Android Malware Family Classification: What Works -- API Calls, Permissions or API Packages?

Saurabh Kumar, Debadatta Mishra, and Sandeep Kumar Shukla

Indian Institute of Technology Kanpur

Presented at

SIN-2021

Motivation

- Rapid growth of Android malware
  - 3.12 million new samples in 2020 (source AV-TEST)

- More attention to malware detection rather than family identification

- If malware family is known
  - Same removal technique can be reuse
  - Identify damages done

- Automatic malware family classification is also important

Dataset

- Collected AMD dataset
  - 24553 unique labeled malware
  - Distributed in 71 families

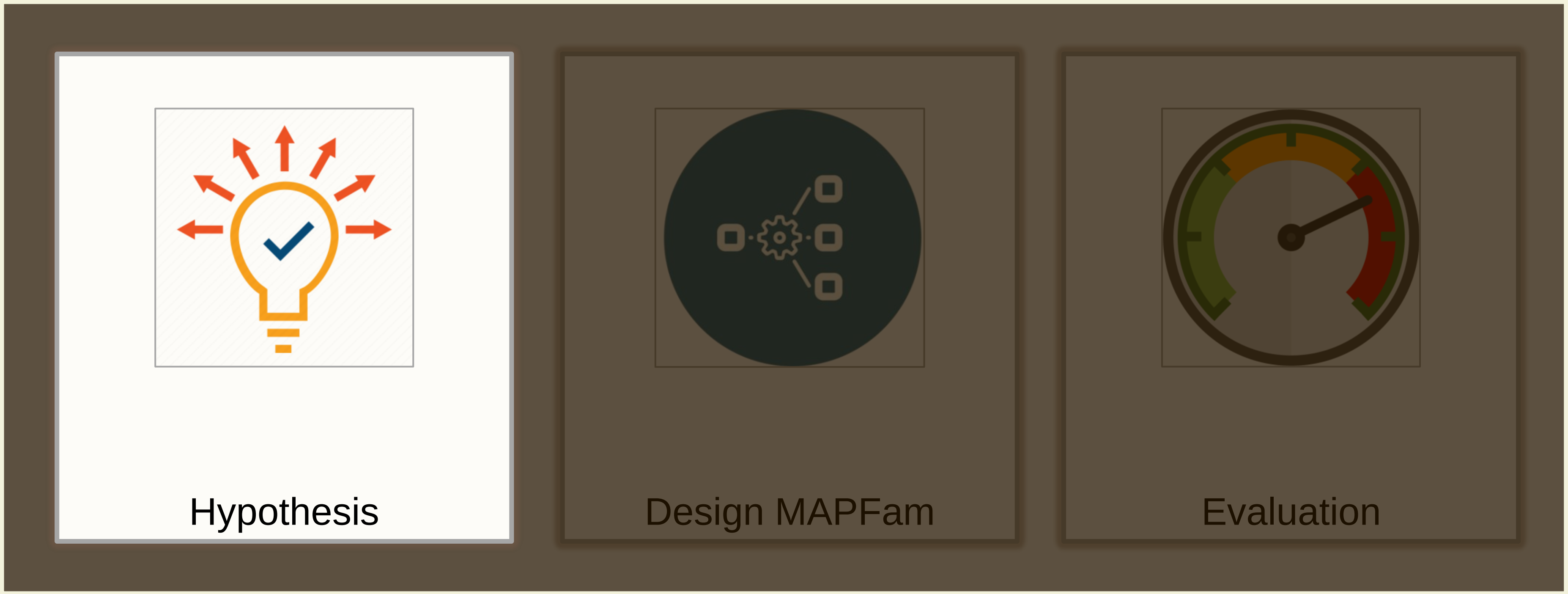
- Select top 60 malware family
  - At least 9 unique samples

