Malware Behavior Analysis Acceleration based on Graph Neural Networks

Yi-Hsien Chen, Steven Lin, Szu-Chun Huang, Chun-Ying Huang

Our Team

Yi-Hsien Chen

Ph.D. candidate in the Department of Electrical Engineering, National Taiwan University (NTU)

Steven Lin

Product Developer @ Synology SIRT

Szu-Chun Huang

Graduate student at the National Yang-Ming Chiao Tung University (NYCU)

Chun-Ying Huang

Professor at the Department of Computer Science, National Yang Ming Chiao Tung University (NYCU)

01

Introduction

Problem Statement & Solution

Problem Statement

Evolving Malware

Millions of new malware samples appear monthly.

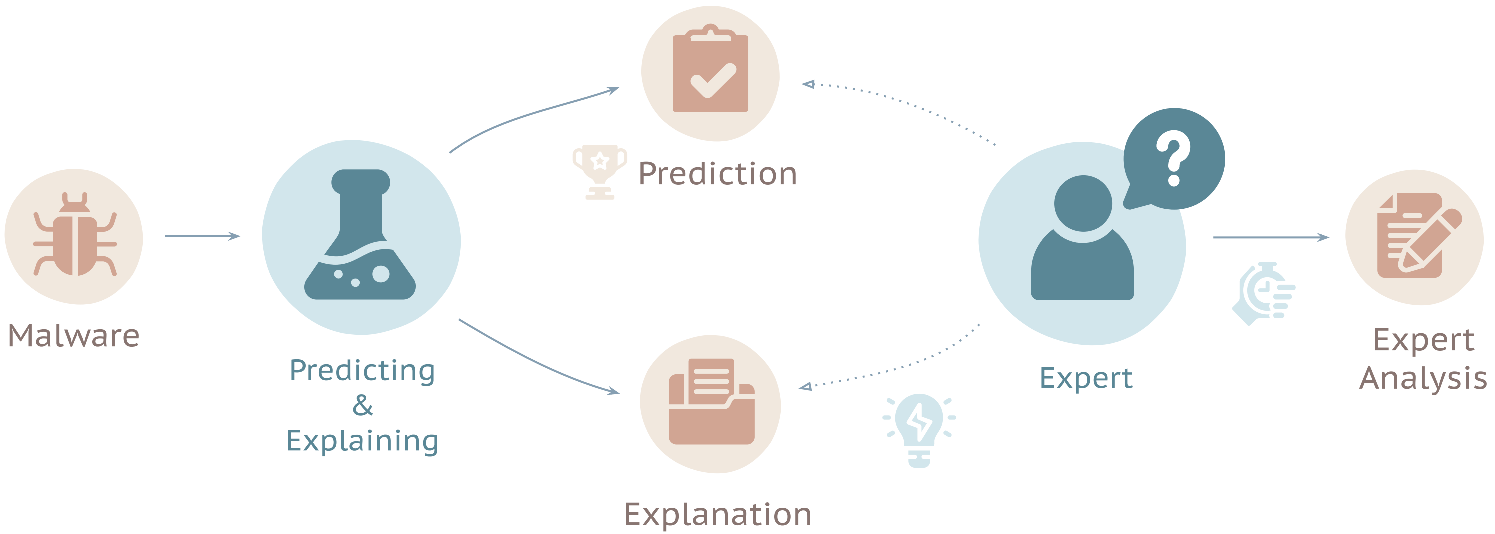
Slow Analysis

In-depth sample analysis is a time-consuming task.

Low Explainability

Models achieve high accuracy but provide no explanations.

