McBoost

Boosting Scalability in Malware Collection and Analysis Using Statistical Classification of Executables

Roberto Perdisci, Andrea Lanzi, Wenke Lee

Outline

- Introduction
- Malware Collection and Analysis
- Packed Malware
- Overview of the McBoost System
- *n*-gram-based Classification
- Application Scenario
- Experimental Results
- Conclusion

Introduction

- Malicious Software (Malware) poses a significant threat to the Internet
- Malware can turn a machine into a spam sender, phishing website, launchpad for DoS attacks, etc.
- We need to ***collect*** and ***analyze*** new malware samples in their early propagation
  - generate detection tools

Malware Collection Strategies

- Passive
  - e.g., Honeypots, Spam Traps
  - Need to wait until hit by malware instance
  - Captured executables are malicious
- Active
  - e.g., URL crawling, P2P crawling, Executable “Sniffing”
  - Do not need to wait: actively find suspicious *exe*
  - Captures both (many) benign and (some) malware

Malware Analysis

- Used to understand what the malware does
- And extract a detection signature
  - traditional AV signature
  - behavioral signature
- Usually very slow, even with use of automatic analysis tools
  - we need to run an executable for at least several minutes to understand its behavior

Malware Analysis

- Slow analysis limits the adoption of active malware collection techniques
  - e.g., P2P crawling, executable “sniffing”
- We cannot afford to waste time on a large number of benign *exe*
  - we need a way to filter them out before detailed analysis starts

Packed vs. Non-Packed

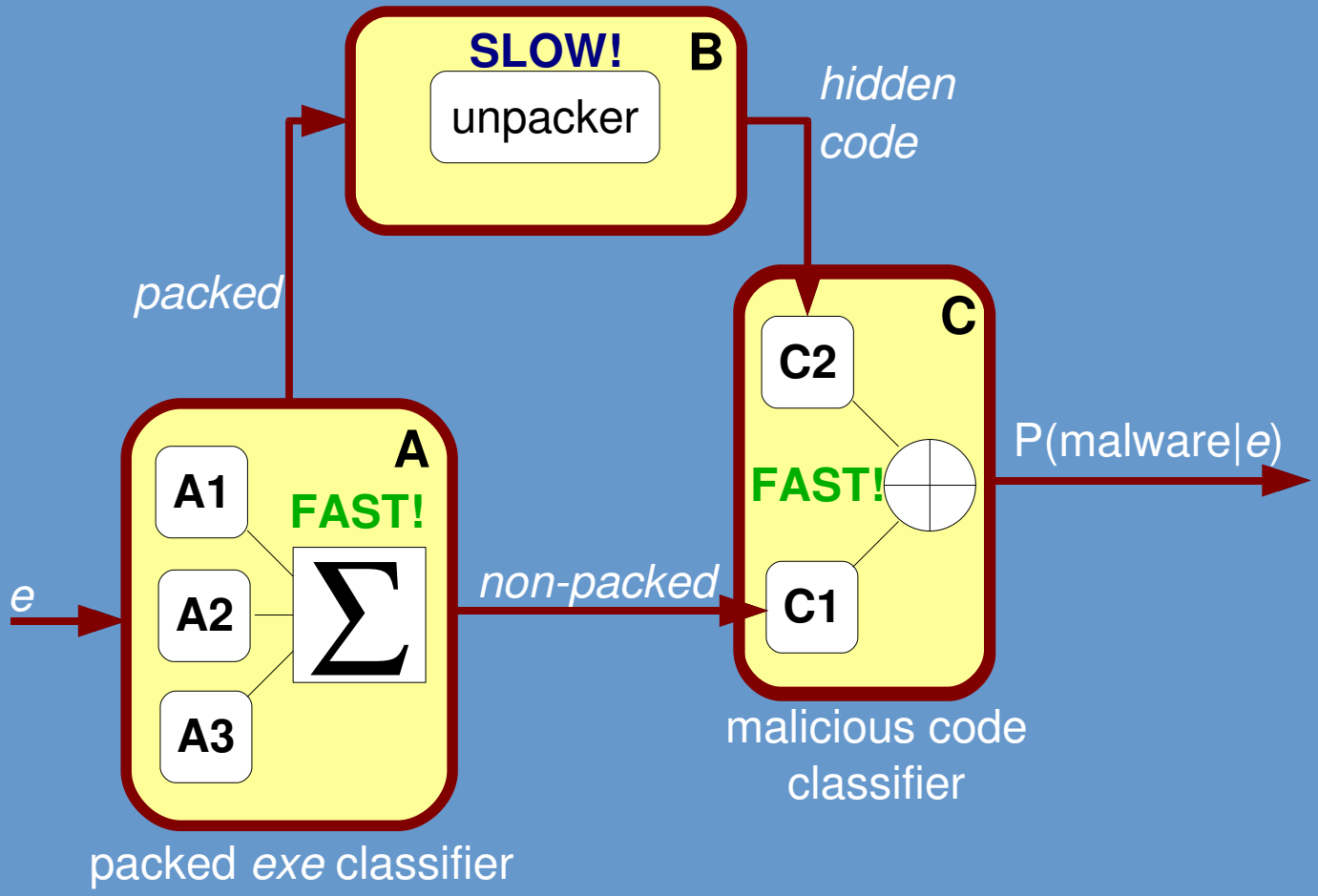
- Packed Malware
  - easy polymorphism
  - the malicious code is hidden (e.g., encrypted)
  - the “real” malware is unpacked and executed on-the-fly

