DeepDetect: A Practical On-device Android Malware Detector

Saurabh Kumar[1], Debadatta Mishra[1], Biswabandan Panda[2], and Sandeep K. Shukla[1]

[1]Indian Institute of Technology Kanpur

[2]Indian Institute of Technology Bombay

Motivation

- Rapid growth of Android malware
  - 4.18 million new samples in 2019 (source G DATA)

- Malware may get unleashed into the device
  - Bypassing the defense system of Play store
  - Third party market and Sideloading

- Required on device malware detection

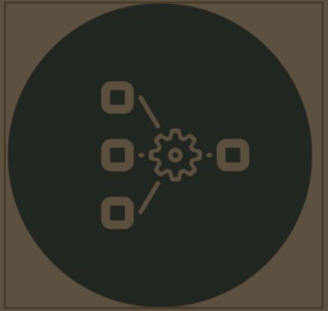
Our Goal

Designing an on-device malware detector that is faster, consume less device energy, provides high malware detection rate and low false alarms

Challenge: Limited energy of mobile device

DeepDetect: Overview

Feature Extraction

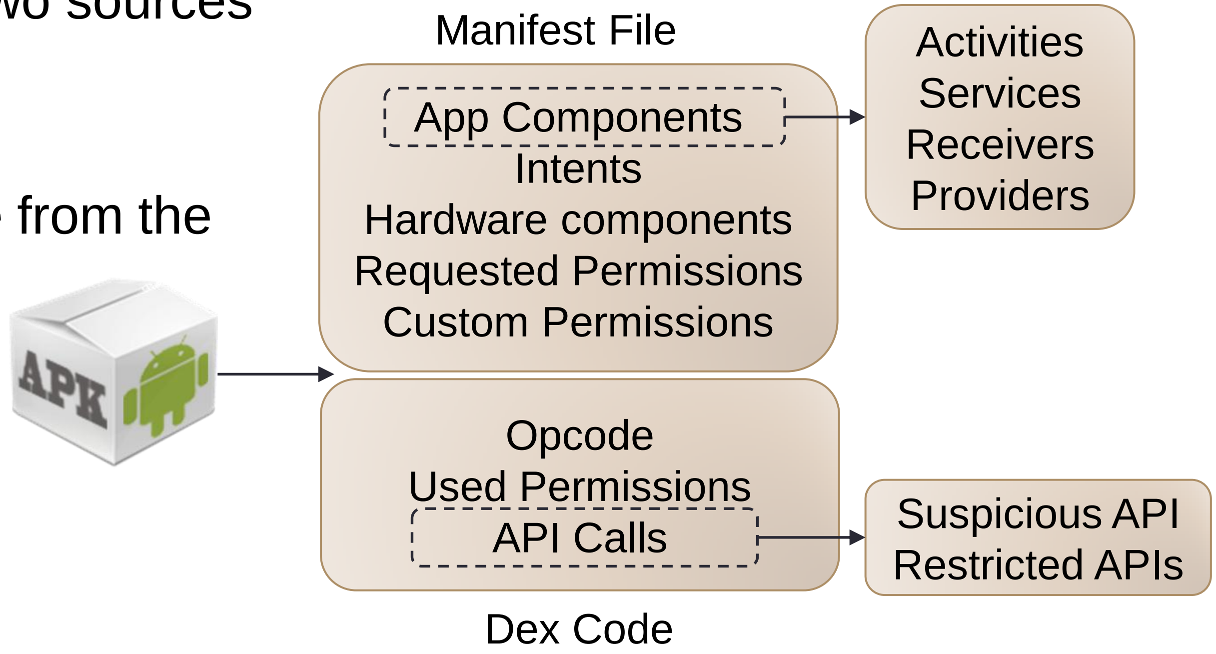


Feature Engineering

Building the System

Feature Extraction

- Extract features from two sources
  - Manifest file
  - Dex Code
- Use Opcode sequence from the Dex Code
  - DexLib2
  - Operate in-memory