Solution



Malware
Predicting & Explaining
Prediction
Explanation
Expert
Expert Analysis

02

Implementation

System Architecture & Methodology

Background

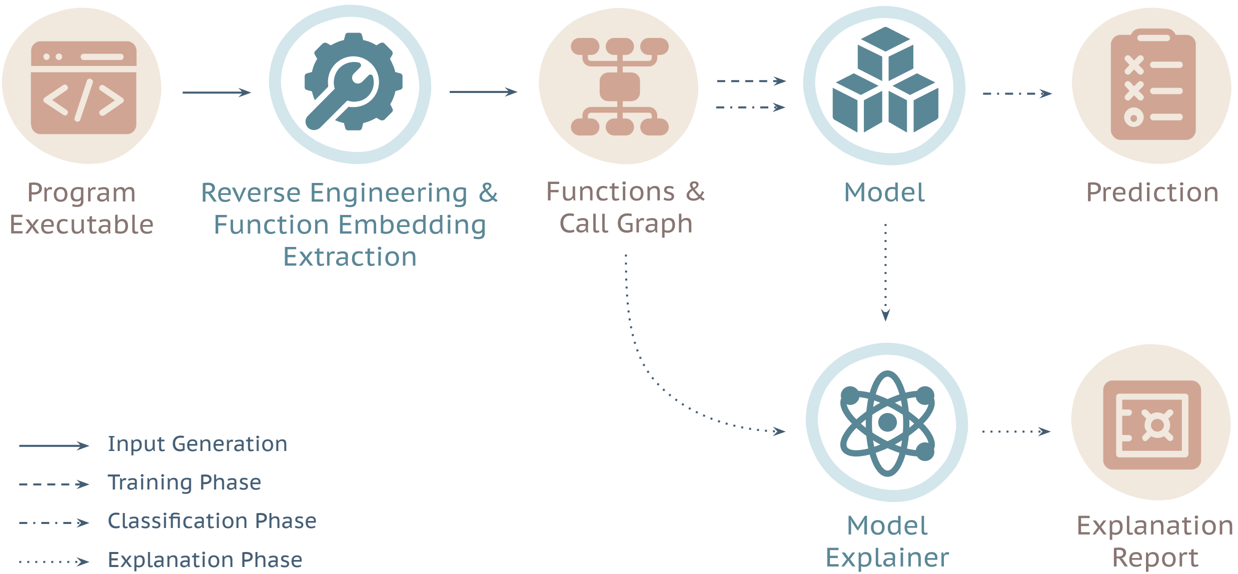
Representation Vector (Embedding)

Low dimension vector transferred from high dimension input containing original input characteristics.

