



```
function T(e, t) {  
  if (e) {  
    var i;  
    if (!_saf10650742) {  
      _saf10650742 = 0;  
      return  
    };  
    for (i = e.length - 1;  
        e > -1 && (!e[i] || !t(e[i], i, e));  
        i -= 1) {  
      ;  
    }  
  }  
  if (_saf10650103 == true) {...
```

LSH

297a7fadb6dc7abded32c8d4a4c70ab9355fe70cd3399eca7de827a747099dc0

29 7a 7fad b6 dc 7a bd ed 32 c8 d4 a4 c7 0a b9 35 5fe7 0c d3 39...

74, 8, 23, 43, 10, 114, 1, 125, 31, 57, 4, 13, 21 184, 231, 15, 82, 52, 12, 55, 73, 23...

Neural Network

MALWARE

Malware Classification using Locality Sensitive Hashing and Neural Networks

Master of Science Thesis in Software Engineering

Ludwig Friberg & Stefan Carl Peiser

MASTER'S THESIS 2019:42

Malware Classification using Locality Sensitive Hashing and Neural Networks

LUDWIG FRIBORG & STEFAN CARL PEISER



Department of Software Engineering
Thesis report group 42
CHALMERS UNIVERSITY OF TECHNOLOGY AND UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2019

The authors grant to Chalmers University of Technology the non-exclusive right to publish the work electronically and in a non-commercial purpose make it accessible on the internet. The authors warrant that they are the authors to the work, and warrant that the work does not contain text, pictures or other material that violates copyright law. The authors shall, when transferring the rights of the work to a third party (for example a publisher or a company), acknowledge the third party about this agreement. If the authors have signed a copyright agreement with a third party regarding the work, the authors warrant hereby that they have obtained any necessary permission from this third party to let Chalmers University of Technology store the work electronically and make it accessible on the internet.

LUDWIG FRIBORG
STEFAN CARL PEISER

© LUDWIG FRIBORG, 2019.
© STEFAN CARL PEISER, 2019.

Supervisor: RICCARDO SCANDARIATO, DEPARTMENT OF SOFTWARE ENGINEERING

Company Supervisor: ARNA MAGNUSARDOTTIR, CYREN

Examiner: JENNIFER HORKOFF, DEPARTMENT OF SOFTWARE ENGINEERING

Master's Thesis 2019:42
Department of Software Engineering
Thesis group 42
Chalmers University of Technology and Gothenburg University
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Abstract

In this thesis, we explore the idea of using locality sensitive hashes as input features to a feedforward neural network to perform static analysis to detect JavaScript malware. An experiment is conducted using a dataset containing 1.5M evenly distributed benign and malicious samples provided by the anti-malware company Cyren, which is the industry collaborator for this thesis. Four different locality sensitive hashing algorithms are tested and evaluated: Nilsimsa, ssdeep, TLSH, and SDHASH. The results show a high prediction accuracy of 98.05% and low false positive and negative rates of 0.94% and 2.69% for the best performing models. These results show that LSH based neural networks are a competitive option against other state-of-the-art JavaScript malware classification solutions.

Keywords: locality sensitive hashing, static analysis, malware detection, artificial neural networks, machine learning, feature extraction

Acknowledgements

We would like to thank Cyren for all the help and support, especially Arna Magnúsardóttir for sharing her domain expertise. The knowledge and resources they have given us in form of access to data and computing power were essential to this project. We would also like to thank Riccardo Scandariato our supervisor and Jennifer Horkoff our examiner for providing valuable feedback.

Ludwig Friborg, Stefan Carl Peiser, Gothenburg, June 2019

Contents

List of Figures	x
List of Tables	xi
1 Introduction	1
2 Background	3
2.1 JavaScript malware	3
2.2 Obfuscation	5
2.3 JavaScript classification	6
2.3.1 Static analysis	6
2.3.2 Dynamic analysis	6
2.4 Locality sensitive hashing	7
2.4.1 Nilsisma	7
2.4.2 TLSH	7
2.4.3 ssdeep	8
2.4.4 SDHASH	9
2.5 Artificial Neural Networks	10
3 Related Work	12
4 Research Methodology	14
4.1 Research questions	14
4.2 System structure	15
4.3 Neural network design and implementation	16
4.3.1 Network training process	18
4.3.1.1 Optimizers and loss function	18
4.3.1.2 Batch sizing	19
4.3.1.3 Number of epochs	19
4.3.2 Other ANN architectures	20
4.4 SDHASH count vectorization	21
4.5 Data selection	21
4.6 Experiment setup	22
4.7 Metrics	23
5 Results	24
5.1 Detection rates	24

5.2	Causes of misclassifications	27
5.2.1	False negatives	27
5.2.2	False positives	28
5.3	Comparison with existing approaches	29
5.4	Answering the research questions	30
5.4.1	Research question 1	30
5.4.2	Research question 2	31
5.4.3	Research question 3	31
5.5	Threats to validity	32
6	Conclusion	33
6.1	Effectiveness of LSH and ML	33
6.2	Practical application	34
6.3	Further research	34
6.4	Closing remarks	35
	Bibliography	I

List of Figures

2.1	Obfuscated code snippet from malicious code	5
2.2	Obfuscated code snippet from CNN.com	5
2.3	Example diagram of an ANN with one hidden layer	10
4.1	Example on how to classify a TLSH hash	15
4.2	Abstract view over our network	16
4.3	ReLU function and its plot	17
4.4	Sigmoid function and its plot	18
4.5	Count Vectorization example	21
5.1	Results in relation to dataset size	26
5.2	CoinHive example	28

List of Tables

2.1	Basic terminology for JavaScript malware	4
4.1	Neural network composition (where L is length of input vector) . . .	16
5.1	Results from 5-fold cross-validation experiment	25
5.2	Top 10 most common false negative categories, ordered by percentage of occurrences	27
5.3	Performance indicators gathered from related works and our best models	30

1

Introduction

Software Engineering encompasses a lot of different research fields, from finding the best way to write requirements to static code analysis. This thesis will focus on exploring a new method to perform static analysis to classify script files, or more explicitly JavaScript files where a JavaScript file's locality sensitive hash will be used as a feature to predict whether it is malicious or benign.

The method proposed in this thesis entails combining locality sensitive hashing with machine learning. Locality sensitive hashing (*LSH*) is a family of dimensionality reducing algorithms which are widely used in different fields like computer vision, recommendation systems, and more. In this thesis, LSH algorithms will be applied to static code to produce a new representation of the original script file, which will be more applicable for statistical learning.

Locality sensitive hashing has potentially many real-world applications where one of them is in the field of malware identification. There are two main approaches used to identify malware: static analysis and dynamic analysis. Static code analysis is a form of analysis where analysis is done on the malware (source code or compiled executable) directly without executing it, and the other is dynamic analysis where the malware is executed, and its behaviour monitored and judged. Static analysis is preferred as it minimises risks of malware spreading as well as potentially being a quicker way to classify files compared to dynamic analysis. Static analysis might be preferred when analysing large datasets of scripts since some might not be easy to execute correctly. Dynamic analysis in comparison can be harder to apply when analysing large datasets or unknown files since it requires sandbox environments or emulation, certainly in the circumstance of malware detection.