DeepDetect: Overview

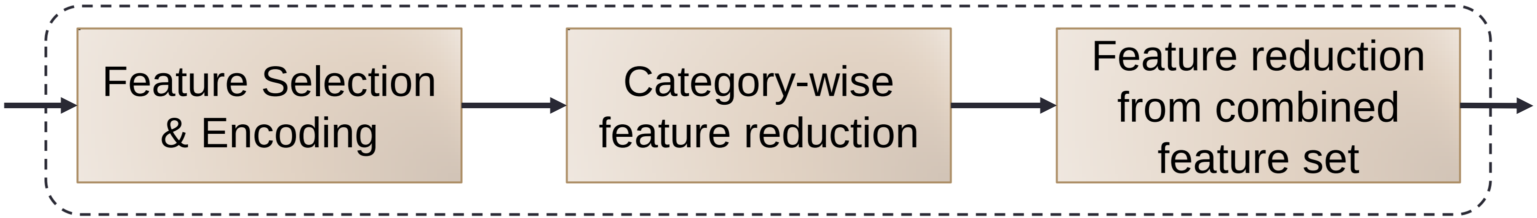
Feature Extraction

Feature Engineering

Building the System

Feature Engineering

- Three major components



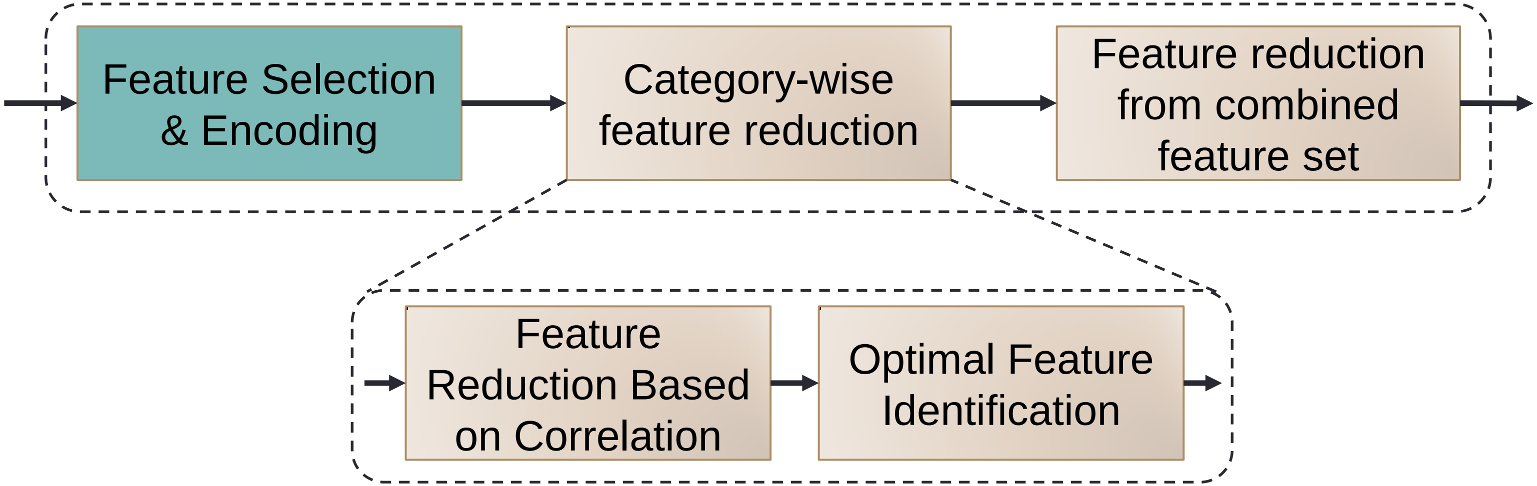
Feature Selection and Encoding

- Encode based on their count and presence (binary)
- Count: frequency of usage
  - User defined components like activities, services, custom permissions, etc…
  - N-Gram Opcode sequences
- Binary: to observe presence
  - System defined components like permissions, hardware features, etc..