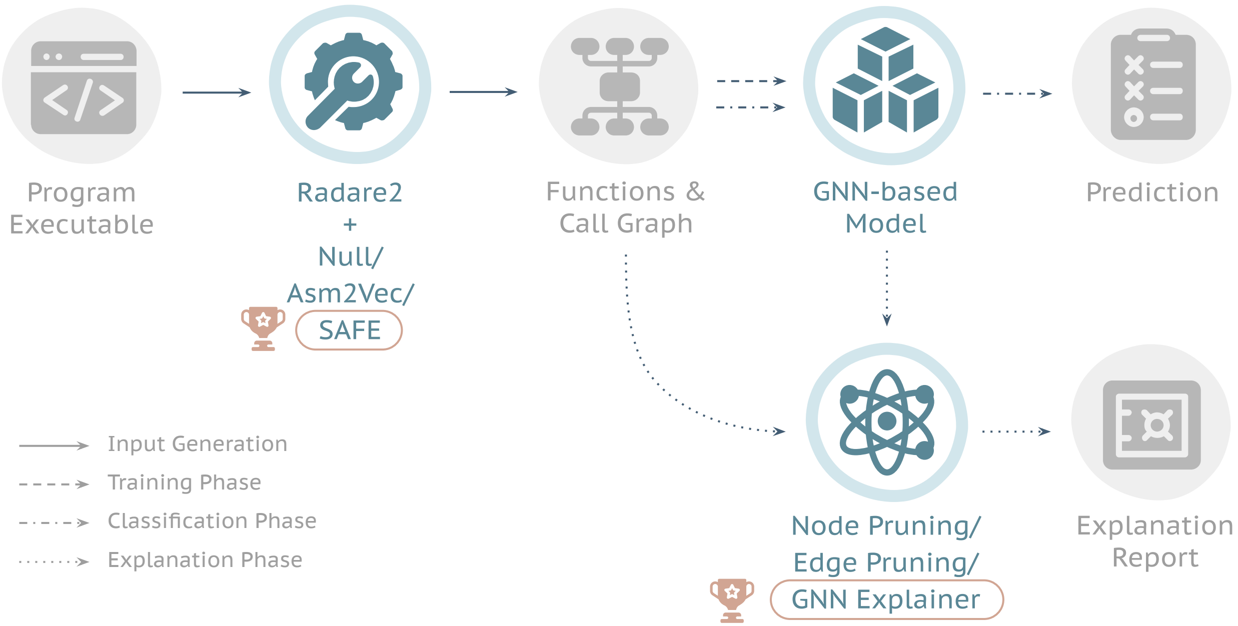
Graph Neural Network

A network that learns nodes and structure information from graph data to obtain graph embedding.