- Most Malware are Packed (80-90%)
- Some Benign are Packed (<10%?)
- Previous related works do not take this into account
  - only distinguish between *malware* and *benign*
  - classifier may make lots of mistakes
    - non-packed->benign (misses non-packed malware)
    - packed->malware (packed benign cause FP)

McBoost's Goal

- New fast statistical malware detection tool
  - classifies *exe* files in PE format
  - takes the prevalence of Packed Malware into account
  - splits classification into two steps
    - packed vs. non-packed PE *exe*
    - malicious vs. benign code
- Given a dataset of benign and malware *exe* as extracted by active collection strategies
  - quickly classify *exe*
  - throw away least suspicious binaries
  - relatively low FP and FN

Overview of McBoost



Errata: In the official printed version **C1** and **C2** are erroneously switched! Please accept our apologies.

n-gram-based Classifiers

- An executable file can be seen as a binary string *s*
- *n*-gram = *n*-bytes-long substring of *s*
- Assume we have 2 classes: *positive*, and *negative* (e.g., malicious and benign code)
  - we extract all the possible n-grams
  - select *M* most discriminant *n*-grams
  - classifier uses the presence or absence of each of the most discriminant *n*-grams as features to make a decision

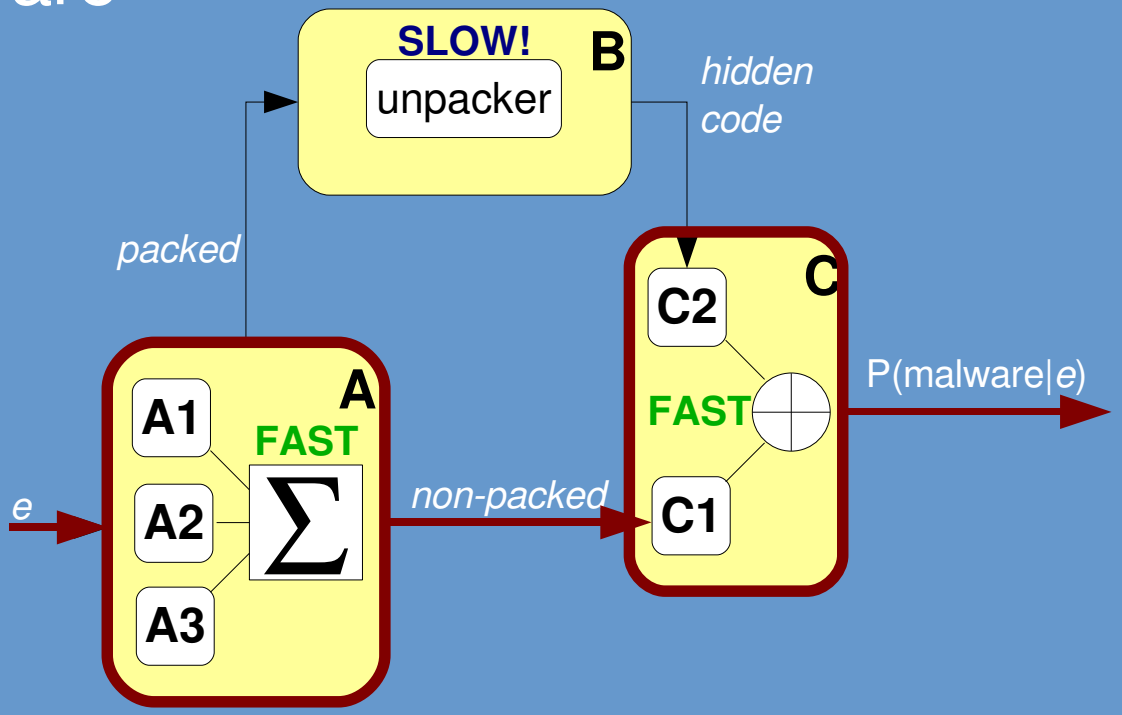
Intuition behind *n*-gram analysis

- Packed vs. non-packed
  - packing “encrypts” the original code