Category	#Features	
	Original	Encoding
Activities	5,24,989	1
Services	57,202	1
Receivers	49,751	1
Providers	6,659	1
Intents	50,257	1
Custom Permissions	0	1
Requested Permissions	23,175	668
Hardware Component	245	245
2-Gram Opcode	317	317
Total	7,12,595	1,236

Feature Engineering

- Three major components



Correlation Based Reduction

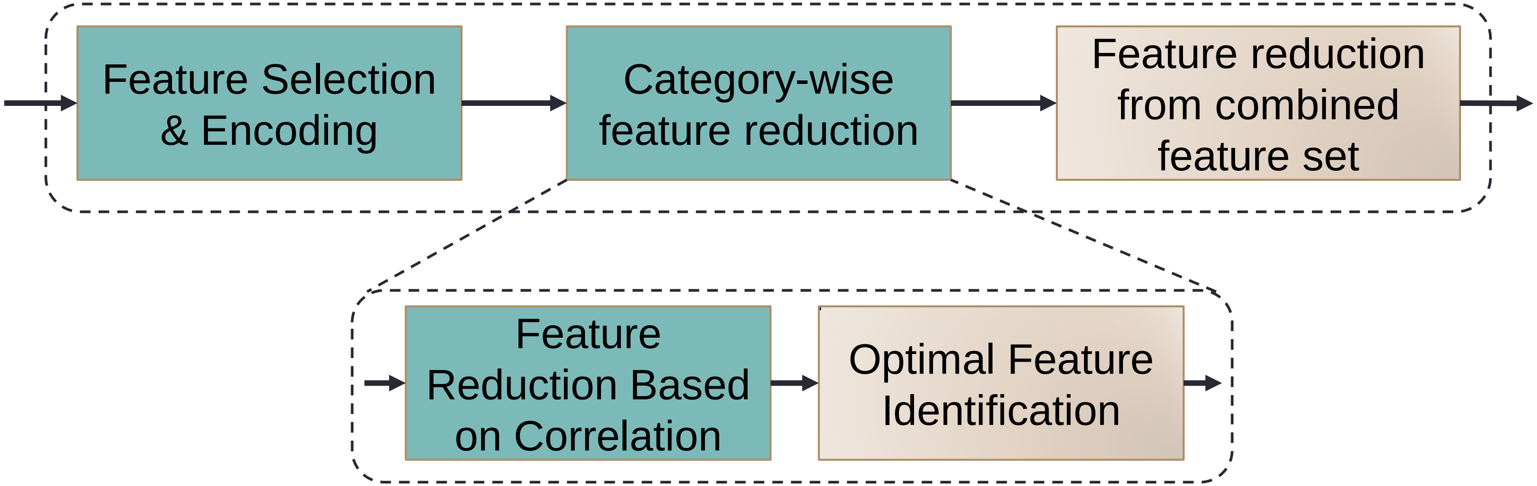
- Use Pearson correlation (-1 to +1)
  - 0: weak
  - ±1: strong positive and negative
- Different threshold (COR_T) 0.5 to 1.0 in both directions

COR_T	Accuracy (%) / #Features		
	ReqPerm	HWC	2-OPC
0.5	93.69 / 533	60.58 / 194	86.32 / 39
0.6	94.00 / 562	60.79 / 207	90.05 / 55
0.7	93.99 / 575	60.82 / 212	94.82 / 72
0.8	94.74 / 602	60.80 / 224	95.50 / 104
0.9	94.86 / 626	60.79 / 226	95.99 / 172
1.0	94.89 / 668	60.85 / 245	96.28 / 317

- 0.8 for requested permissions and hardware components
- 0.9 for 2-Gram Opcode sequence