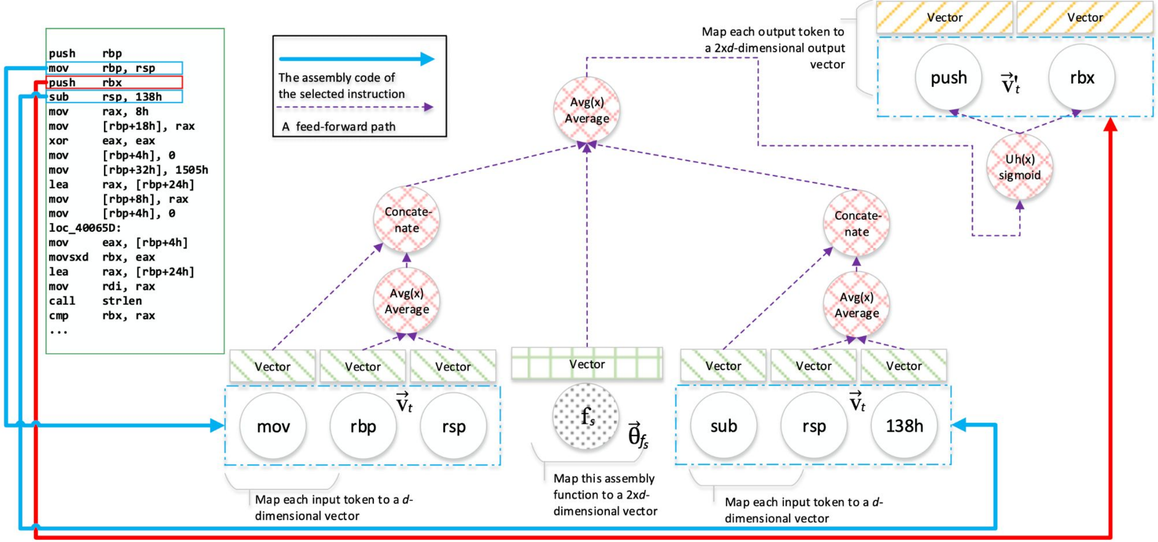
System Architecture



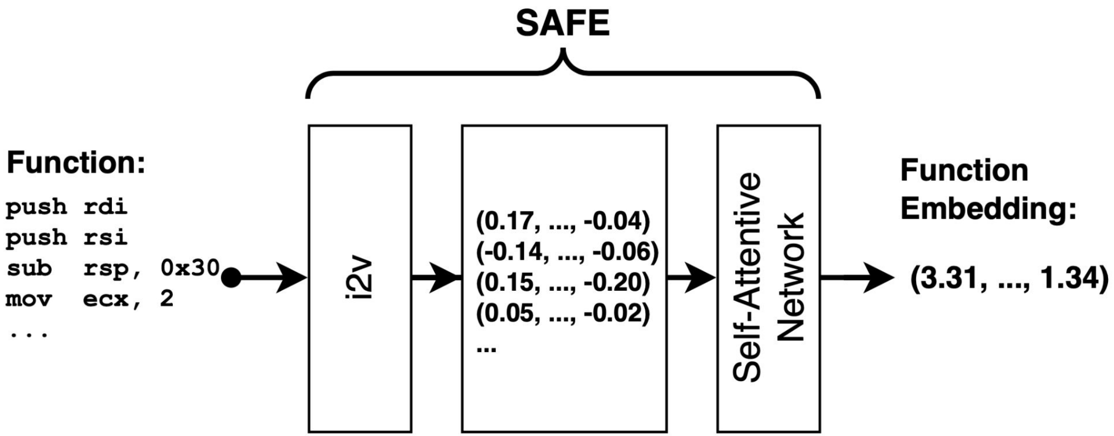
System Architecture



Asm2Vec

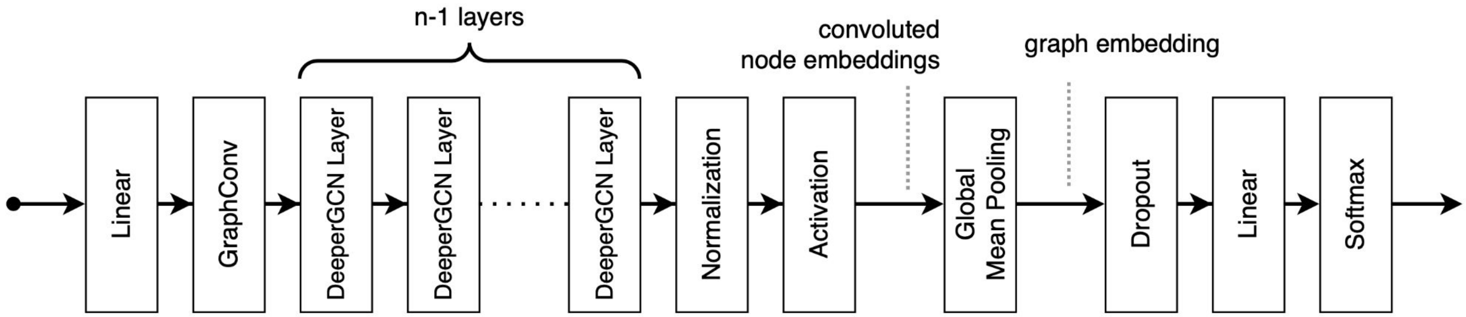

push rbp
mov rbp, rsp
push rbx
sub rsp, 138h
mov rax, 8h
mov [rbp+18h], rax
xor eax, eax
mov [rbp+4h], 0
mov [rbp+32h], 1505h
lea rax, [rbp+24h]
mov [rbp+8h], rax
mov [rbp+4h], 0
loc_40065D:
mov eax, [rbp+4h]
movsxd rbx, eax
lea rax, [rbp+24h]
mov rdi, rax
call strlen
cmp rbx, rax
...
The assembly code of the selected instruction
A feed-forward path
Map each output token to a 2xd-dimensional output vector
Vector
Vector
push
rbx
Uh(x) sigmoid
Avg(x) Average
Concate-nate
Avg(x) Average
Concate-nate
Avg(x) Average
Vector
Vector
Vector
Vector
Vector
Vector
Vector
mov
rbp
rsp
sub
rsp
138h
Map each input token to a d-dimensional vector
Map this assembly function to a 2xd-dimensional vector
Map each input token to a d-dimensional vector

SAFE

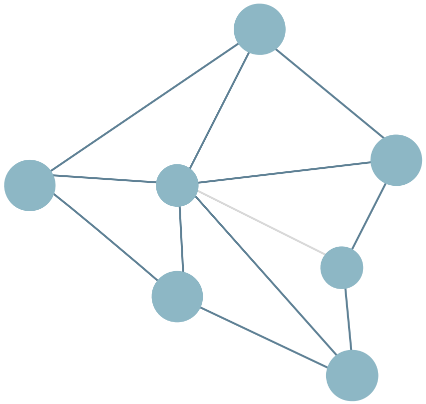


SAFE
Function:
push rdi
push rsi
sub rsp, 0x30
mov ecx, 2
...
i2v
(0.17, ..., -0.04)
(-0.14, ..., -0.06)
(0.15, ..., -0.20)
(0.05, ..., -0.02)
...
Self-Attentive Network
Function Embedding:
(3.31, ..., 1.34)