  - alters the distribution of n-grams
- Malicious vs. benign code
  - malicious code may use sequences of instructions that are not common in benign *exe* [Bilar. BH 2008]
  - this translates into some n-grams that appear with different freq in malware compared to benign
  - they become “powerful” discriminant features

Application Scenario

- Assume we collect PE files using executable “sniffing”
  - mostly non-packed benign
  - some new malware

- non-packed binaries can be quickly classified
- quickly throw away benign *exe*
- detailed analysis of remaining suspicious binaries

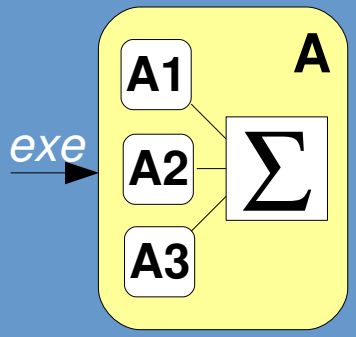


Experiments: Datasets

- 5,586 Malware (Malfease)
  - 2,078 packed according to sig-based detection
  - 3,362 packed according to universal unpacker
  - 146 non-packed
- 2,453 Benign (WinXP+Applications)
  - 2,231 non-packed
  - 27 packed according to sig-based detection
  - 195 “manually” packed using 17 packing tools

Experimental Results

- Packed vs. Non-Packed

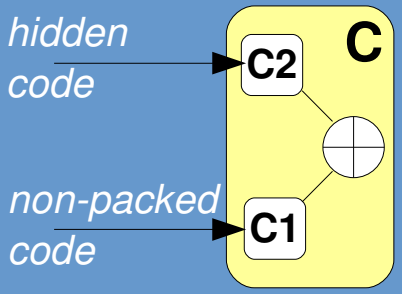


Overall Accuracy = 0.994

AUC = 0.997

Avg Classification Time/exe = 1.03 sec

- Malicious vs. Benign Code



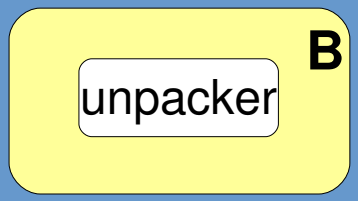
C1: Accuracy = 0.823, AUC = 0.959

C2 : Accuracy = 0.938, AUC = 0.988

Avg Classification Time/*exe* = 0.03 sec

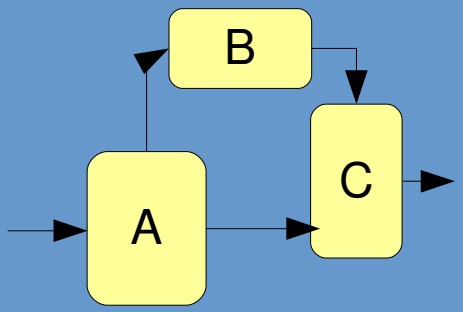
Experimental Results

- Unpacking



Avg Unpacking Time = 5 min

- Overall Malware vs. Benign
  - Output = P(malware|*exe*)



