black hat ARSENAL

DECEMBER 9-12, 2024
EXCEL LONDON / UNITED KINGDOM

Packing Box

Improving Detection of Executable Packing

By Alexandre D’Hondt, Alex Van Mechelen, Charles-Henry Bertrand Van Ouytsel and Axel Legay

Outline

Outline

- Problem statement
- Objectives

Problem statement (1)

Packing =

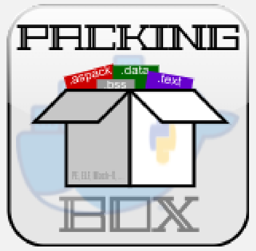
- Set of transformations
- On binary file
- That preserves the original working at runtime

→ Large coverage in scientific literature, yet an open issue

→ Often employed with malware

→ Static detection increasingly relying on Machine Learning

Problem statement (2)



- Dedicated experimental toolkit
- Solves experiments repeatability
- Includes adversarial and unsupervised learning capabilities

[Packing Box: Playing with Executable Packing (BHEU22)]
[Packing-Box: Breaking Detectors & Visualizing Packing (BHEU23)]

Detection challenges (con't) :

- Diversity of packing techniques
- Feature engineering for adding new relevant ones
- Feature selection for getting the most significant ones

- Good features base but limited set
- No focus on packing techniques
- Few works showing economical analysis and categorizing features

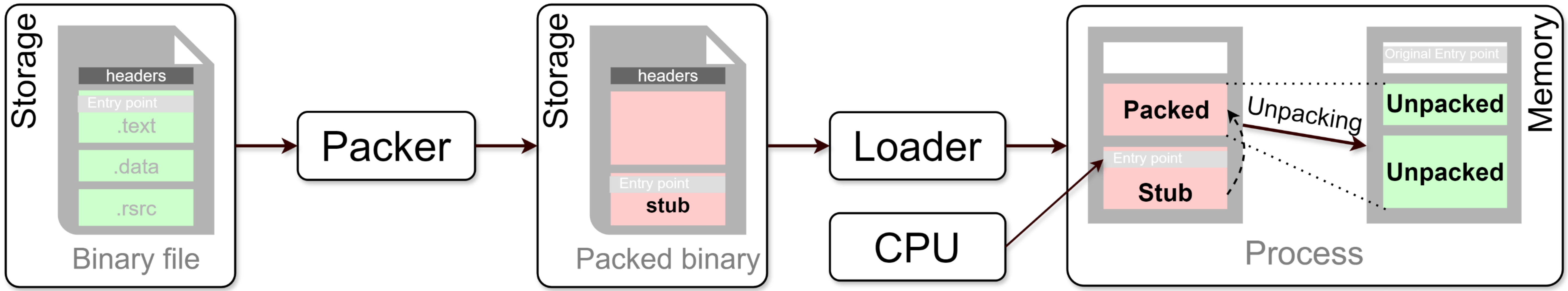
Objectives

1. Extend Packing Box with new feature extraction mechanisms
2. Provide current features set with new relevant features
3. Introduce feature selection methods to identify the most significant

Outline

- Packing / unpacking
- Static detection & features
- Learning Pipeline
- Feature Engineering
- Control Flow Graphs
- Features Selection

Packing / unpacking



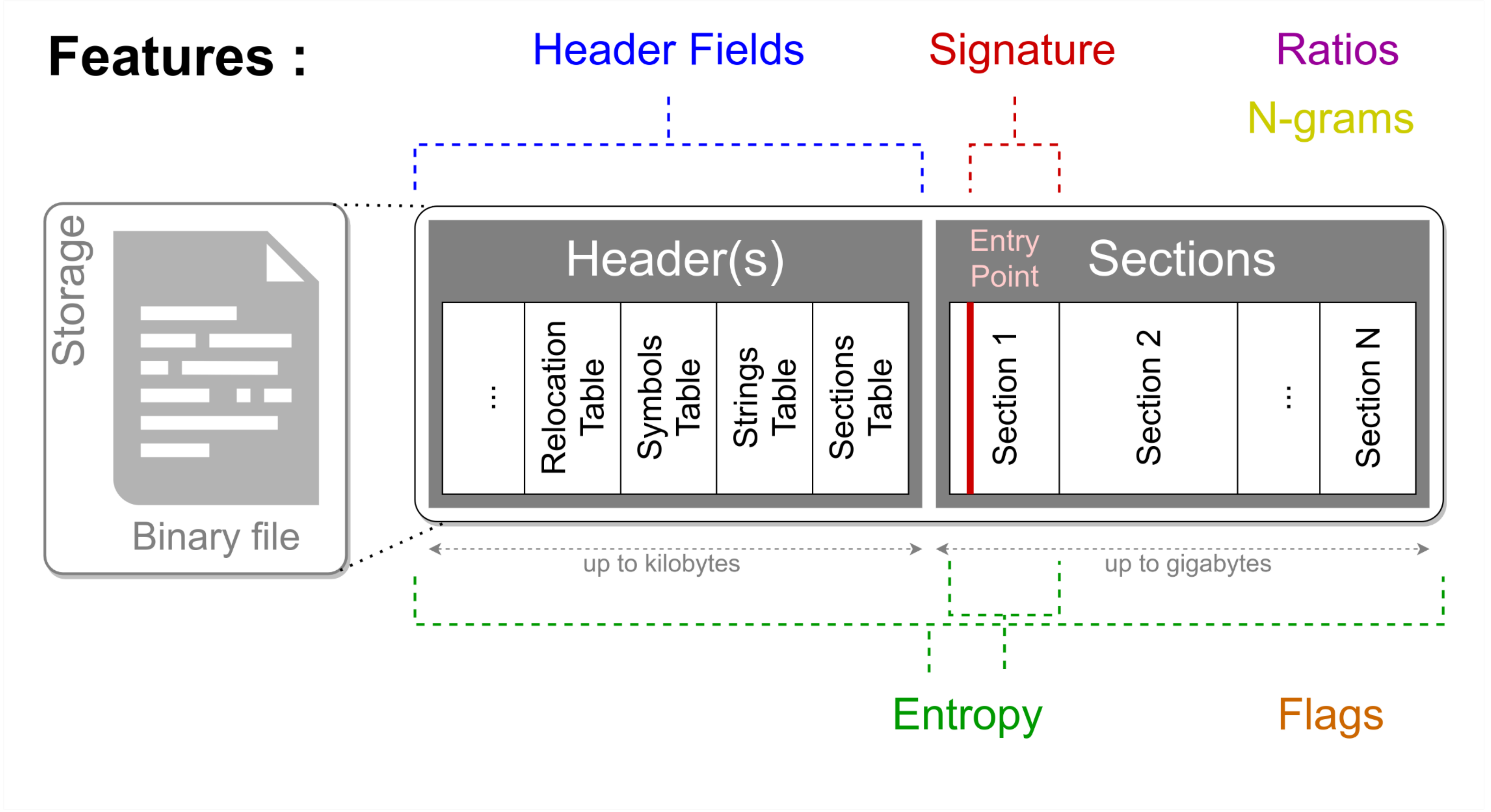
Transformations :

- Compression
- Encryption
- Encoding
- Protection
- Bundling
- Mutation
- Virtualization

Common usage :

