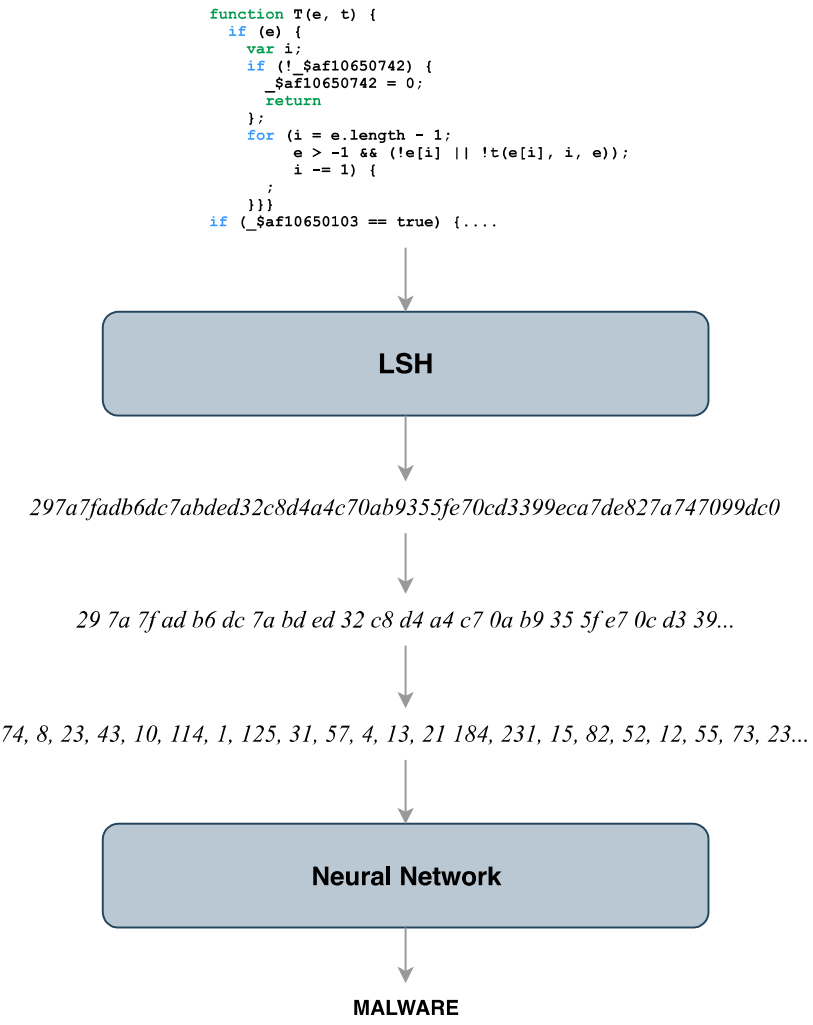
CHALMERS
UNIVERSITY OF TECHNOLOGY



Malware Classification using Locality Sensitive Hashing and Neural Networks

Master of Science Thesis in Software Engineering

Ludwig Friborg & Stefan Carl Peiser

Department of Software Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY & UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2019

Master's thesis 2019:42

Malware Classification using Locality Sensitive Hashing and Neural Networks

LUDWIG FRIBORG & STEFAN CARL PEISER

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Software Engineering
Thesis report group 42
Chalmers University of Technology and University of Gothenburg
Gothenburg, Sweden 2019

LUDWIG FRIBORG
STEFAN CARL PEISER

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

List of Tables

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.