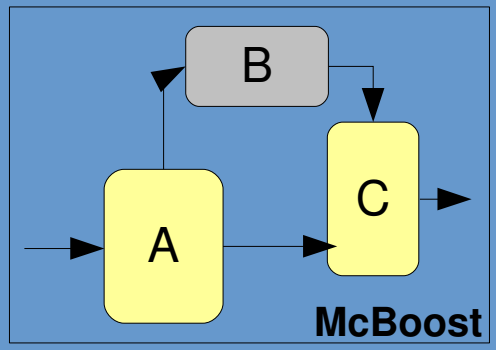
Points on ROC Curve

Threshold	False Positive Rate	Detection Rate	Accuracy
0.902	0	0.617	0.673
0.686	0.010	0.836	0.859
0.500	*0.025*	*0.856*	*0.873*
0.284	0.050	0.881	0.891
0.126	0.100	0.916	0.913
0.029	0.200	0.980	0.953
0.007	0.270	0.993	0.954

  - McBoost's AUC=0.977

Improvement in Scalability

- Assume executable “sniffer” at the edge of enterprise network
  - 85% non-packed benign
  - 2% packed benign
  - 13% new malware (all packed)



- McBoost requires only 13.4% of time compared to running all the samples through malware analysis tools similar to our Universal Unpacker
  - malware analysis needs several minutes per *exe*
  - non-packed benign will be classified in 1.06 sec/*exe* and can be quickly discarded

Conclusion

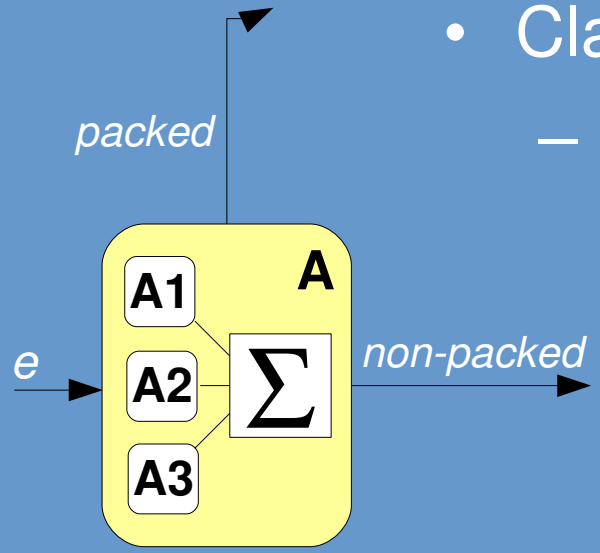
- We introduced McBoost, a new system for statistical classification of *exe*
- McBoost takes *exe* packing into account
- Improves classification in case of mixed set of (mostly) benign and unkown malware
- Quickly filters out least suspicious *exe*
- Boosts scalability of malware collection and analysis

Thank You!

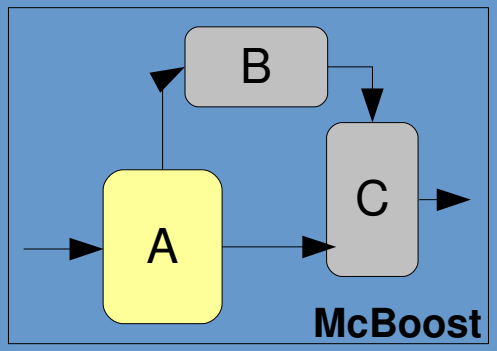
Questions?