- 👍 Size reduction
- 👍 SW piracy prevention / License management
- 👎 Malware

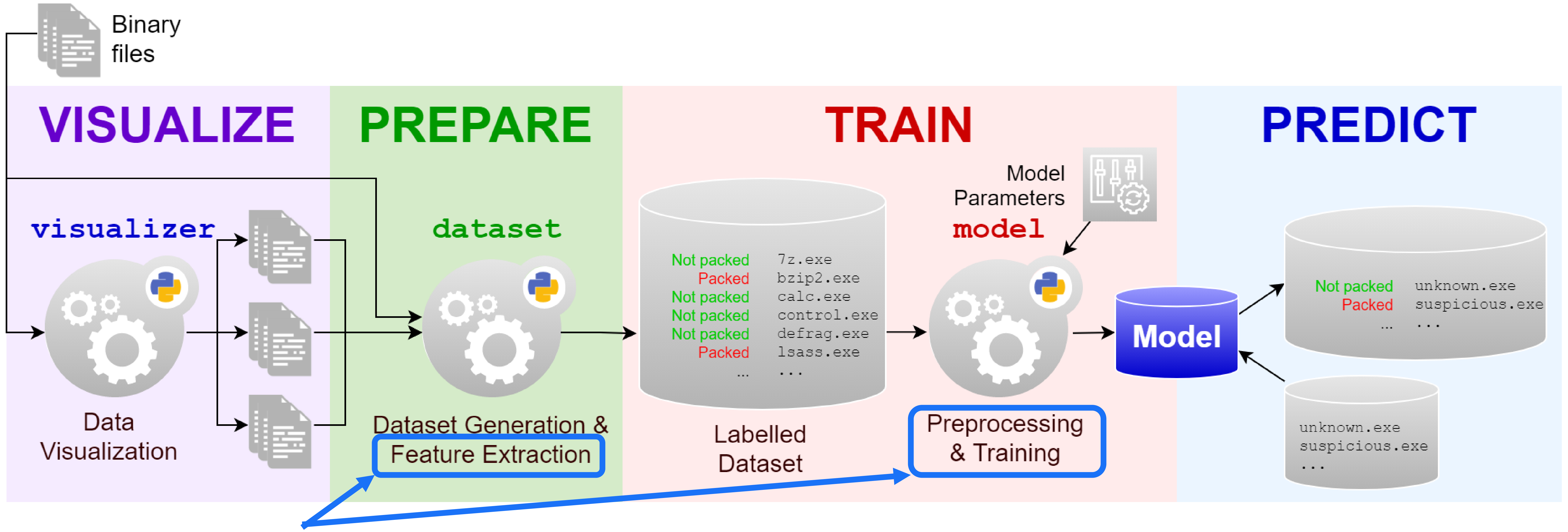
Static detection & features



Static (no execution) :

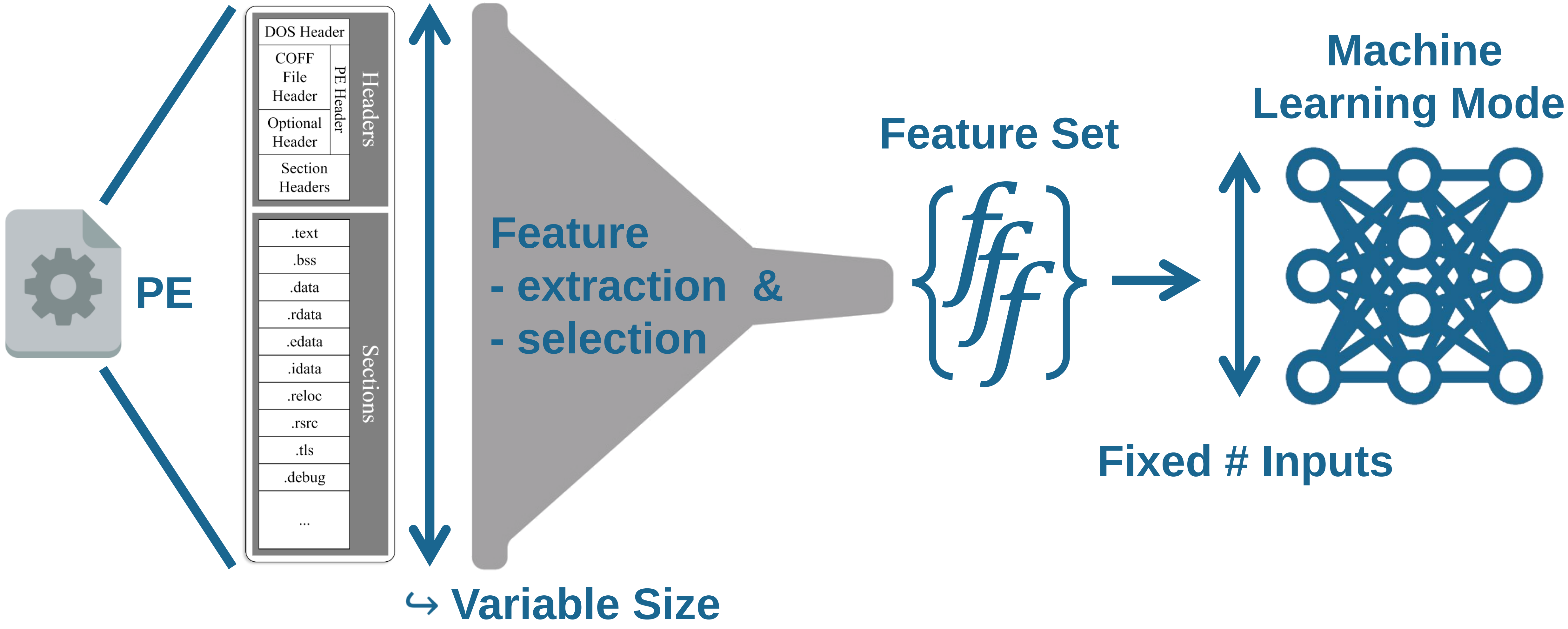
- Entropy threshold
- Pattern matching
- Signatures
- Heuristics
- Disassembly
- Control-Flow Graphs
- …