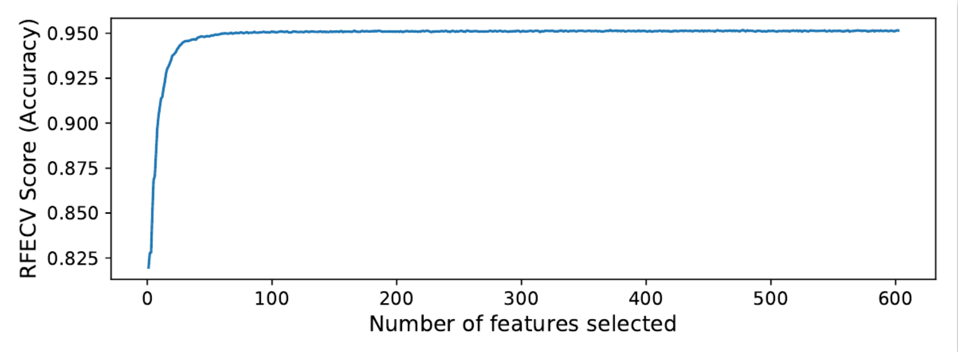
Feature Engineering

- Three major components



Optimal Feature Identification

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



Optimal #features Vs. accuracy for requested permissions

Feature	Accuracy	#Features
ReqPerm	94.77	371
HWC	60.79	185
2-Opc	90.01	169

Observation: With a significant less number of features result in an accuracy close to maximum achievable accuracy

Optimal Feature Identification cont..

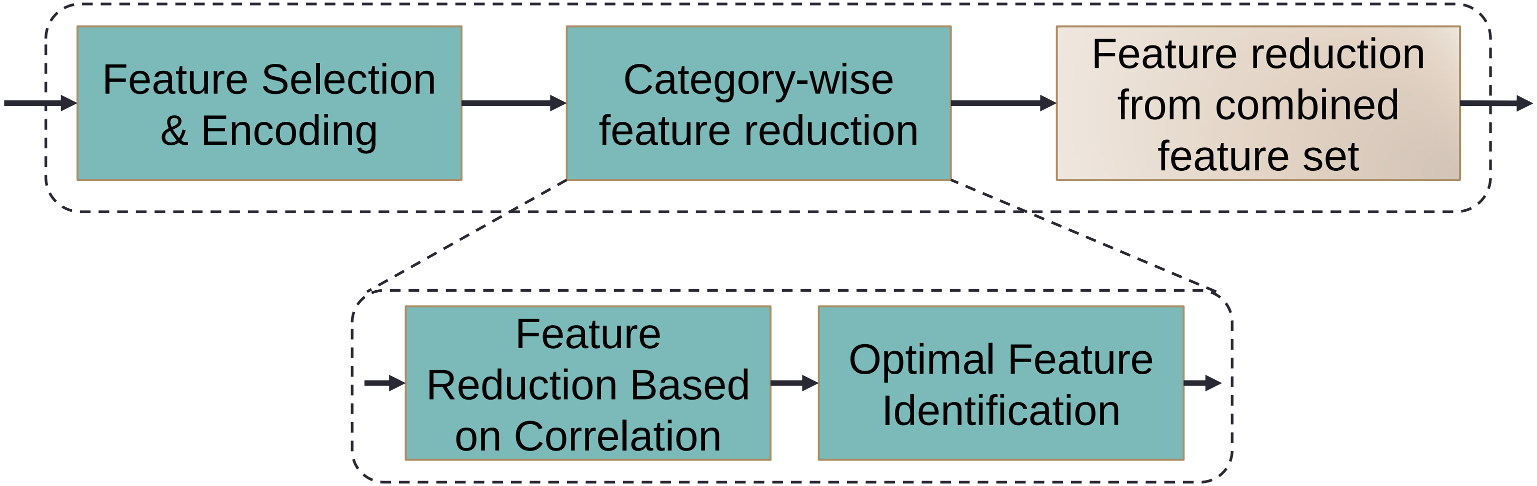
- Define threshold RFE_T, penalty in choosing less #features in terms of accuracy
- Evaluated for different RFE_T values from 0.0 to 0.5

RFE_T	Accuracy (%) / #Features		
	ReqPerm	HWC	2-OPC
0.0	94.77 / 371	60.79 / 185	96.01 / 169
0.1	94.76 / 86	60.75 / 17	95.92 / 69
0.2	94.68 / 60	60.72 / 13	95.90 / 62
0.3	94.57 / 52	60.65 / 13	95.76 / 48
0.4	94.47 / 41	60.62 / 13	95.76 / 37
0.5	94.34 / 40	60.60 / 12	95.64 / 37

