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). ***DO NOT DOWNLOAD THE VIRUS THAT WAS USED TO GENERATE THE DATA UNLESS USING A VM OR OTHER SANDBOX***. 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)

- Download **scenario 10** (https://mcfp.felk.cvut.cz/publicDatasets/CTU-Malware-Capture-Botnet-51/detailed-bidirectional-flow-labels/capture20110818.binetflow) from the CTU-13 datasets (see paper 4 from below resources)
- Remove all background flows from the data
- 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 4 individual tasks should only be performed on CTU-13 scenario 10. One student works on frequent+sketching and the other on minwise+hyperplane LSH.

Frequent task, individual (5 points)

- Use the SPACE SAVING algorithm to estimate the distribution over 3-grams of discretised symbols. Write code for the algorithm, use it to estimate the distribution in one pass (no need to actually stream the data, you may store it in memory, or run every file separately, but do store and load the intermediate results)
- Use a range of number of counters. What are the 10 most frequent 3-grams and their frequencies when approximated? Use the theory to explain any approximation errors you observe

Sketching task, individual (5 points)

- Build code for computing a COUNT-MIN sketch to estimate occurrence counts for the 3-grams. Make sure the hash functions are pairwise independent. Estimate the distribution in one pass (no need to actually stream the data, you may store it in memory, or run every file separately, but do store and load the intermediate results)
- 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? Use the theory to explain any approximation errors you observe

Min-wise locality sensitive hashing task, individual (5 points)

- Implement min-wise locality sensitive hashing 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, 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.
- Compare the run-time of a pair-wise distance computation with one that only considers profiles that end in the same bin. Explain any differences you observe.

Random hyperplane locality sensitive hashing task, individual (5 points)

- Implement locality sensitive hashing using random hyperplanes as explained in the slides and the documents on Brightspace. This can be used to quickly compute Euclidean distance for N-gram profiles. Use the discretization from task 1, build 3-gram profiles for every individual connection (pair of IP-addresses). For this task, the profiles are counts

for every possible 3-gram (subsequence. Use random hyperplanes LSH to map the N-gram profiles to a small set of bins of your choice.
- Compare the run-time of a pair-wise distance computation with one that only considers profiles that end in the same bin. Explain any differences you observe.

The botnet tasks should be performed on CTU-13 scenarios 9-12 (capture 50-53).

Botnet profiling task (5 points)

- Use a sliding window to obtain sequence data for every host in all scenarios considered in paper 4 (with multiple infected hosts), with a length of your choice. Learn an n-gram, state machine, or hidden Markov model from the data of one infected host and match its profile (e.g., n-grams using cosine distance, or probabilities using KL-divergence) with all other hosts from the same scenario. Evaluate how many new infections your method finds and false positives it raises (as in paper 4).

Botnet fingerprinting task (5 points)

- Use the obtained botnet profiles for fingerprinting, i.e., look for the occurrence of an n-gram, transition, or event that does not occur in any benign traffic. Whenever this subsequence occurs, you raise an alarm. Evaluate how many new infections your method finds and false positives it raises (as in paper 4). Compare it to profiling and explain the results.
- Remember to write clear code and explain your results understandably. During peer review, your fellow students will be asked to run and understand your code!
- More text is not always better

Bonus!: Non-sequential machine learning (5 points)

- Study paper 3 and construct a classifier for detecting anomalous behavior in individual NetFlows (every flow is a row, ignoring sequences). Do not forget to study and deal with properties of your data such as class imbalance.
- Evaluate your method in two ways: on the packet level (as in paper 3), and on the host level (as in paper 4). Do you prefer using a sequential model or a classifier for detecting botnets? Explain 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: (i) a Jupyter Python notebook of max. 1000 added words (ii) two individual Jupyter Python notebooks for the individual assignements of max. 300 added words (iii) and the libraries used to run the code other than numpy, scipy, pandas, and scikitlearn. The notebook 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, and the result investigated.*	*0-5 points*

Frequent/Sketching	*Frequent/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.*	*0-5 points*
LSH	*LSH is implemented correctly. The number of bins is set sensibly. The resulting comparison explains differences in run-time and quality.*	*0-5 points*
Profiling	*Advanced sequential model learning is used correctly, one for each host. Profile matching and evaluation are correct.*	*0-5 points*
Fingerprinting	*Fingerprinting is correctly applied. Comparison to profiling is sound and considers both run-time and the kinds of behaviors that can be detected.*	*0-5 points*
Bonus	*Non-sequential machine learning setup is correct, results are sound, and preference is motivated.*	*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. averaging to account for the number of peer reviews. In total 35 points (including bonus) can be obtained in each lab assignment, of which 10 are for the individual parts. In total, 140 points (including bonus) can be obtained in the 4 lab assignments, of which 40 are individual. The total number of obtained points will be divided by 12 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 Lab3, 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.

There is no separate lab session hosted at the university, it is your own responsibility to start and finish on time.

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.