- Randomly selected 70% sample for the training and rest for the evaluation

Selected Families

ID	Family	Size
0	airpush	7843
1	dowgin	3384
2	fakeinst	2172
3	mecor	1820
4	youmi	1300
5	fusob	1270
6	kuguo	1199
7	jisut	558
8	droidkungfu	546
9	bankbot	460
10	rumms	402
11	lotoor	329
12	mseg	235
13	boqx	215
14	minimob	203
15	triada	197
16	kyview	175
17	slembunk	174
18	simplelocker	172
19	smskey	165
20	gumen	145
21	gingermaster	128
22	leech	109
23	nandrobox	76
24	bankun	70
25	koler	69
26	mtk	67
27	golddream	53
28	androrat	46
29	erop	46
30	andup	44
31	boxer	44
32	ksapp	36
33	gorpo	32
34	stealer	25
35	updtkiller	24
36	zitmo	24
37	vidro	23
38	aples	21
39	fakedoc	21
40	fakeplayer	21
41	ztorg	20
42	winge	19
43	penetho	18
44	cova	17
45	mobiletx	17
46	fjcon	16
47	kemoge	15
48	spambot	15
49	mmarketpay	14
50	svpeng	13
51	vmvol	13
52	faketimer	12
53	steek	12
54	utchi	12
55	fakeangry	10
56	opfake	10
57	spybubble	10
58	univert	10
59	finspy	9

MAPFam: Overview



Hypothesis

Design MAPFam

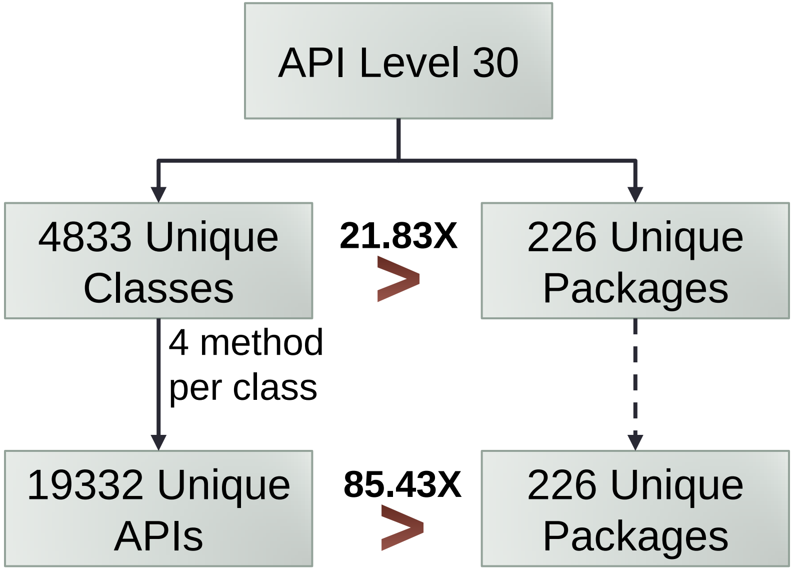
Evaluation

Hypothesis

Use of system API package improves the performance of family classifier with less number of features as compared to API calls

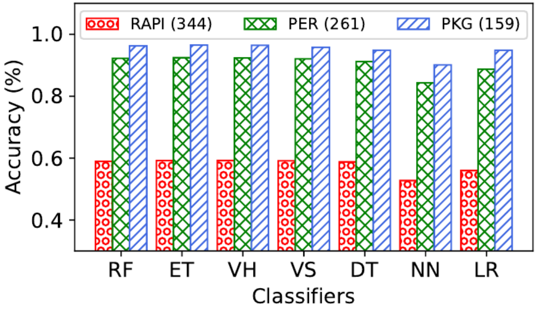
Observation

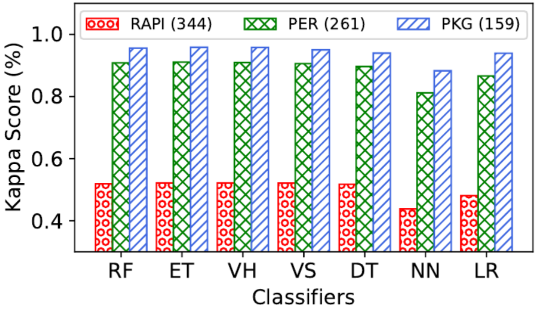
- Performance of API based classifiers
  - Negatively impacted due to obfuscation
  - Increases size of feature set
- API package can be used alternate to API calls
- Benefit
  - Free from obfuscation attack
  - Reduces size of feature set
- Example:
  - Android API level 30



Testing The Hypothesis

- ❑ Extracted
  - ➢ Restricted APIs (RAPI)
  - ➢ Requested Permissions (PER)
  - ➢ API Packages (PKG)
- ❑ Trained 7 classifiers and observes
  - ➢ Accuracy
  - ➢ Reliability (Kappa Score)