A drastic reduction in feature set size for 0.5 as RFE_T value

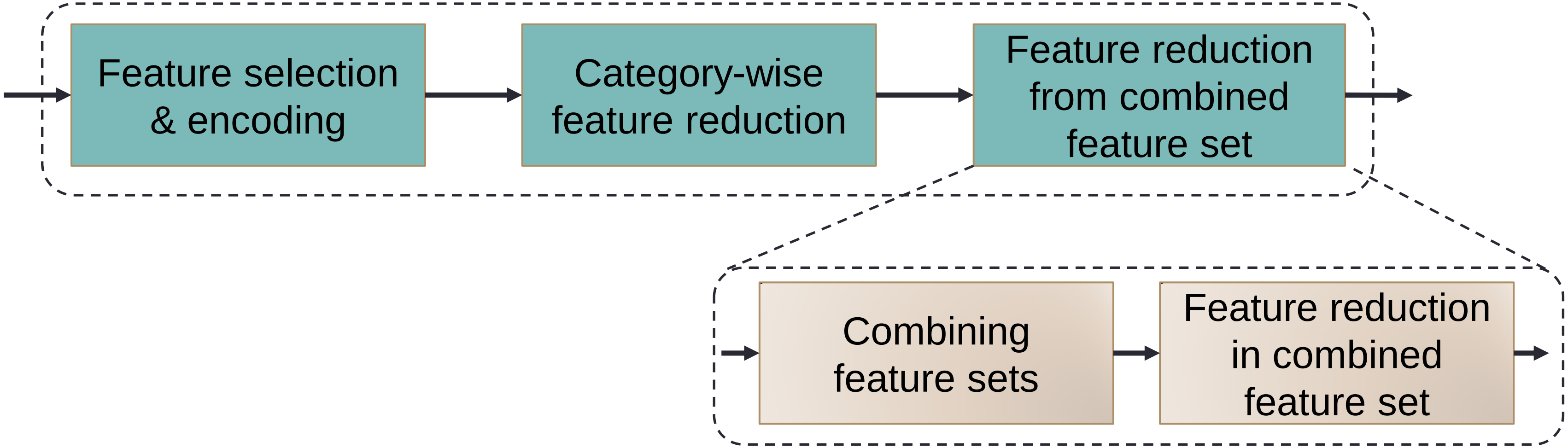
Feature Engineering

- Three major components



Feature Engineering

- Three major components



Combining Feature Sets

- Combined all the feature set in different combination
  - N: numeric features obtained using encoding
  - R: reduced requested permissions
  - H: reduced hardware components
  - O: reduced 2-Gram Opcode
- Two different combination with highest accuracy.

Combination	#Features	Acc	Pre	Rec
N+H	18	86.45	86.55	85.46
N+O	36	96.90	96.93	96.90
H+O	42	96.07	96.19	96.07
N+R	46	96.45	96.46	96.24
N+H+O	48	96.87	96.89	96.87
R+H	52	95.16	95.08	94.96
N+R+H	58	96.57	96.59	96.37
R+O	70	98.15	98.15	98.15
N+R+O	76	98.14	98.15	98.14
R+H+O	82	98.12	98.12	98.12
N+R+H+O	88	98.12	98.12	98.12