perdisci@damballa.com

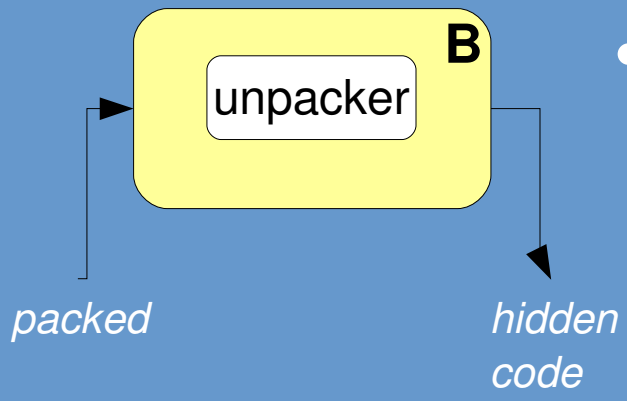
Packed vs. Non-Packed Classifier



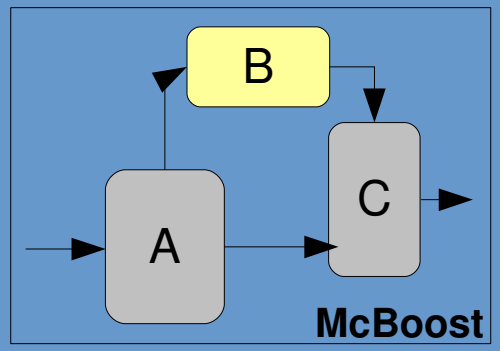
- Classifies packed vs. non-packed
  - A1: uses heuristics [Perdisci et al. PRL 2008]
    - num of “standard” sections in PE
    - num of “non-standard” sections in PE
    - num of IAT entries
    - num of RWX sections
    - byte entropy, etc...
  - A2: n-gram analysis on code section
  - A3: n-gram analysis on entire file
  - A1, A2, and A3 outputs are combine to make a final decision



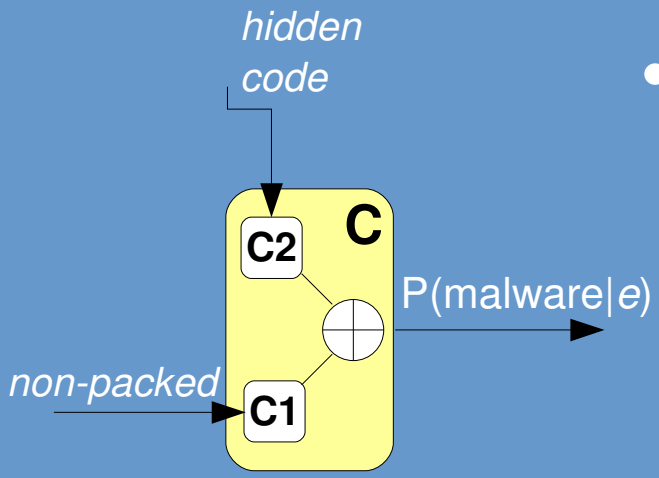
Extracting Hidden Code



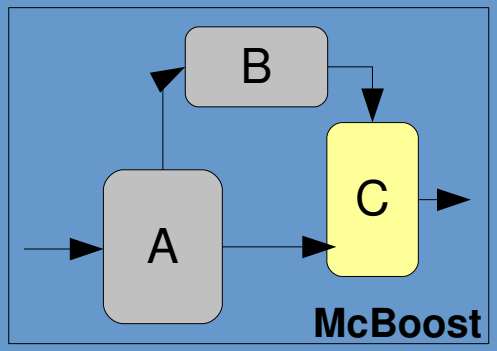
- Universal Unpacker
  - based on dynamic analysis
  - implemented as a plug-in of QEMU
  - executes packed *exe* in isolated environment
  - extracts hidden code using two strategies
    - bbexec (executed basic blocks of last layer)
    - bpage (enitre memory page(s) of last layer)



Malicious Code Classifier

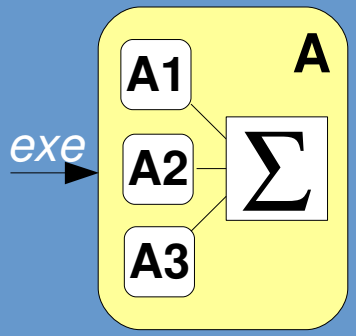


- Classifies Malicious vs. Benign Code
  - C1: classifies the code section of non-packed *exe*
  - C2: classifies hidden code as extracted by the unpacker (B)
  - both classifiers are based on n-gram analysis



