Cyber Data Analytics Assignment 3

INTRODUCTION

Most datasets from software or hardware systems do not fit into memory. Data stream mining is a machine learning framework that aims to store only the data parts that are relevant for learning, e.g., by sampling and hashing.

Profiling and fingerprinting are two key techniques (in addition to anomaly detection) for threat discovery. Profiling builds an overall picture, typically a probability distribution, from training data and matches this against new data. Fingerprinting instead looks for very specific patterns that only occur when a threat is present.

In this exercise, you will apply the techniques taught in class to build approximations of a large network data streams on-the-fly and evaluate the quality of the obtained approximations. In addition, you will apply fingerprinting/profiling to the problem of botnet detection in computer networks and compare it with flow classification.

LEARNING OUTCOMES

After completing this assignment, you will be able to:

1. Use sampling and hashing to create approximations from large data streams.
2. Build a locality sensitive hashing scheme for fast distance computations.
3. Build methods for fingerprinting and profiling sequential behaviours.
4. Successfully detect botnets in network data.

INSTRUCTIONS

In this assignment you will work with the *bidirectional* netflows fom the CTU-13 datasets (Malware capture 50 to 53, scenario 9 to 12). They will be made available on Brightspace. The flows are collected from a host in the network. Its IP address should be obvious from the data sample.

Familiarization and discretization task (5 points)

- Investigate **scenario 10** from the CTU-13 datasets (see paper 4 from below resources)
- You are to discretize the NetFlows, investigate the data from one of the infected hosts. Select and visualize two features that you believe are most relevant for modeling the behavior of the infected host. Discretize these features using use

any of the methods discussed in class (combine the two values into a single discrete value)
- Do you observe any behavior in the two features that could be useful for detecting the infection? Explain and visualize. Apply the discretization to data from all hosts in scenario 10

The following individual tasks should only be performed on CTU-13 scenario 10. One student works on sketching and the other on min-wise LSH.

Sketching task, individual (10 points)

- Implement a COUNT-MIN sketch (do not use a library!) to estimate occurrence counts for the 3-grams. Make sure the hash functions are pairwise independent.
- Use the discretization from task 1 to build to estimate the 3-gram distribution of the entire data in one pass
- Play with different heights and widths for the COUNT-MIN sketch matrix. What are the 10 most frequent 3-grams and their frequencies when approximated?
- Compare the performance, run-time, and space requirements of the frequency estimation with the count-min estimation. When do you recommend using count-min sketches?

Min-wise locality sensitive hashing task, individual (10 points)

- Implement min-wise locality sensitive hashing (do not use a library! No need to implement banding!) as explained in the slides and the documents on Brightspace. This can be used to quickly compute the Jaccard distance for N-gram profiles.
- Use the discretization from task 1 to build 3-gram profiles for every individual connection (pair of IP-addresses). For this task, the profiles are binary, an N-gram (subsequence) exists (a 1 in the table) or does not (a 0 in the table).
- Use min-wise LSH to map the 3-gram profiles to a small set of bins of your choice (a signature). Play with the number of bins. Compute how often the true nearest neighbor is equivalent to the nearest neighbor using the signatures.
- Compare the performance, run-time, and space requirements of the nearest-neighbor search with the min-wise search. When do you recommend using min-wise LSH?

The profiling and fingerprinting tasks should be performed on CTU-13 scenarios 9-12, per scenario (learn and evaluate on the same scenario).

Profiling and Fingerprinting task (5 points)

- Use the discretization from task 1 to learn an advanced sequential model, either a Deterministic Finite Automaton or a hidden Markov model, from a known infected botnet IP, see Brightspace for details.
- Collect and sum up the state-symbol counts of the train data for this botnet as its profile.
- Compute a distance (e.g., cosine) between the state counts of other hosts and the learned profile, set a threshold and plot the counts of a true/false positive and true/false negative in a scatter plot as in the slides.
- Find fingerprints (a state that occurs frequently in the botnet but never/rarely in normal hosts) and determine whether these fingerprints occur in all other hosts.
- Explain why the profile/fingerprint matches for at least one true and one false positive (if any), from any scenario.

