',kernel\_initializer='he\_normal')) model.add(Activation('elu'))

model.add(BatchNormalization()) model.add(MaxPooling2D(pool\_size=(2,2))) model.add(Dropout(0.2))

# Block-4

model.add(Conv2D(256,(3,3),padding='same',kernel\_initializer='he\_normal')) model.add(Activation('elu'))

model.add(BatchNormalization()) model.add(Conv2D(256,(3,3),padding='same',kernel\_initializer='he\_normal')) model.add(Activation('elu'))

model.add(BatchNormalization()) model.add(MaxPooling2D(pool\_size=(2,2))) model.add(Dropout(0.2))

# Block-5

model.add(Flatten()) model.add(Dense(64,kernel\_initializer='he\_normal')) model.add(Activation('elu'))

model.add(BatchNormalization()) model.add(Dropout(0.5))

# Block-6

model.add(Dense(64,kernel\_initializer='he\_normal')) model.add(Activation('elu')) model.add(BatchNormalization()) model.add(Dropout(0.5))

# Block-7

model.add(Dense(num\_classes,kernel\_initializer='he\_normal')) model.add(Activation('softmax'))

print(model.summary())

from tensorflow.keras.optimizers import RMSprop,SGD,Adam

from keras.callbacks import ModelCheckpoint, EarlyStopping,

ReduceLROnPlateau

checkpoint = ModelCheckpoint('Emotion\_training.h5', monitor='val\_loss',

mode='min',

save\_best\_only=True, verbose=1)

earlystop = EarlyStopping(monitor='val\_loss', min\_delta=0,

patience=3, verbose=1,

restore\_best\_weights=True

)

reduce\_lr = ReduceLROnPlateau(monitor='val\_loss',

factor=0.2, patience=3, verbose=1, min\_delta=0.0001)

callbacks = [earlystop,checkpoint,reduce\_lr]

model.compile(loss='categorical\_crossentropy', optimizer = Adam(learning\_rate=0.001), metrics=['accuracy'])

nb\_train\_samples = 16095

nb\_validation\_samples = 3924

epochs=40

history=model.fit(train\_generator,

steps\_per\_epoch=nb\_train\_samples//batch\_size, epochs=epochs,

callbacks=callbacks, validation\_data=validation\_generator, validation\_steps=nb\_validation\_samples//batch\_size)

**Testing**

from keras.models import load\_model from time import sleep

from keras.preprocessing.image import img\_to\_array from keras.preprocessing import image

import cv2 import os

import numpy as np

cam = cv2.VideoCapture(r'Rajpal.mp4')

try:

if not os.path.exists('data'): os.makedirs('data')

except OSError:

print ('Error: Creating directory of data')

currentframe = 0

while(True):

ret,frame = cam.read()

if ret:

name = './data/frame' + str(currentframe) + '.jpg' print ('Creating...' + name)

cv2.imwrite(name, frame)

currentframe += 1 else:

break

cam.release() cv2.destroyAllWindows()

face\_classifier = cv2.CascadeClassifier(r'D:\Dixant\CDAC\Project\Facial- Expressions-Recognition\Facial-Expressions-Recognition- master\haarcascade\_frontalface\_default.xml')

classifier =load\_model('Emotion\_training.h5')

class\_labels = ['Happy','Sad']

emotions=[] i=0

while True: try:

print(i)

file=(r"C:\Users\divya\Documents\data\frame"+str(i)+".jpg") img=cv2.imread(file) gray=cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY)

i+=1

faces=face\_classifier.detectMultiScale(gray,1.1,4) for(x,y,w,h) in faces:

cv2.rectangle(img,(x,y),(x+w,y+h),(0,255,0),3) roi\_gray = gray[y:y+h,x:x+w]

roi\_gray = cv2.resize(roi\_gray,(48,48),interpolation=cv2.INTER\_AREA)

if np.sum([roi\_gray])!=0:

roi = roi\_gray.astype('float')/255.0 roi = img\_to\_array(roi)

roi = np.expand\_dims(roi,axis=0)

preds = classifier.predict(roi)[0] label=class\_labels[preds.argmax()] label\_position = (x,y) emotions.append(label)

cv2.putText(img,label,label\_position,cv2.FONT\_HERSHEY\_SIMPLEX,2,(0,2 55,0),3)

else:

cv2.putText(img,'No Face Found',(20,60),cv2.FONT\_HERSHEY\_SIMPLEX,2,(0,255,0),3)

continue

cv2.waitKey(0) cv2.destroyAllWindows()

except:

"error in image" break

import shutil path='C:/Users/divya/Documents/data' shutil.rmtree(path)

len(emotions)

from collections import Counter d = Counter(emotions)

Sad = d['Sad'] Happy = d['Happy'] Sad

Total=Sad + Happy emotion\_1 = Sad/Total emotion\_1

if emotion\_1 > 0.70:

print("Severe Depression : consult with psychology immediately") elif emotion\_1 > 0.40:

print("Mild Depression ") elif emotion\_1 > 0.10:

print("Low Depression ") else:

print("You don't have Depression")

# RESULTS

# CONCLUSION AND FUTURE SCOPE

In conclusion, malicious URL detection is an essential aspect of cybersecurity that helps protect individuals and organizations from various online threats. There are several techniques and tools available for detecting and blocking malicious URLs, including signature-based methods, heuristic-based methods, machine learning-based methods, and blacklists. These techniques leverage different sources of information such as domain names, IP addresses, and web content to identify and classify malicious URLs accurately.

To improve the effectiveness of malicious URL detection, it is essential to keep the detection mechanisms up to date with the latest threats and attack vectors. This requires continuous monitoring and analysis of online activities, as well as collaboration among cyber-security experts and organizations to share threat intelligence and best practices.

Overall, malicious URL detection is a critical component of a comprehensive cyber-security strategy, and individuals and organizations must remain vigilant in protecting themselves from online threats by staying informed and adopting

Informed.

# REFERENCES & BIBLIOGRAPHY