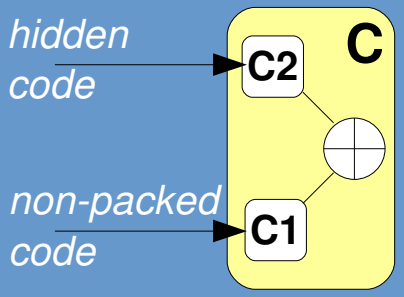
Experimental Results

- Packed vs. Non-Packed



Classifier	Accuracy	FP	DR	AUC
A1 (heuristics)	0.973	0.012	0.958	**0.995**
A2 (n-gram on code)	0.976	0.034	0.987	**0.981**
A3 (n-gram on file)	0.993	0.004	0.990	**0.993**
A (multiple classifiers)	0.994	0.008	0.996	**0.997**

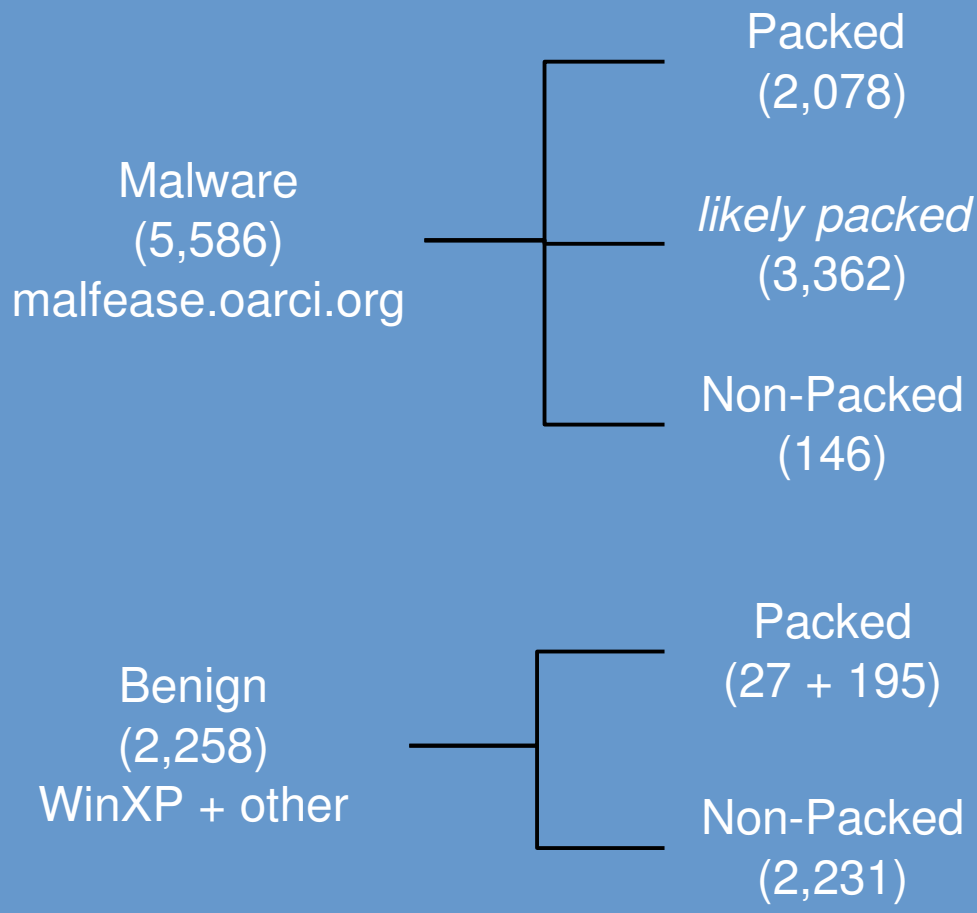
- Malicious vs. Benign Code



Classifier	Accuracy	FP	DR	AUC
C1 (non-packed code)	0.823	0.026	0.772	**0.959**
C2 *bpage* (hidden code)	0.938	0.0	0.937	**0.988**
C2 *bbexec* (hidden code)	0.745	0.112	0.738	**0.901**