Learning pipeline

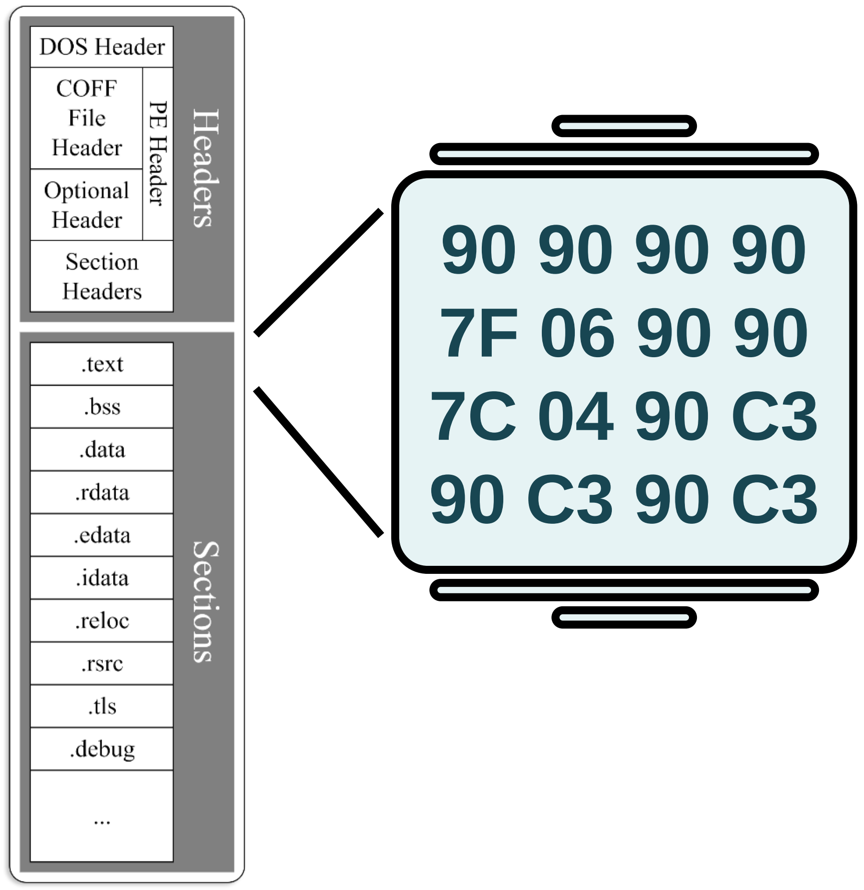


Our focus

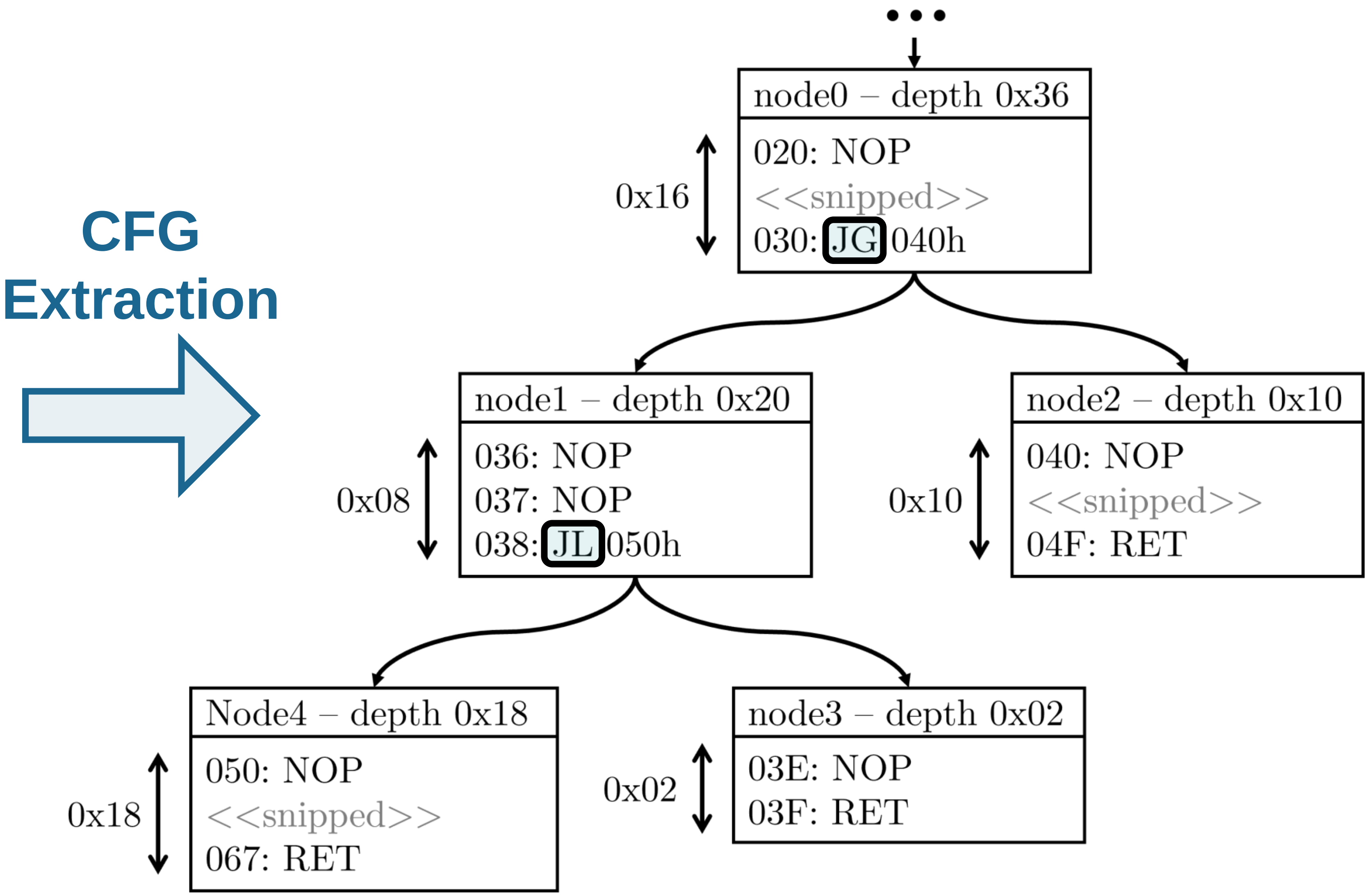
Feature engineering



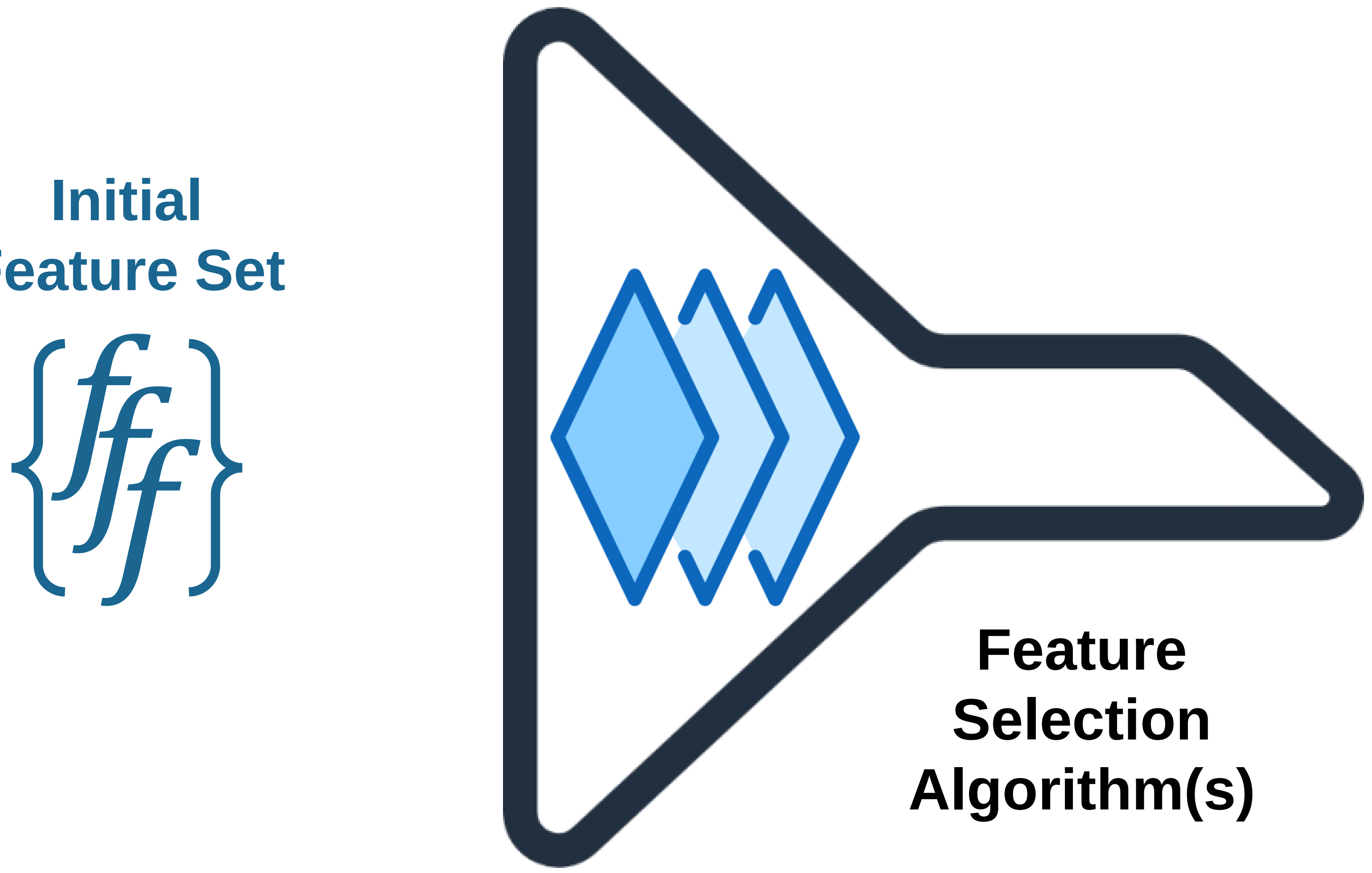
Control Flow Graphs (CFG)