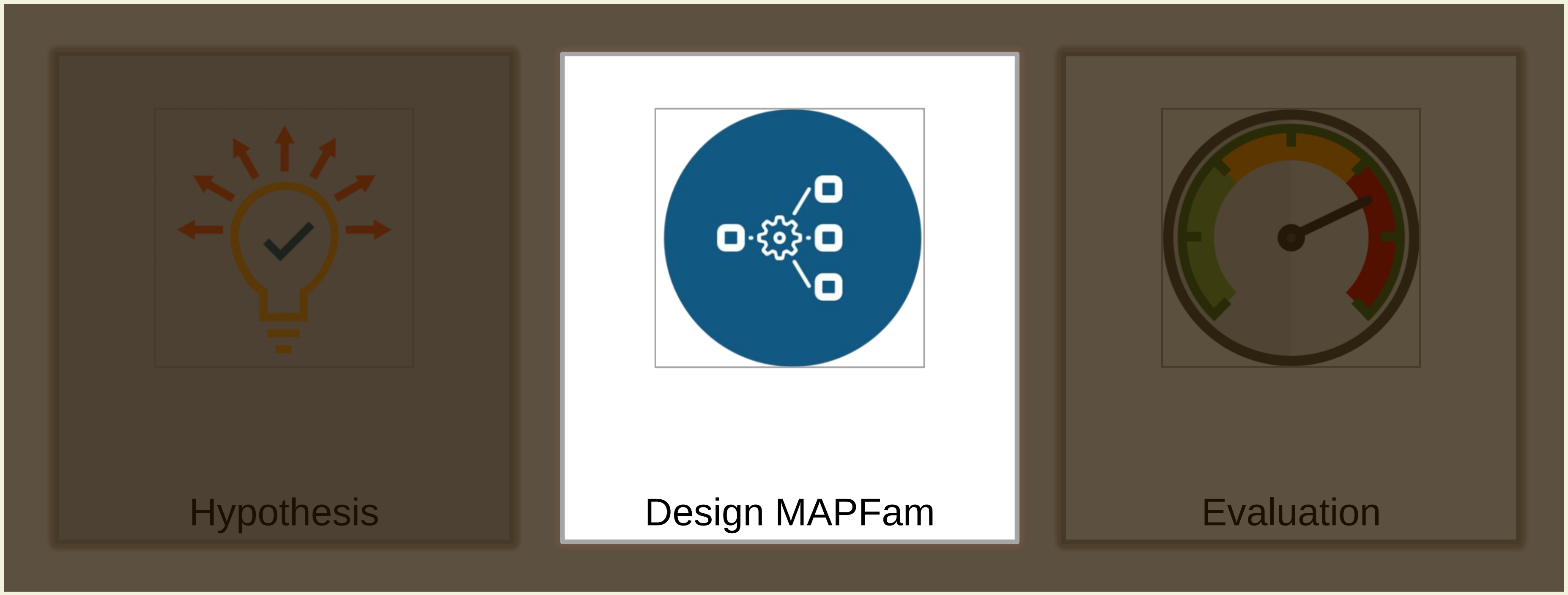




API packages are 1.63X and 1.04X accurate than APIs and permissions

1.84X and 1.05X more reliable than APIs and permissions

MAPFam: Overview



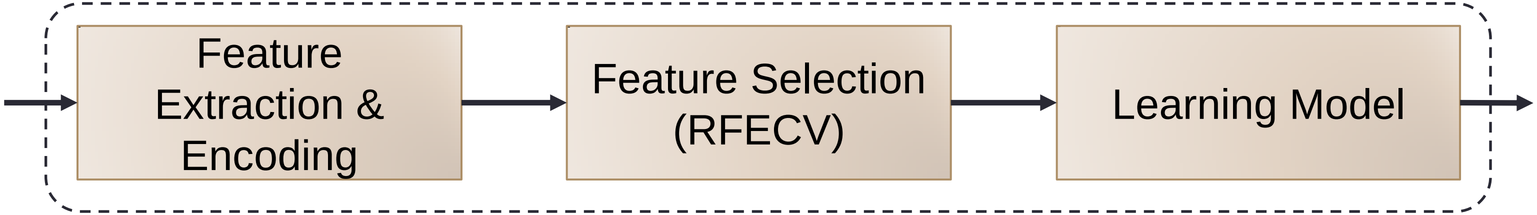
Hypothesis

Design MAPFam

Evaluation

MAPFam Design

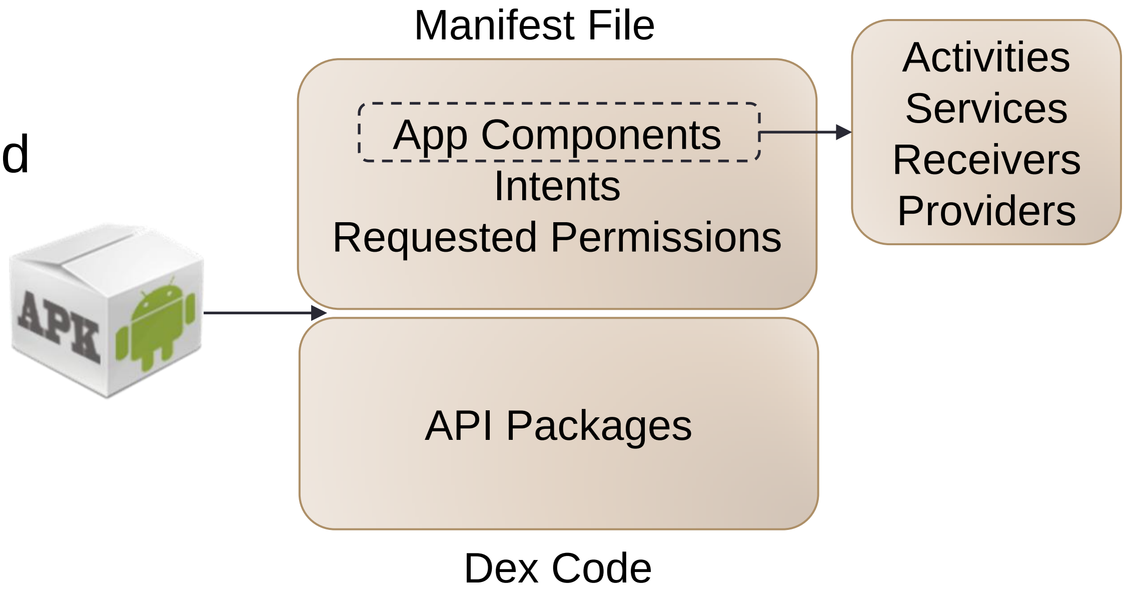
- Three major components



Feature Extraction

❑ Extract features from two sources
  - ➢ Manifest file
  - ➢ Dex Code

❑ Extracted using Androguard
  - ➢ Represented as string



Manifest File

App Components

Intents

Requested Permissions

Activities

Services

Receivers

Providers

API Packages

Dex Code