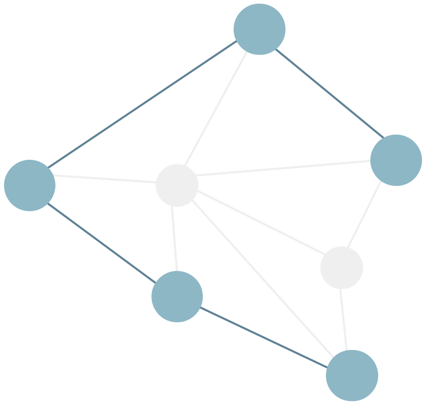
GNN-based Model



Model Explainer

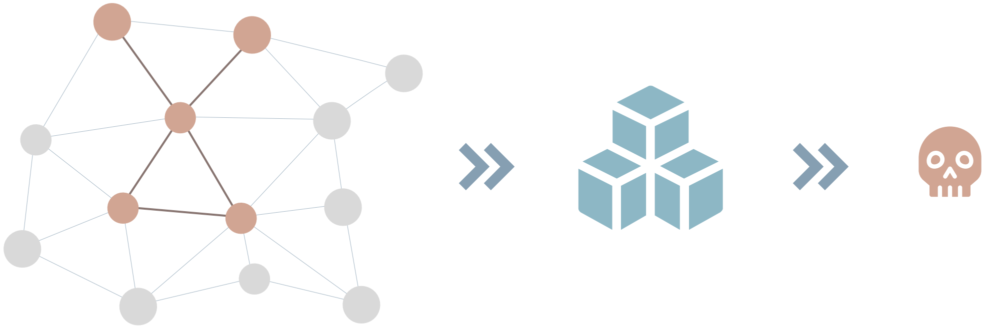


Edge Pruning



Node Pruning

GNN Explainer



$$\max_{G_S} MI\left(Y, (G_S, X_S)\right) = H(Y) - H(Y|G = G_S, X = X_S).$$

03

Evaluation

Experiments Setup & Results

63%

Malicious Dataset

75,257 samples

37%

Benign Dataset

44,953 samples

Experiment Setup

Delete Samples w/o Edge

Samples without any function calls cannot build function call graphs.

Drop Packed Samples

Packed binaries may mislead the model to detect packers.

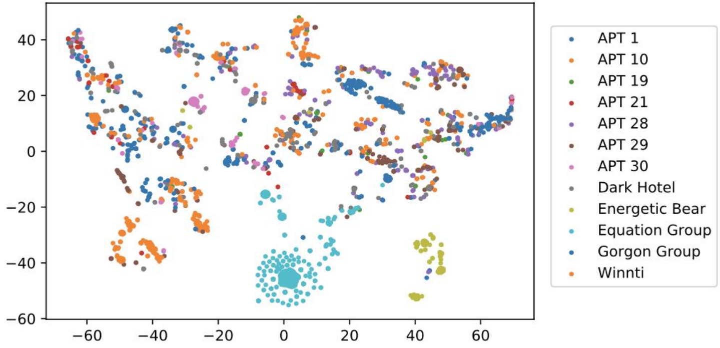
Detection Performance

- Collect 15,000 benign and 15,000 malicious samples
- Split training set and testing set with ratio 8:2
- Use 0.5 as the malware decision threshold for LightGBM and MalConv

Model	Accuracy	Precision	Recall	F1-score
EMBER (LightGBM)	99%	0.989499	0.998370	0.993915
MalConv (NN)	80%	0.844156	0.847596	0.845872
Our Model (GNN)	97%	0.981752	0.965715	0.973632

Handling Unknown Samples

Structure	Model	Recall
LightGBM	EMBER (pre-trained)	0.761257
	EMBER (self-trained)	0.991657
CNN	MalConv (pre-trained)	0.536126
	MalConv (self-trained)	0.533370
GNN	Our Model	0.950094