Bonus!: Competition (5 points)

- Apply a method of your choice to the competition test set. This may be anomaly detection (profiling normal instead of malicious), fingerprinting, profiling, hashing/sketching, or a combination. You may apply this to individual connections or hosts but remember to assign labels to every flow in the test data. Describe why you chose this method, what performance you expect, and why.

RESOURCES

Slides from Lectures 5, 6. Study:

1. https://stratosphereips.org
2. In particular the CTU-13 data: https://www.stratosphereips.org/datasets-ctu13/
3. Garcia, Sebastian, et al. "An empirical comparison of botnet detection methods." *computers & security* 45 (2014): 100-123.
4. Pellegrino, Gaetano, et al. "Learning Behavioral Fingerprints From Netflows Using Timed Automata."
5. Papers on hashing available on Brightspace.

Links on Brightspace to online tutorials. Code samples available on Brightspace.

PRODUCTS

A zip containing:

- A Jupyter Python notebook for all parts of the assignment (including individual tasks). The word count should not exceed 1600 words (see first cell). Include libraries used to run the code other than numpy, scipy, pandas, and scikit-learn.

The notebooks will be assessed using the below criteria.

ASSESSMENT CRITERIA

The assignment will be reviewed by your peers, and you are expected to individually review 2 reports. The estimated time you should spend on a review (including code review) is 1 hour. The login details will be provided in the week of the deadline.

Knockout criteria (will not be evaluated if unsatisfied):

Your code needs to execute successfully on computers/laptops of your fellow students (who will assess your work). You may assume the availability of 4GB RAM. Please test your code before submitting. In addition, the flow from data to prediction has to be highlighted, e.g., using inline comments.

Your report needs to satisfy the page limit requirements for the different parts. Submissions submitted after the deadline will not be graded.

The report/code will be assessed using these criteria:

Criteria	*Description*	*Evaluation*
Familiarization	*Shows the behavior of two features conditioned on the infection status. The discretization is sound.*	*0-5 points*
Count-min sketching	*Sketching is implemented correctly, with explanations for the number of used counters/bins. The 3-gram count approximation is correct, and its quality is related to theory.*	*0-10 points*
Min-Hash LSH	*LSH is implemented correctly. The number of bins is set sensibly. The resulting comparison explains differences in run-time and quality, and its quality is related to the theory*	*0-10 points*
Profiling/ Fingerprinting	*Profiles/Fingerprints are built correctly using state-symbol counts. Evaluation and visualization are sound and insightful. Analysis of false/true positives are correct.*	*0-5 points*
Bonus	*Performance is OK, reasoning for used technique is sound and reason for expected performance or insightful.*	*0-5 points*
Report and code	*The data-detection flow is clearly described, including preprocessing and post-processing steps.*	*0-5 points*

Your total score will be determined by summing up the points assigned to the individual criteria. Your report and code will be graded by the teacher and assistants, and the peer reviews are used as guidance.

In total 130 points (including bonus) can be obtained in the 4 lab assignments, of which 30 are individual. The total number of obtained points will be divided by 110 to determine the final course grade.

You will receive a penalty of 5 points for each peer review not performed. Significantly different reviews will be subject to investigation. If deemed badly done by the teacher or TA, you will also receive 5 penalty points.

SUPERVISION AND HELP

We use Mattermost for this assignment. Under channel Lab1, you may ask questions to the teacher, TAs, and fellow students. It is wise to ask for help when encountering start-up problems related to loading the data or getting a machine learning platform to execute. Experience teaches that students typically answer within an hour, TAs within a day, and the teacher the next working day. When asking a question to a TA or teacher, your questions may be forwarded to the channel to get answers from fellow students. Important questions and issues may lead to discussions in class.

Lab sessions are Friday's 10:45-12:45 physically at ECHO room C or virtually on gather.town. Please see Brightspace for details.

SUBMISSION AND FEEDBACK

Submit your work in Brightspace, under assignments. Within a day after the deadline, you will receive several (typically two) reports to grade for peer review as well as access to the online peer review form. You have 5 days to complete these reviews. You will then receive the anonymous review forms for your groups report and code.
There is the possibility to question the amount of points given to your work, up to one week after receiving the completed forms. You should do so via a private message to the teacher and TA in Mattermost.