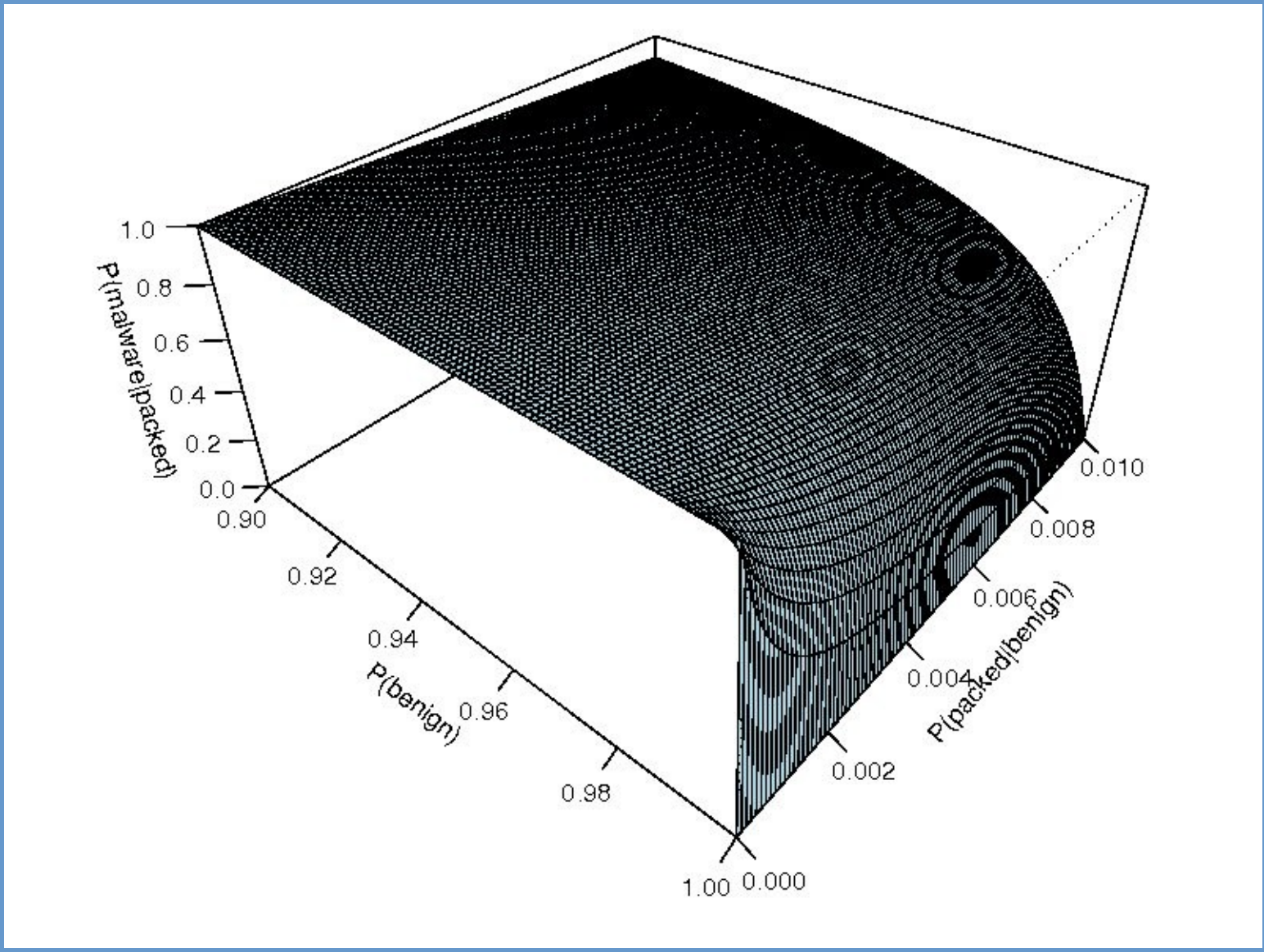
- Time for classification (A+C) = 1.06*s* per *exe*

Experiments: Datasets



- Classifiers are trained on 80% of datasets, and tested on remaining 20%

Appendix


P(malware|packed)
1.0
0.8
0.6
0.4
0.2
0.0
P(benign)
0.90
0.92
0.94
0.96
0.98
1.00
P(packed|benign)
0.000
0.002
0.004
0.006
0.008
0.010