CFG Extraction



Feature selection



Outline

- New requirements
- CFG Feature Extraction Process
- Updated architecture
- Added capabilities
- Getting started

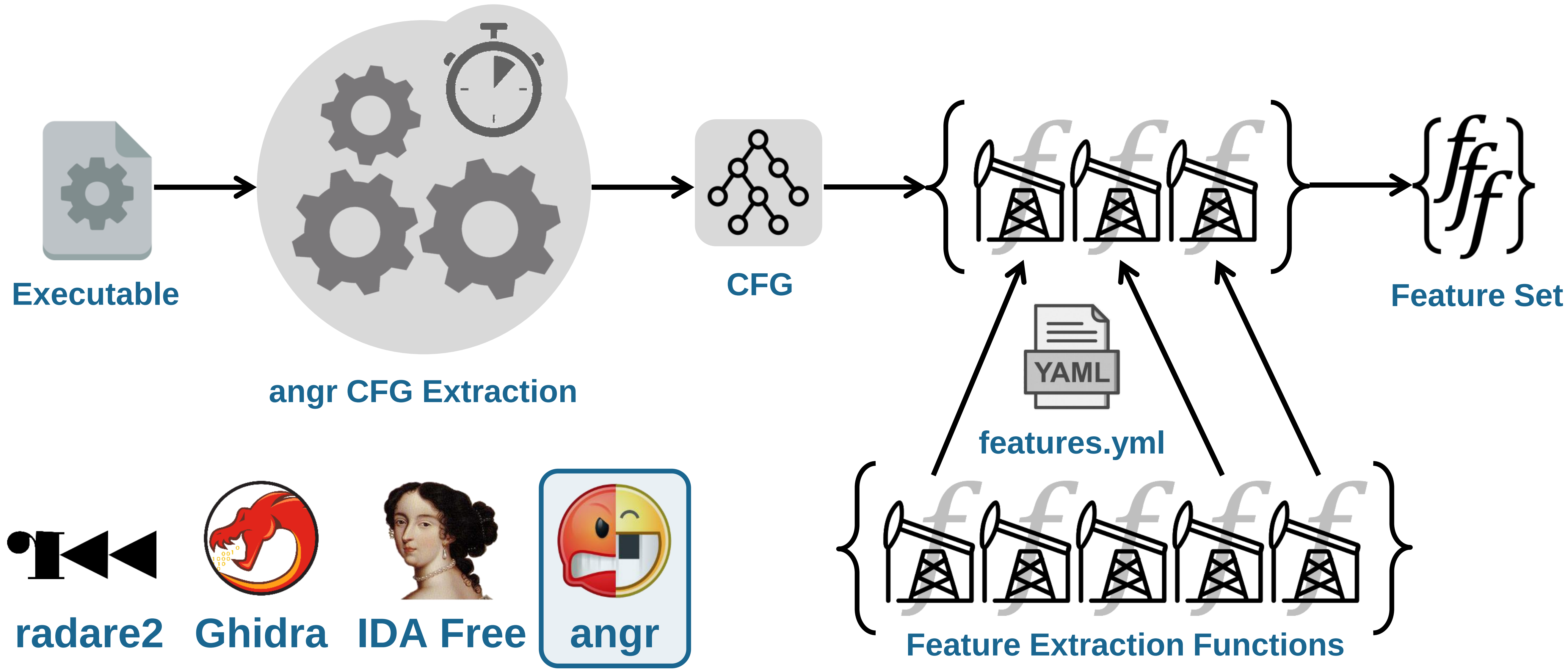
New requirements

CFG Feature Extraction Process

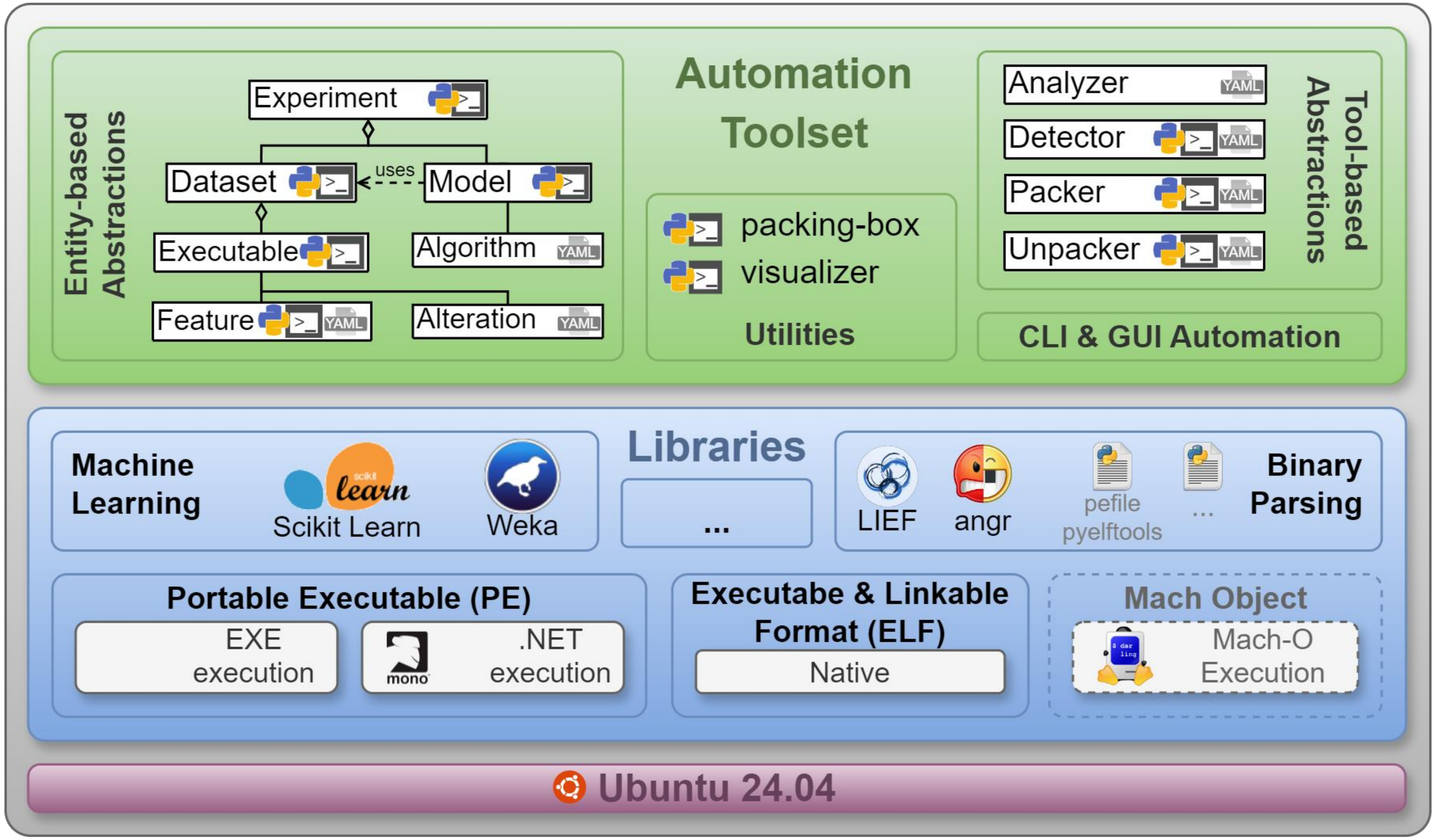
- Open source implementation
- Capability to return complete CFG
- (Easy to integrate into the Packing Box)