Combining Feature Sets

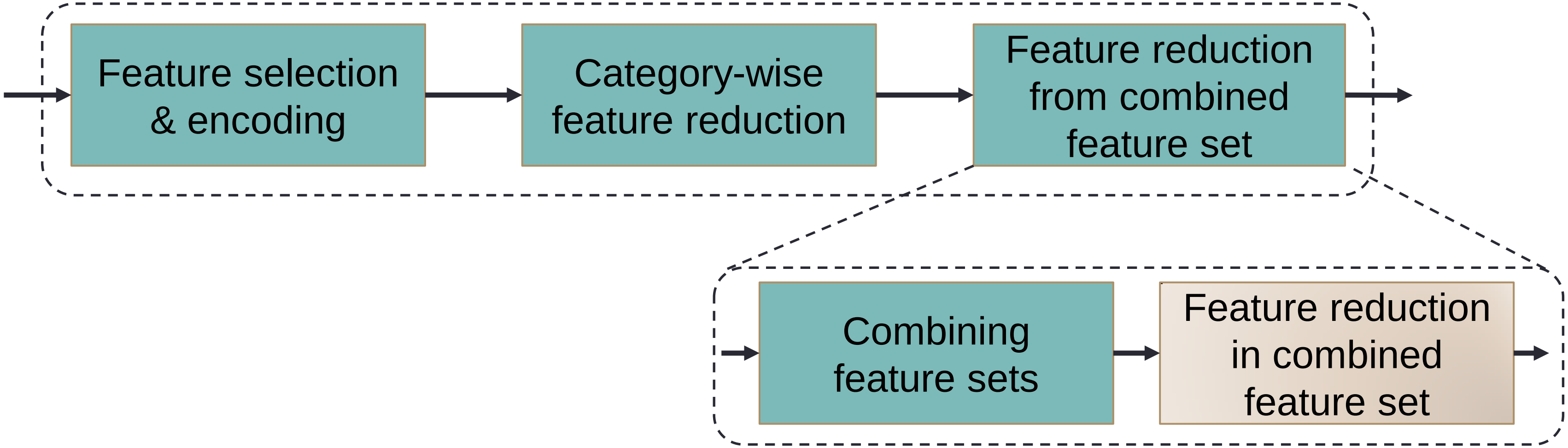
- ❑ Combined all the feature set in different combination
  - N: numeric features obtained using encoding
  - R: reduced requested permissions
  - H: reduced hardware components
  - O: reduced 2-Gram Opcode
- ❑ Two different combination with highest accuracy.

Combination	#Features	Acc	Pre	Rec
N+H	18	86.45	86.55	85.46
N+O	36	96.90	96.93	96.90
H+O	42	96.07	96.19	96.07
N+R	46	96.45	96.46	96.24
N+H+O	48	96.87	96.89	96.87
R+H	52	95.16	95.08	94.96
N+R+H	58	96.57	96.59	96.37
R+O	70	98.15	98.15	98.15
N+R+O	76	98.14	98.15	98.14
R+H+O	82	98.12	98.12	98.12
[illegible]	[illegible]	[illegible]	[illegible]	98.12

Select N+R+O as contribution of numeric feature is significant when combined with requested permissions

Feature Engineering

- Three major components



Reduction in Selected Feature Set

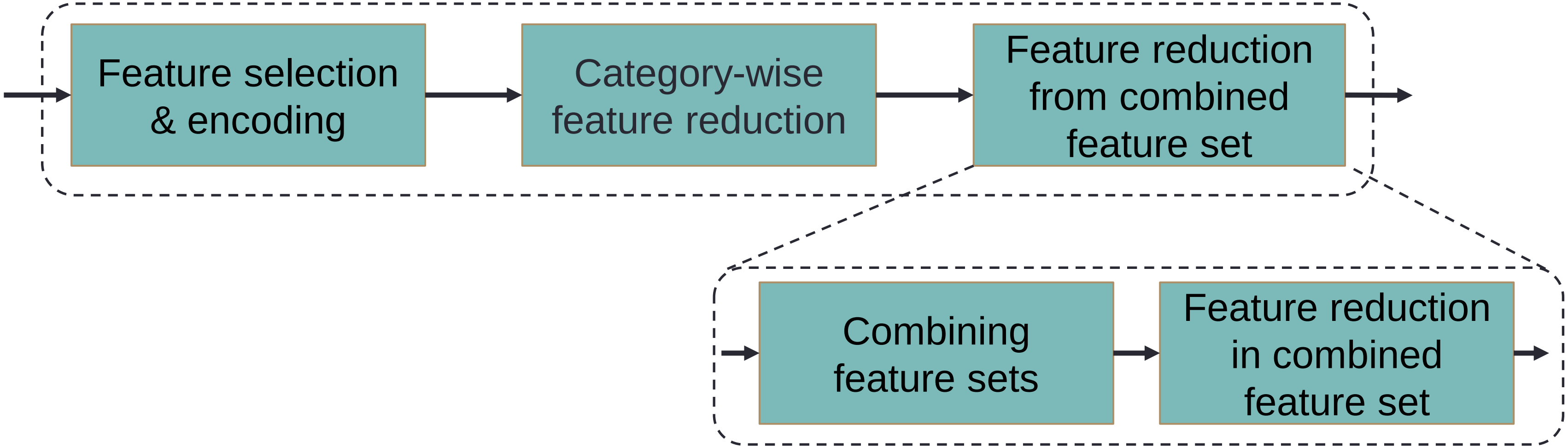
- Eliminate features that require extra support
  - Intents (I)
  - Custom permissions (C)
- Observe effect either removing one or both