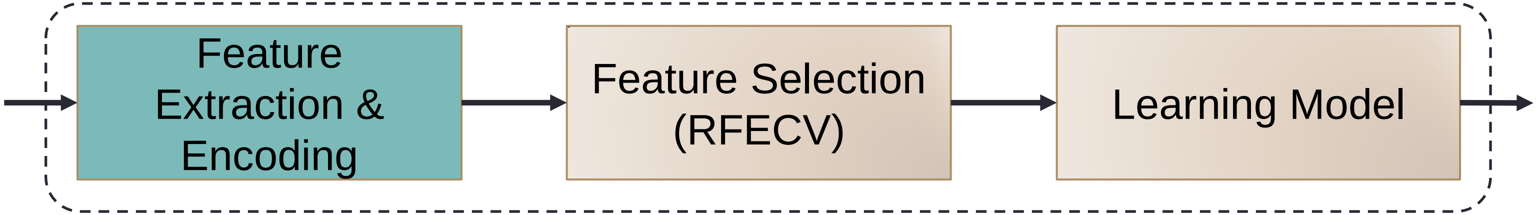
Feature Encoding

- ❑ Encode based on their count and presence (binary)
- ❑ Count: frequency of usage
  - ➢ User defined components like activities, services, custom permissions, etc…
  - ➢ #API packages used
- ❑ Binary: to observe presence
  - ➢ System defined components like permissions, and API Packages

Category	**#Features**
	Encoding
Activities	1
Services	1
Receivers	1
Providers	1
Intents	1
Custom Permissions	1
Package Counts	1
Requested Permissions	261
API Packages	159
Total	428

MAPFam Design

- Three major components



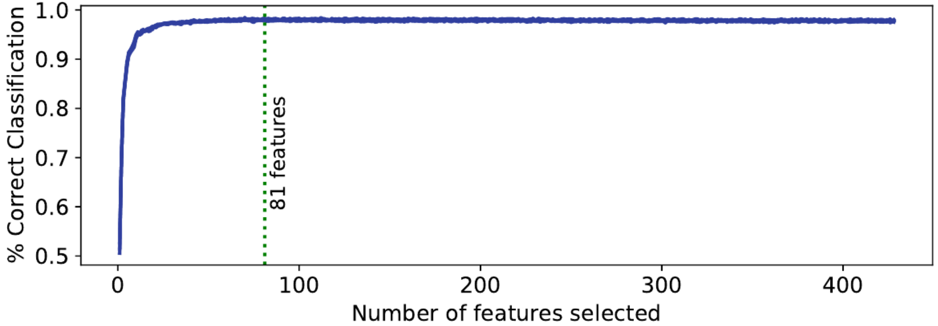
Feature Extraction & Encoding

Feature Selection (RFECV)