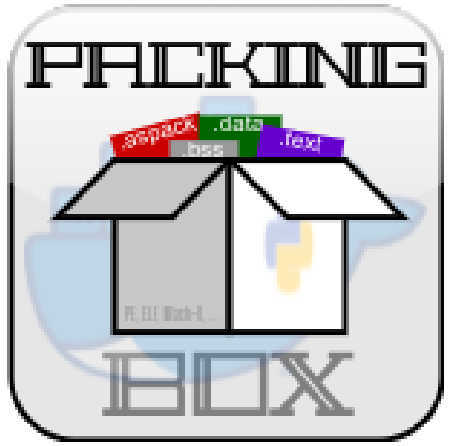
radare2 Ghidra IDA Free angr

CFG Feature Extraction Process



Updated architecture





Added Capabilities

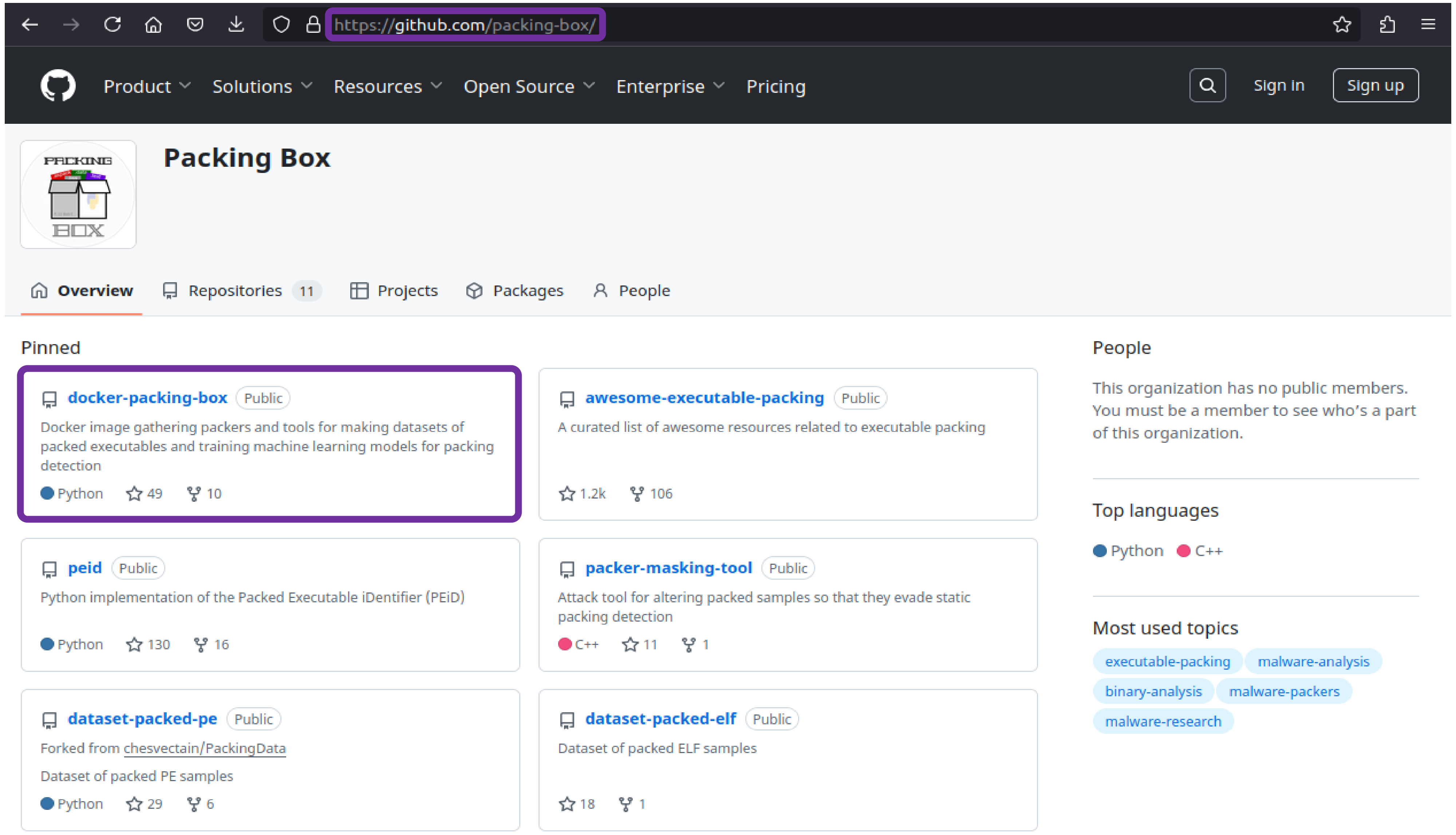
- CFG extraction using angr
- New CFG-based features (from the literature and new ones)
- Multiprocessing for mass feature computation
- Feature tool for interacting with feature sets
- 3 types of feature selection algorithms (filter, embedded, wrapper)
  + possibility to combine them in a layered selection methodology

Getting started (1)

See: Packing Box: Playing with Executable Packing (BHEU22)

<u>Starting point</u> :

1. Open terminal
2. Clone the repo


https://github.com/packing-box/

Product Solutions Resources Open Source Enterprise Pricing Sign in Sign up

Packing Box

Overview | Repositories 11 | Projects | Packages | People

Pinned

docker-packing-box Public
Docker image gathering packers and tools for making datasets of packed executables and training machine learning models for packing detection
● Python ☆ 49 10

awesome-executable-packing Public
A curated list of awesome resources related to executable packing
☆ 1.2k 106

peid Public
Python implementation of the Packed Executable iDentifier (PEiD)
● Python ☆ 130 16

packer-masking-tool Public
Attack tool for altering packed samples so that they evade static packing detection
● C++ ☆ 11 1

dataset-packed-pe Public
Forked from chesvectain/PackingData
Dataset of packed PE samples
● Python ☆ 29 6

dataset-packed-elf Public
Dataset of packed ELF samples
☆ 18 1

People

This organization has no public members. You must be a member to see who's a part of this organization.

Top languages

● Python ● C++

Most used topics

executable-packing malware-analysis binary-analysis malware-packers malware-research

Getting started (2)

See presentation of Black Hat Europe 2022 for basic demonstrations :

Basic Usage

- Build & run Docker image
- Getting help
- Installing items
- Playing with datasets
- Playing with models

Advanced Use Cases

- Model for PE packers
- Visualization of files & models
- Evaluation of detectors

Getting started (3)

See presentation of Black Hat Europe 2023 for more advanced demonstrations :

Adversarial Learning

- Samples inspection
- Performance evaluation of static detectors
- Build and apply alterations

Unsupervised Learning

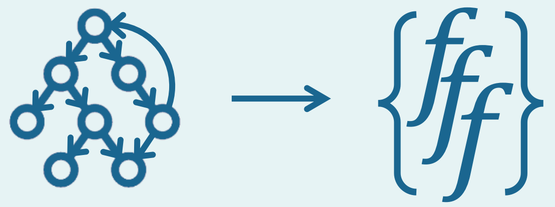
- Exploratory Data Analysis
- Unsupervised model training
- Dataset description