Feature Set	#Features	Acc	Pre	Rec
N+R+O	76	98.14	98.15	98.14
N+R+O-I	75	98.18	98.18	98.18
N+R+O-C	75	98.08	98.08	98.08
N+R+O-I-C	74	98.13	98.13	98.13

Removes Intents as it does not impact model performance, hence selects N+R+O-I

Feature Engineering

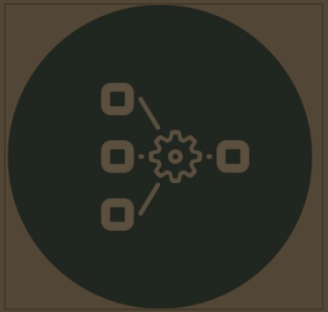
- Three major components



Use final feature set with 75 features to design DeepDetect

DeepDetect: Overview

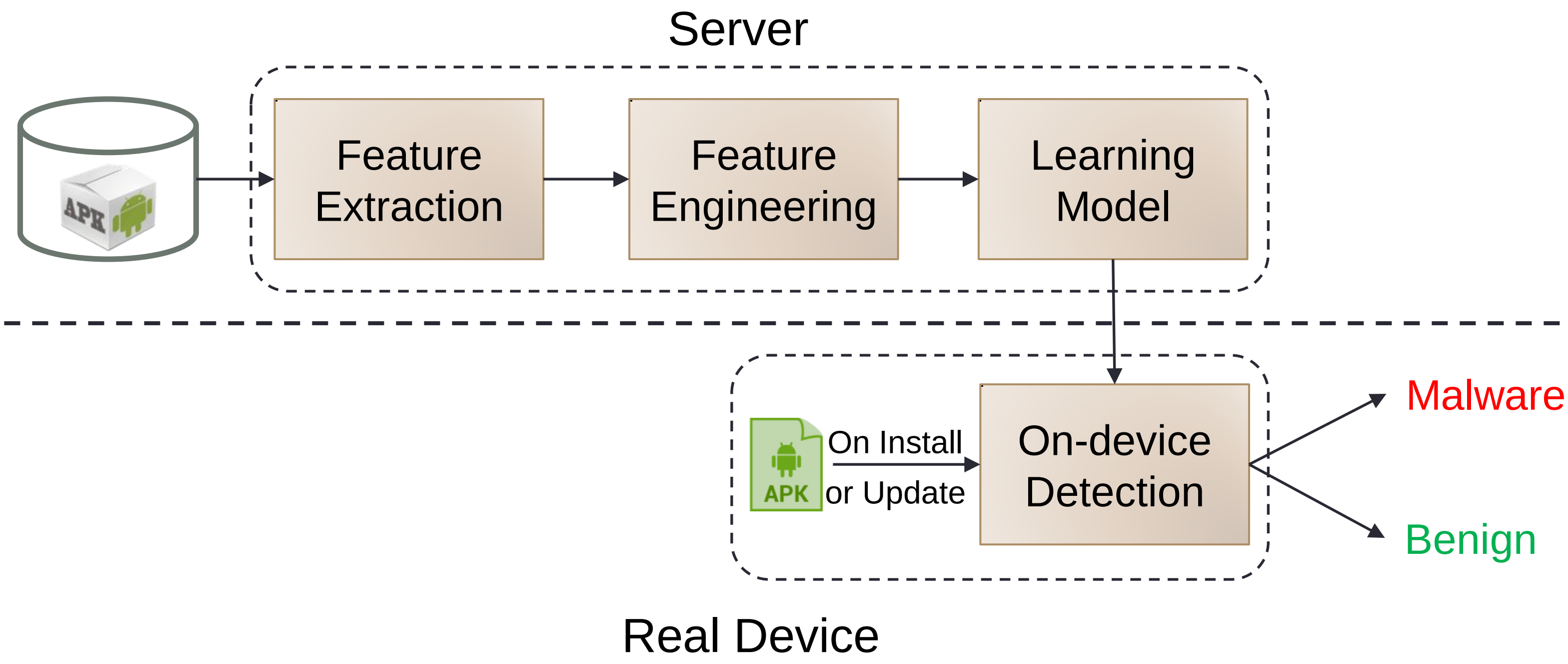
Feature Extraction