Learning Model

Feature Selection

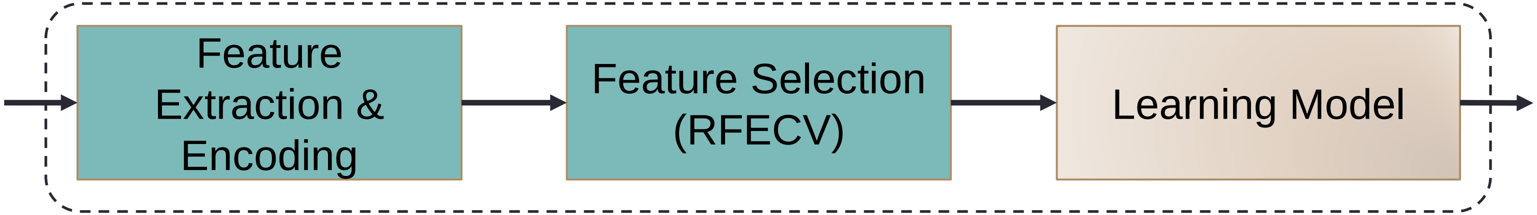
- Use RFECV
  - Classifier: RandomForest
  - Ranking Function: Accuracy
  - Eliminate feature in each step: 1
- Provides optimal #features with highest accuracy



Category	#Features	
	Encoding	Selected
Activities	1	1
Services	1	1
Receivers	1	0
Providers	1	1
Intents	1	1
Custom Permissions	1	1
Package Counts	1	1
Requested Permissions	261	33
API Packages	159	41
Total	428	81

MAPFam Design

- Three major components



Learning Model

- Use ExtraTree to learn final model
  - Ensemble method
  - Information gain
  - Does not require feature scaling
- Train model on 70% of samples AMD dataset
- Remaining 30% for evaluation

MAPFam: Overview

Hypothesis

Design MAPFam

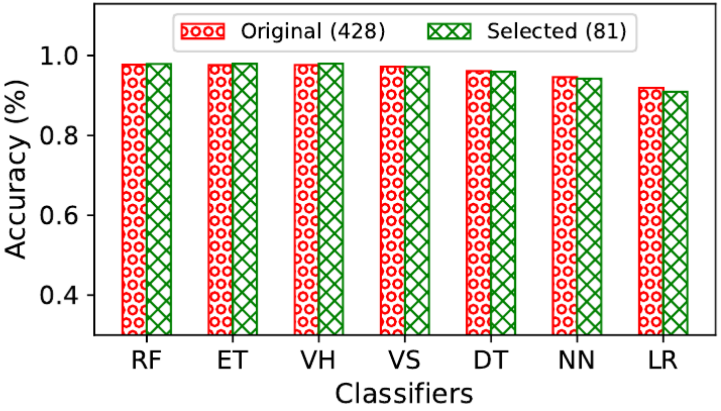
Evaluation

Evaluation

- Evaluation metrics
  - Accuracy
  - Kappa Score
  - Recall
  - Precision

Performance

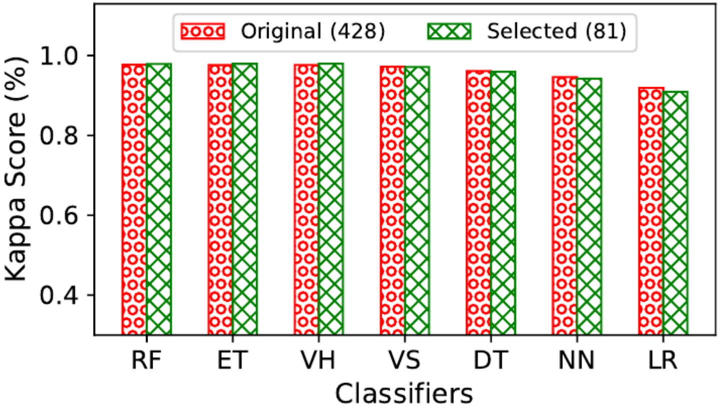
- ❑ Trained 7 different classifiers
  - ➢ Before and after feature selection
- ❑ Observes
  - ➢ Accuracy