Outline

- CFG features
- Selection methods

CFG Features

Fundamental



- Immediate extraction
- No operations on CFG required

- Not much structural information

→ Lower robustness

Variable Length

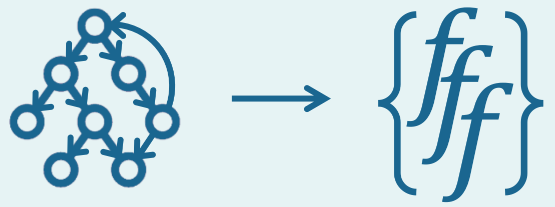
Subgraph

CFG Features

Variable Length

Subgraph

Fundamental

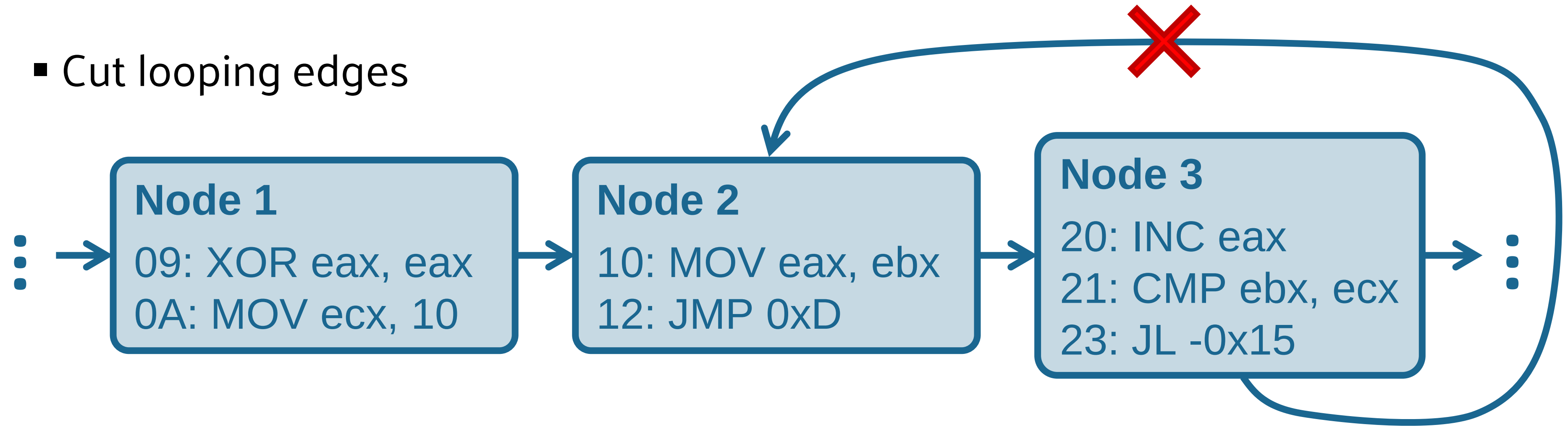


- Immediate extraction
- No operations on CFG required

- Not much structural information
→ Lower robustness

Make CFG acyclic

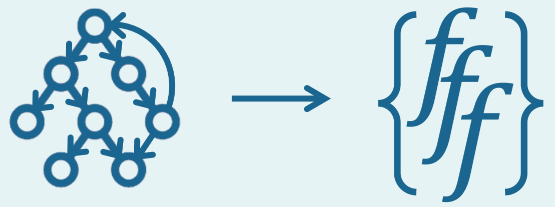
- Cut looping edges



CFG Features

Variable Length | **Subgraph**

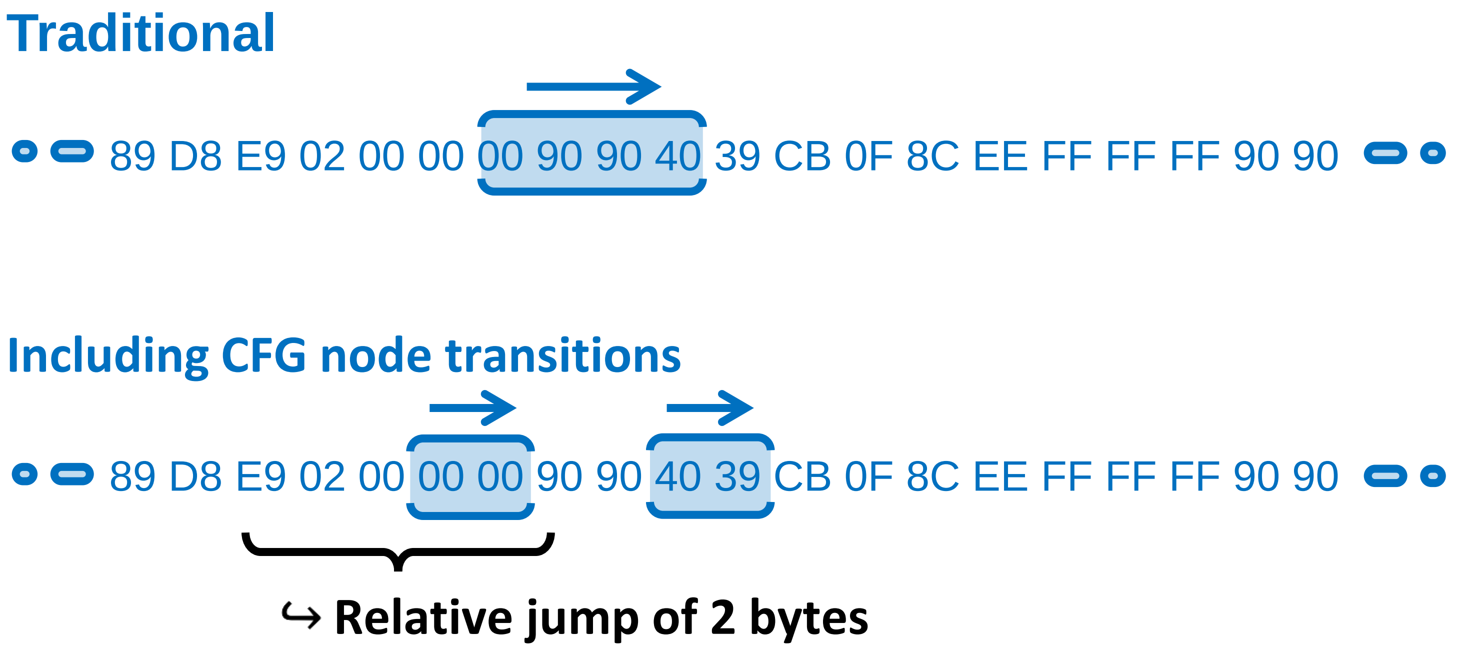
Fundamental



- Immediate extraction
- No operations on CFG required

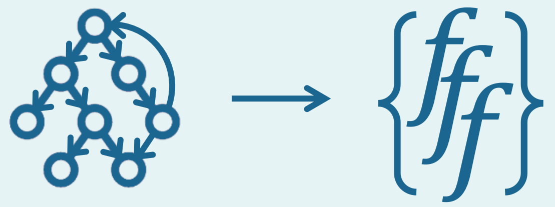
- Not much structural information
→ Lower robustness

N-gram extraction


Traditional
89 D8 E9 02 00 00 00 90 90 40 39 CB 0F 8C EE FF FF FF 90 90
Including CFG node transitions
89 D8 E9 02 00 00 00 90 90 40 39 CB 0F 8C EE FF FF FF 90 90
↪ Relative jump of 2 bytes