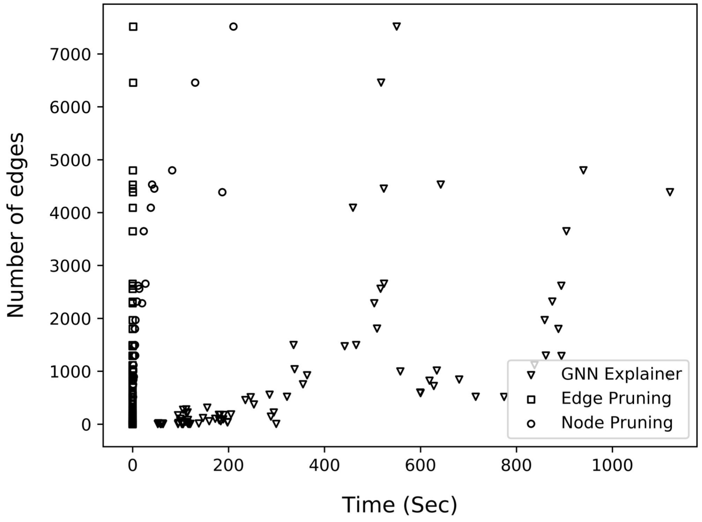
Function Embedding Impact

- Use 209,000 UNIX libraries functions for self-trained models
- Collect 15,000 benign and 15,000 malicious samples
- Split training set and testing set with ratio 8:2

Model	Accuracy	Precision	Recall	F1-score
Null (Zero vector)	67%	0.773491	0.593621	0.633797
Asm2vec (Self-trained)	89%	0.926837	0.863340	0.893472
SAFE (Self-trained)	97%	0.976921	0.969835	0.973338
SAFE (Pre-trained)	97%	0.981752	0.965715	0.973632

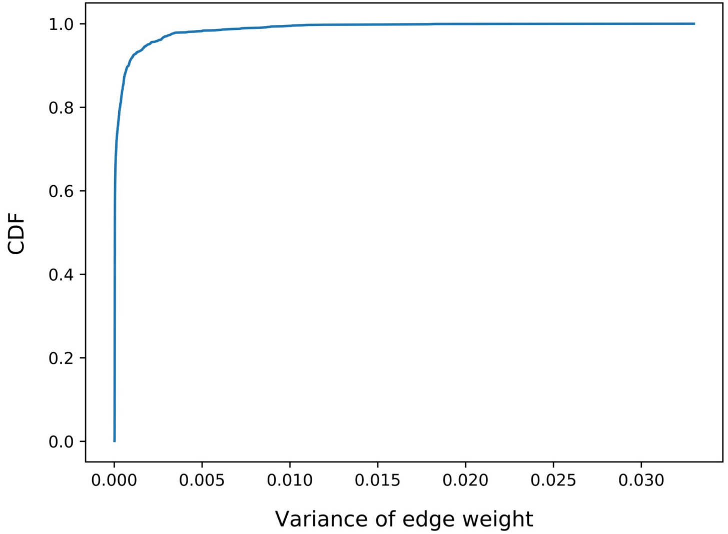
Explaining Efficiency

- Use the same 100 benign files
- Apply pre-trained SAFE embedding
- Generate neural network each time for GNN Explainer



Stability of GNN Explainer

- Use function call graph of AZORult
- Apply pre-trained SAFE embedding
- Explain 100 times for 1645 edges



04

Discussion

Model Explanation Analyses & Case Studies

Are The Explanations Meaningful?

Malware Samples

Phobos Sample

A ransomware with 291 functions and 807 function calls that encrypts files in the victim's computer.

AZORult Sample

An information stealer with 484 functions and 1645 call relations that steals sensitive data from victims.

Malware Samples (Con't)

Equation Sample

A first-stage malware dropper from the Equation APT group, with 319 functions and 684 function calls, trapping users into installing actual malware.

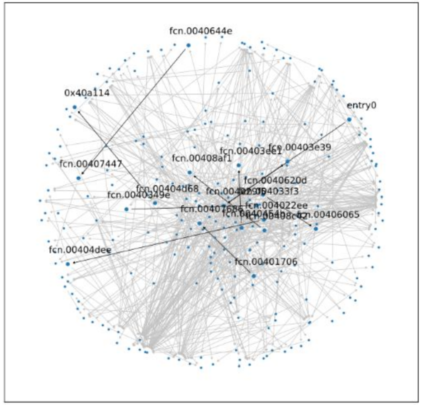
WannaCry Sample

The WannCry malware, with 132 functions and 174 function calls, spreads itself via the SMB service and executes malicious codes to encrypt files on infected systems.

