Predicting Malicious URLs via their lexical properties using a Deep Neural Network

Justin Fulner

Project Overview

For this machine learning project, I decided to try and build a deep neural network to predict if a URL is malicious or not.

To accomplish this, URLs are categorized into 4 types:

- Benign
- Defacement
- Malware
- Phishing

Overview Cont.

The dataset that the model is trained on was authored by Manu Siddhartha

(https://www.kaggle.com/sid321axn/malicious).

(For the purposes of this project, the dataset is hosted on my personal Google drive and automatically downloaded by the python script)

However, the dataset only contains a list of URLs followed by their threat status.

Thus I had to extract various lexical features from typical URL schemes and build them onto the dataset in order to use it to train my model.

Lexical Features

I extracted 8 common features in total:

- URL length
- Presence of HTTPS in the url
- Presence and number of HTTP in the URL
- Length of the URL domain
- Number of digits in the URL
- Number of ‘@’s in the URL
- Number of ‘.’s in the URL
- Number of sub-directories in the URL path

Overview Cont.

Once these features are extracted and added to the dataset, the dataset is converted into a series of tensors and sent off to the DNN classifier for training.

Once training is complete, the python script will wait for a user to input new URLs in order to predict their class based on the model produced by training.

Libraries/packages

Pandas

Urllib.parse

tensorflow

Algorithm

For this project, I elected to use a Deep Neural Network.

This is partly due to the fact that I had not built a DNN before. However, I also thought it was necessary given that my data set is comprised of over 650 thousand samples. Thus the performance advantages of a DNN appealed to me over other classifiers.

Algorithm Performance

Based on my analysis of the samples I fed to the algorithm for testing, I came to the conclusion that the model I developed is wildly inaccurate. The precision and accuracy of the predictions tend to perform better if I adjust my hyperparameters to higher levels (i.e. adding layers and increasing depth, increasing batch size, etc.); however, the gains are minimal.

I suspect this is due to my selection of features. It could be that there simply aren't enough, or that the ones I have extracted from the samples are poorly chosen, or both.

It is also possible that I simply chose the wrong type of classifier for this problem; however, I don't think it is prudent to change the model given my above conclusions.

Future Plans

- Examine and extract more lexical features in order to make the model more robust.
- Automate the supply of URLs to the model and direct results to a database.
  - through use of a web crawler or something similar
- Possibly some form of front end GUI

Thank you for your time