CFG Features

Fundamental



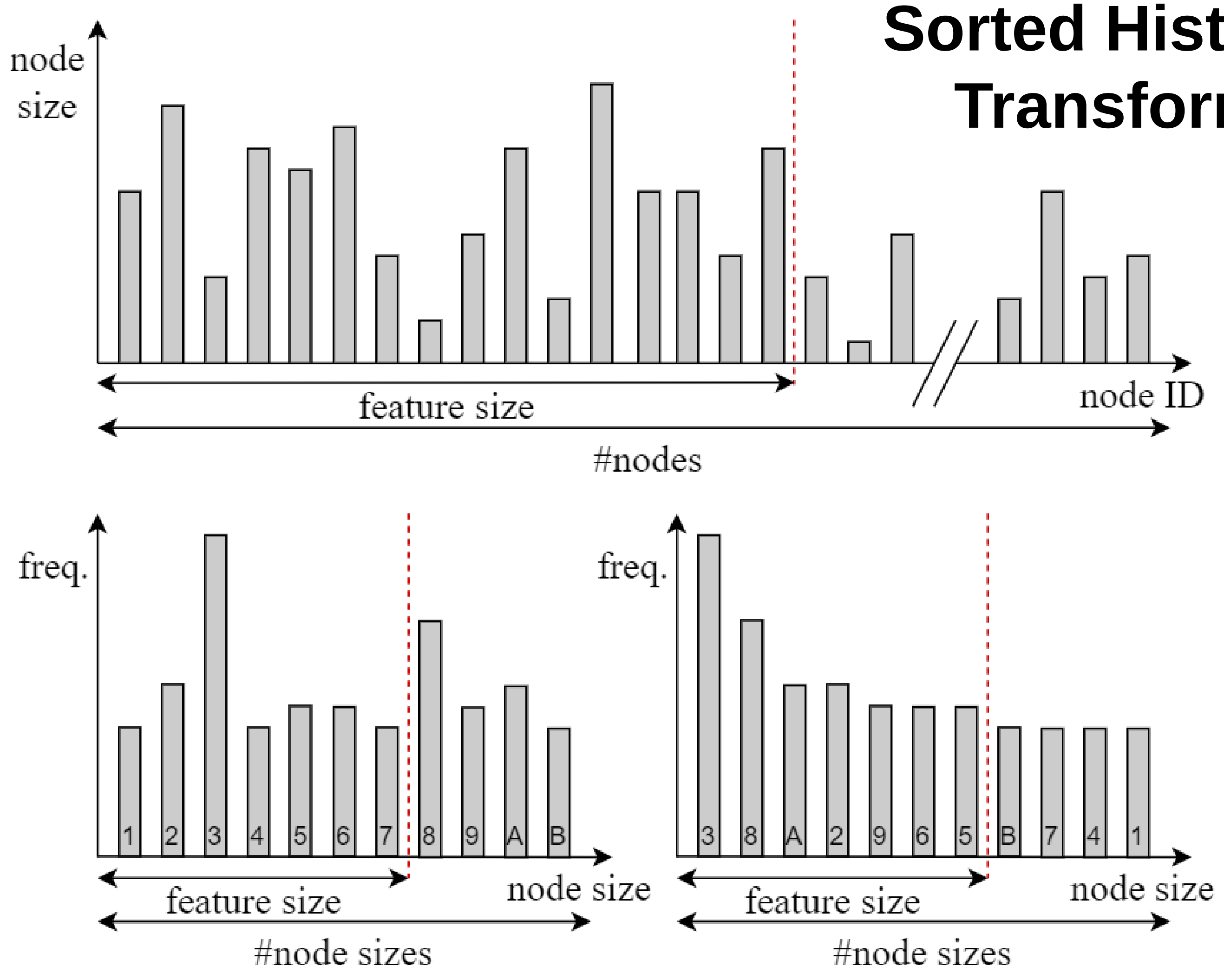
- Immediate extraction
- No operations on CFG required

- Not much structural information
→ Lower robustness

Variable Length

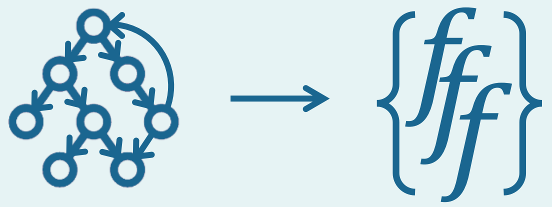
Subgraph

Sorted Histogram Transformation



CFG Features

Fundamental

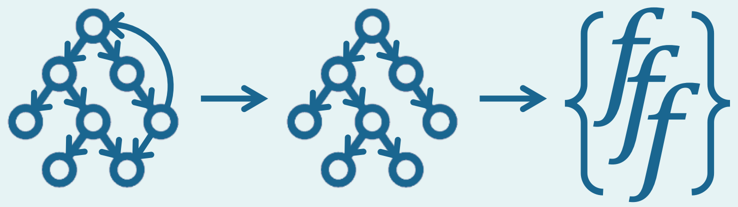


- Immediate extraction
- No operations on CFG required

- Not much structural information

→ Lower robustness

Variable Length

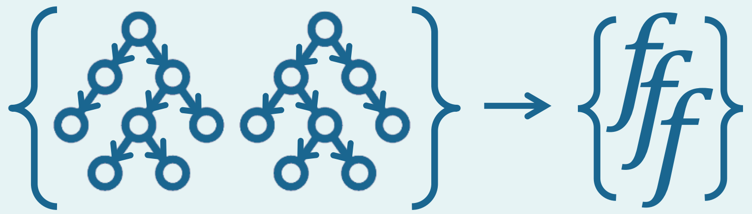


- Increased structural information

→ Higher robustness

- Only takes instructions in the root subgraph of the CFG into account *

Subgraph



- Takes every subgraph into account

→ Less information loss

- Higher extraction time
- Some subgraphs might contain redundant info

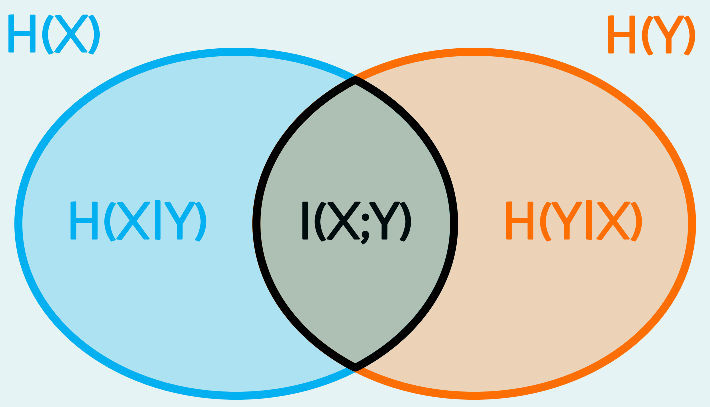
* "Root Subgraph" = Subgraph downwards connected to the entry point

Feature Selection

Filter

- Statistical measure
- Threshold

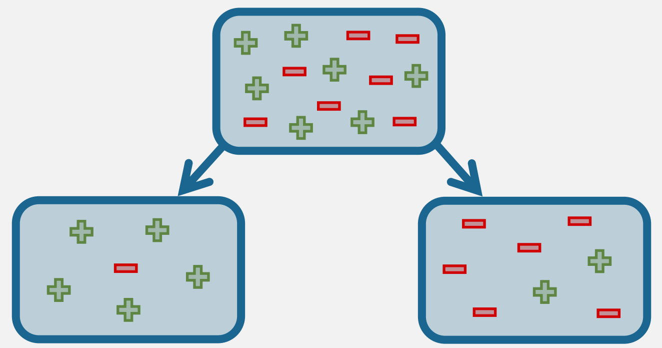
Mutual Information (MI) = I(X;Y)



Embedded

- Training algorithm assigns score and selects

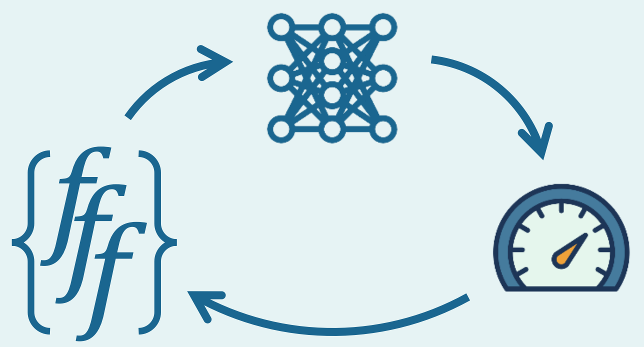
Random Forest → Average decrease in impurity