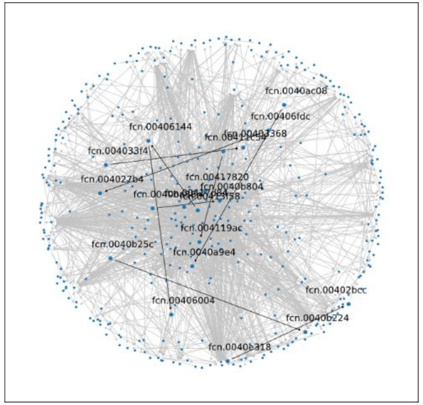
Node Pruning Explanation

(a) Phobos (Graph Pruning/Node). (b) AZORult (Graph Pruning/Node). (c) Equation (Graph Pruning/Node). (d) WannaCry (Graph Pruning/Node).

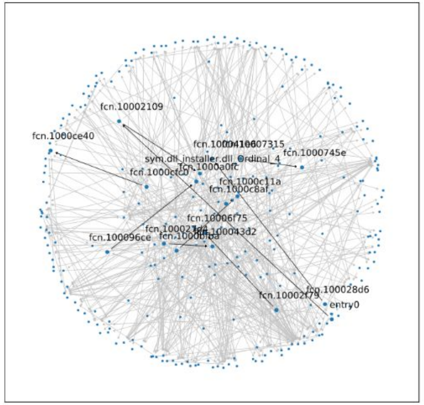
Edge Pruning Explanation



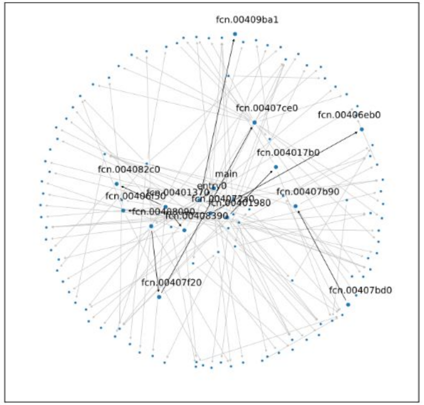
(e) Phobos (Graph Pruning/Edge).



(f) AZORult (Graph Pruning/Edge).

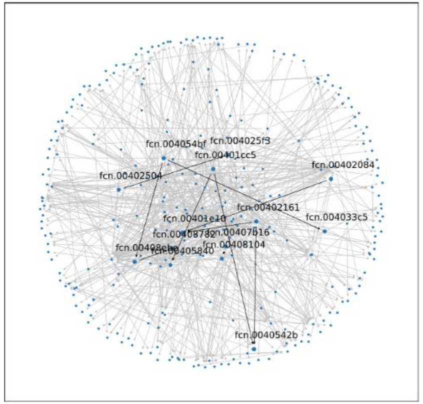


(g) Equation (Graph Pruning/Edge).

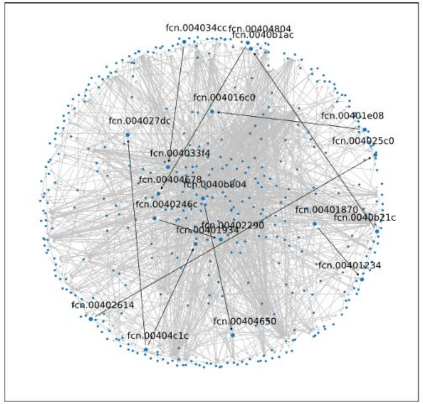


(h) WannaCry (Graph Pruning/Edge).

