ICYBM201 – Flagging Suspicious Hosts From DNS Traces

1 Overview of the Project

We assume a very simple universe in which only traffic from Web requests are considered to be generated by a human, and authorised in the network. Bots are also inside the network, potentially from different Botnets, and also performing DNS requests for an unspecified purpose. Your task would be to build a classifier to classify each host from the dataset as either human, bot or human+bot, and report your methodology, results, and analysis. To do so, you will have access to several labeled datasets containing DNS traces captured only from humans, only from bots, and from hosts potentially mixing bot and human activity. The method to classify is free for you to choose, yet you should in any case provide a critical discussion of your results, hopefully related to the material seen in class.

Your classifier would then be evaluated over several datasets with a similar structure and containing various amounts of bots/humans.

2 Datasets

Several UNamur hosts are connected to the Internet and configured with the 1.1.1.1 public resolver. Some of their DNS traffic has been captured from a vantage point in the network using tcpdump on port 53. First, two training datasets are given:

- webclients_tcpdump.txt contains the DNS traces of 120 UNamur hosts browsing several frontpages of the top-1000 Alexa [1] list.
- bots_tcpdump.txt contains the DNS traces of 120 bots, also in relation to the top-1000 Alexa list.

[1] Alexa is an Amazon service ranking the most visited domains

You receive also two typical datasets on which you can evaluate your classifier.

- eval1_tcpdump.txt and eval2_tcpdump.txt contain both human and bots, and are typical dataset examples on which we will evaluate your classifier. In eval1_tcpdump.txt bots and humans are guaranteed to be a separated set of hosts. In eval2_tcpdump.txt, some hosts emit traffic from a human and from a bot. The lists of bots are given in the files eval1_botlist.txt and eval2_botlist.txt. You should use these datasets to evaluate your classifier and report your results.

3 Deliverable Instructions

You must give a zip file containing several python3.8+ scripts, but you're free to use them to call any other program:

- `train.py` is an optional step to train your classifier and output a trained model. It should support the following interface:

```
python train.py --webclients path/to/webclients_tcpdump.txt --bots
path/to/bots_tcpdump.txt --output path/to/trained_model
```

- `eval.py` evaluates whether the hosts within captured DNS traffic are suspicious. It optionally takes in input the trained model produced from `train.py` and should output a list of suspicious host names (one per line).

```
python eval.py --trained_model path/to/trained_model --dataset
path/to/dataset --output path/to/suspicious.txt
```

  Example of output:

```
unamur033
unamur111
unamur018
```

- An optional `requirements.txt` file listing package dependencies to run `train.py` and `eval.py` if any.

You should also give us a report explaining how you approached the problem, what algorithm(s) and technique(s) you decided to experiment and use, potentially how they compared to each other. You're expected to be critical on potential issues you faced. The report should be a .pdf, and zipped with your code.

Your report should contain a critical discussion of your methodology and evaluation.

4 Evaluation

You will be evaluated on your approach over the problem, your methodology and on the discussion of your results. Your code and results obtained over our datasets will also contribute to the final grade, as well as whether you correctly respected all the specifications written in these instructions. Try to follow coding conventions: https://peps.python.org/pep-0008/.