Wrapper

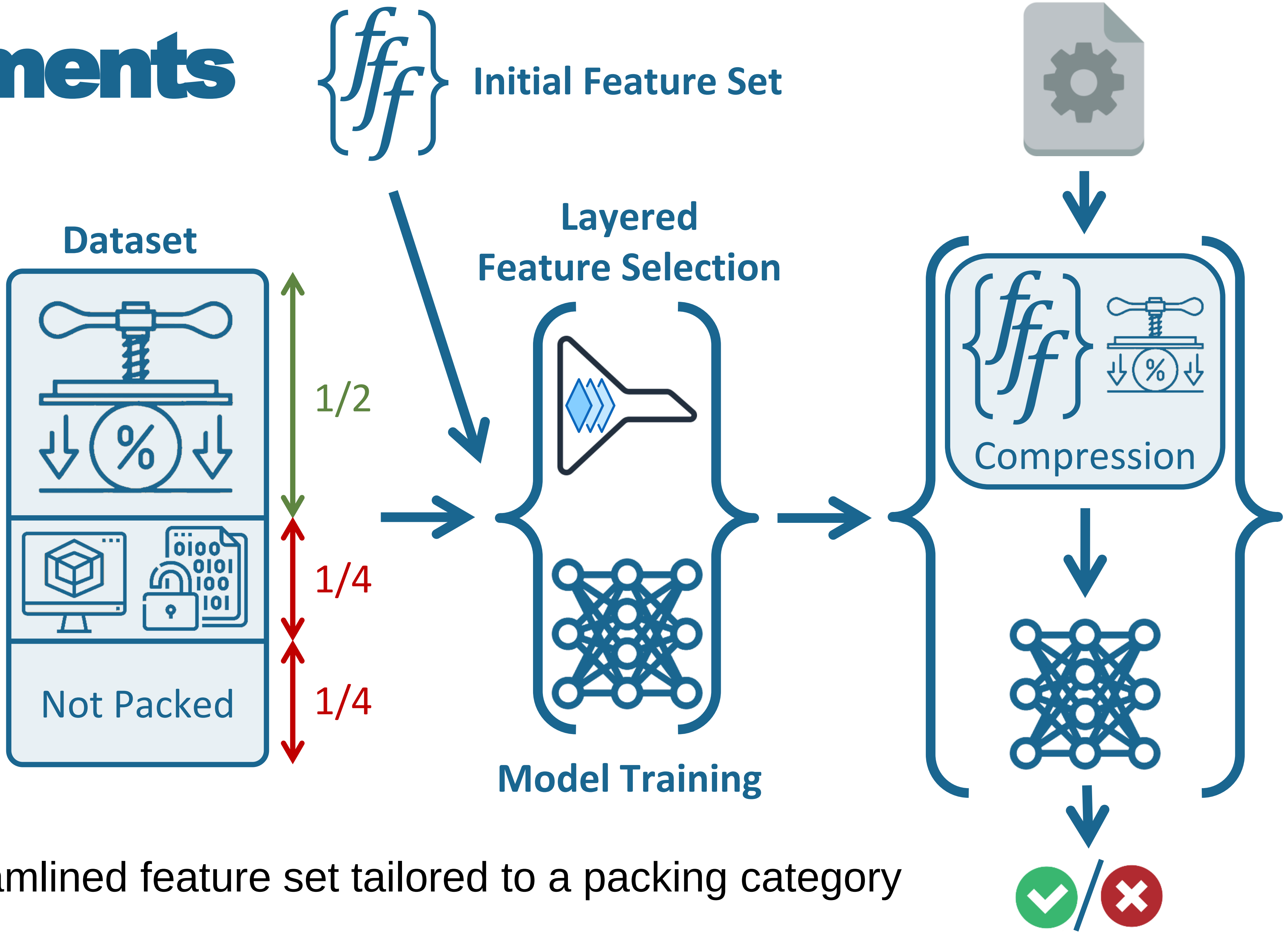
- Model performance forms benchmark

Recursive Feature Elimination with Cross-Validation (RFECV)



4. Feature Engineering

Experiments



➔ Streamlined feature set tailored to a packing category

4. Feature Engineering

Experiments

Dataset Creation

Try It Now !

```
$ for P in ASPack BeRoEXEPacker MEW MPRESS Neolite NSPack Packman PECompact PEtite RLPack TELock UPX WinUpack; \
    do dataset update com-1a -n 66 --source-dir dataset-packed-pe/packed/$P \
        --labels dataset-packed-pe/labels/labels-compressor.json; done
$ for P in EXpressor 'Eronana Packer' Exe32pack FSG; \
    do dataset update com-1b -n 115 --source-dir dataset-packed-pe/packed/"$P" \
        --labels dataset-packed-pe/labels/labels-compressor.json; done
$ for P in Alienyze Yoda-Crypter Yoda-Protector; \
    do dataset update com-cry -n 110 --source-dir dataset-packed-pe/packed/$P; done
$ for P in 'Enigma Virtual Box' Molebox Themida; \
    do dataset update com-vir -n 110 --source-dir dataset-packed-pe/packed/"$P"; done
```

Try It Now !

```
$ dataset update com-1a --source-dir dataset-packed-pe/not-packed -n 426
$ for C in cry vir; do dataset select --split -n 216 "com-$C" com-1a; done
$ dataset update com-1b --source-dir dataset-packed-pe/not-packed -n 232
$ for C in cry vir; do dataset merge com-1b "com-$C"; done
```

Experiments

Feature Selection

Try It Now !

```
$ model train com-1a -A rf -M -k 0.9 --wrapper-select --wrapper-param scoring="matthews_corrcoef" \
    --true-class compressor --features-set features.yml
```

Try It Now !

```
$ model test `model list | awk '$1 ~ /^com-1a_/ {print $1}'` com-1b \
    --true-class compressor --features-set features.yml
```

4. Feature Engineering

EP = Entry Point

Observations

Most (diverse) training data
➔ More structural features required

Compression

Encryption

Virtualization

- Entropy features
- Useful info close to EP
- Lots of CFG features

- Function import features
- Ratio of static size EP section over virtual size EP section

- Section features
- No feature overlap among these three packing categories

Outline

- Contribution
- Future work

Contribution

Toolkit extensions for feature engineering

- ✓ Design & implementation of new structural CFG-features
- ✓ Integration of MI-filter & RFECV-wrapper feature selection algorithms
- ✓ Construction of a layered selection methodology
- ✓ Creation of a feature tool for interacting with feature sets

Future work

- Engineer even more feature classes
- Aggregation of category detectors in one superclassifier
- Robustness analysis of CFG features in adversarial context
- Dataset expansion to avoid overfitting

Packing Box: Improving Detection of Executable Packing

black hat ARSENAL

DECEMBER 9-12, 2024

EXCEL LONDON / UNITED KINGDOM

Awesome list gathering our whole bibliography and many other references to documentation, tools, etc.

Ready-to-use dataset of packed and not-packed ELF files

Ready-to-use dataset of packed and not-packed PE files (enriched version of Choi's dataset)

Entropy-based tool inspired from the study of Lyda et al. in 2007

Heuristic-based tool inspired from the study of Han et al. in 2009

Operationalized fork of https://github.com/cylance/PyPackerDetect

Python fork of the popular tool, PEiD

Attack tool for altering packed samples so that they evade static packing detection

Custom exchange format for datasets (supports conversion to ARFF, CSV, Packing-Box dataset)

Library for getting samples from multiple malware databases