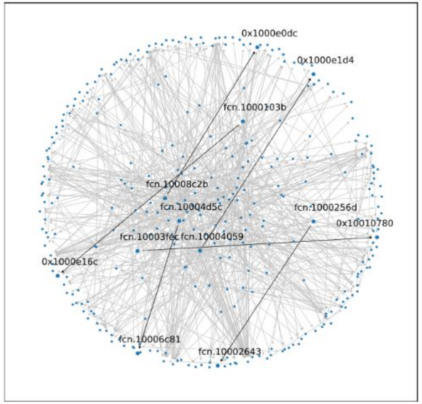
GNN Explainer Explanation



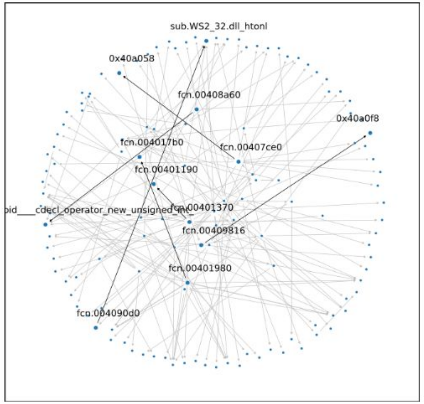
(i) Phobos (GNN Explainer).



(j) AZORult (GNN Explainer).



(k) Equation (GNN Explainer).

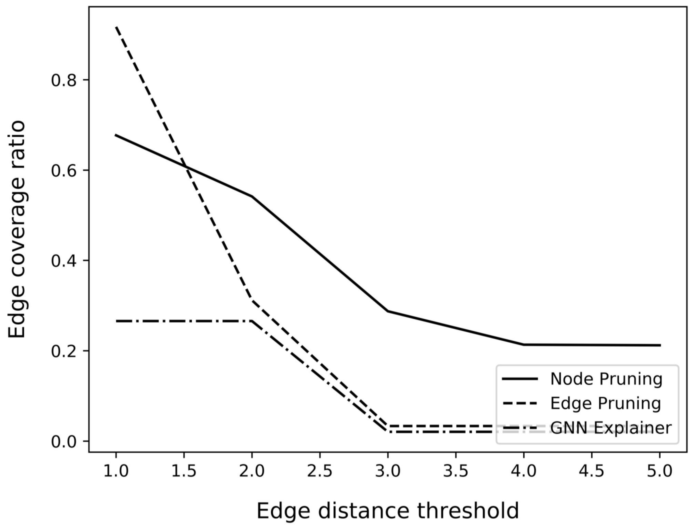


(l) WannaCry (GNN Explainer).

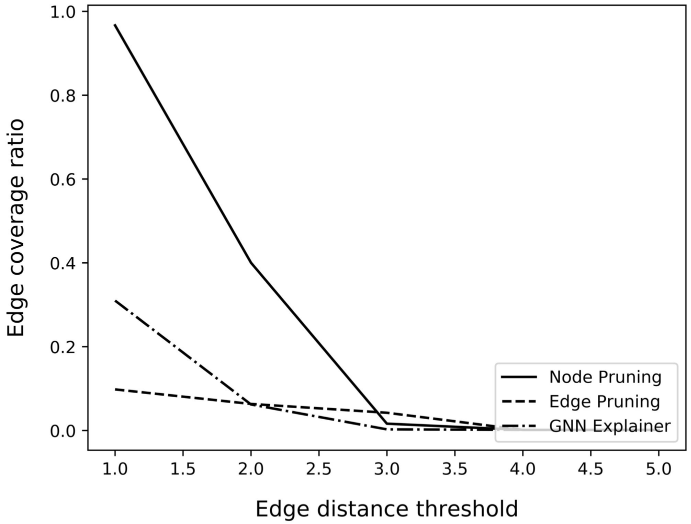
Clues Quality

Quantification for the quality of explanations

Explained Edges Coverage



Lockbit



Phobos

05

Conclusion

Concluding notes & Future Work

Concluding Notes

Model Performance

The proposed malware classification model achieves outstanding performance with an accuracy of 97.0% and a recall rate of 97.6%.

Prediction Explainability

The model explainers can recognize critical graph structures of samples and provide good directions for malware analysis.

Future Work

External API Calls

Embeddings of external API calls may provide more information to the model.

More Embeddings

The performance of other function embedding models is worth evaluating.

Analysis Automation

Automatically classifying functionalities of unknown samples is worth exploring.

Thanks!

CREDITS: This presentation template was created by Slidesgo, including icons by Flaticon, and infographics & images by Freepik