97.92% accurate for malware family identification

Performance

- Trained 7 different classifiers
  - Before and after feature selection
- Observes
  - Accuracy
  - Reliability

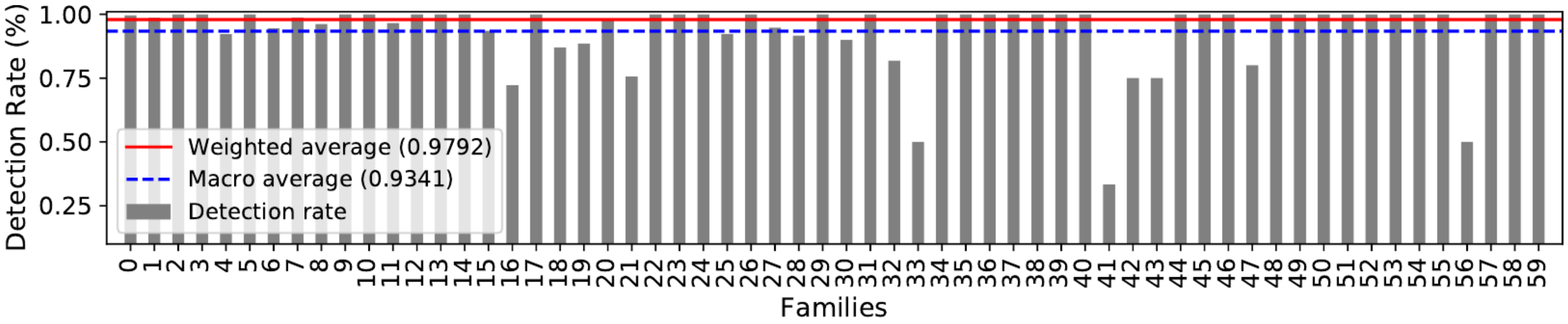


97.92% accurate for malware family identification

MAPFam is 97.55% reliable

Individual Family: Detection Rate

- Trained ExtraTree classifier after feature selection

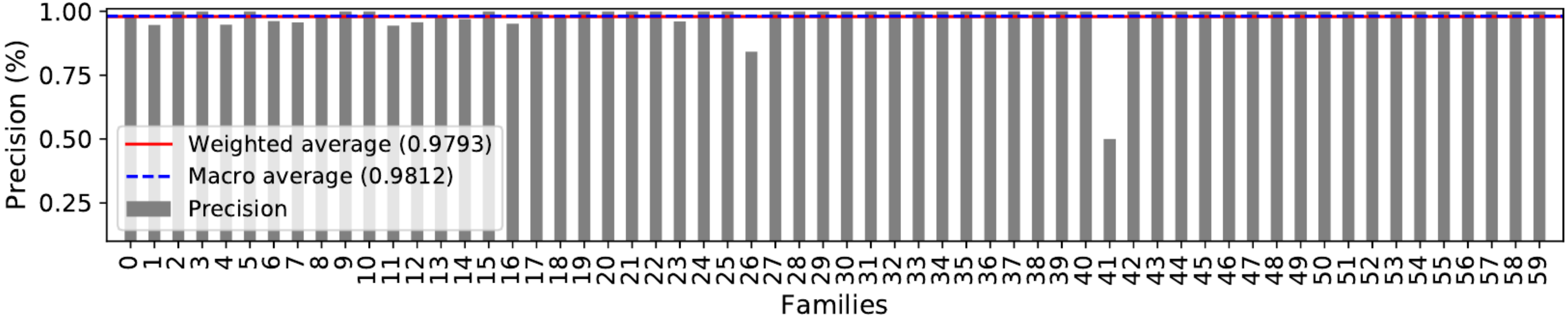


On average, it identify malware family with 97.92% of detection rate

Perfectly identify 36 malware family with 100% detection rate

Individual Family: Precision

- Trained ExtraTree classifier after feature selection



MAPFam can precisely identify malware family with average precision rate of 97.93%

Limitations

- Cannot identify malware family
  - Packed malware
  - Download malicious code from external source at runtime

Conclusion

API Packages are ~1.63X more accurate than API call based model

Precisely classify malware family with average precision and accuracy of more than 97%

MAPFam model is 97.55% reliable

Perfectly identify 36 malware families out of 60

Thank You