Feature Engineering

Building the System

Building The System

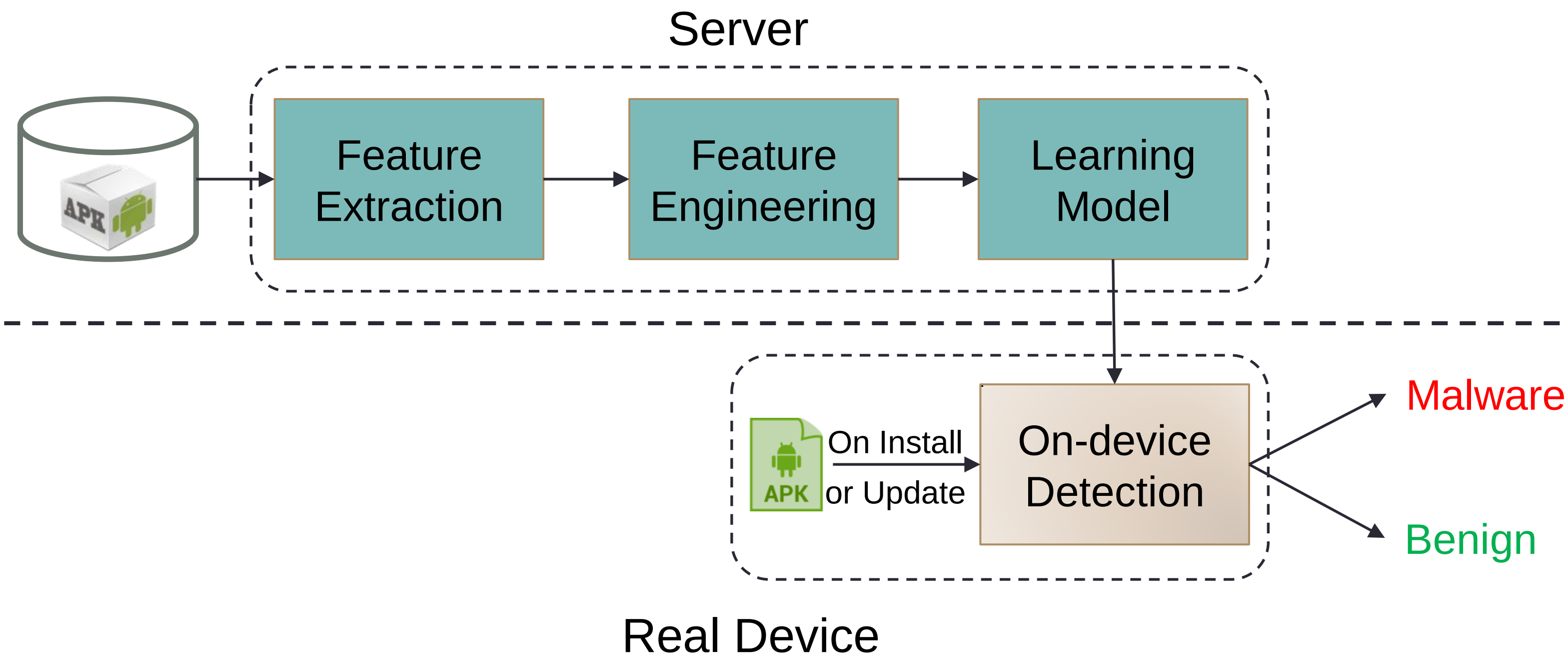


Learning Model

- Use RandomForest to learn final model **Why?**
  - Ensemble method
  - Information gain
  - Does not require feature scaling
- Use TensorFlow library to learn final model
- Converted to TensorFlow Lite model for on-device detection

	Classifier	Recall	FPR	Training time
No Scaling	RF	97.50	1.51	0.5134
	SVM	84.29	32.11	533.6512
	KNN	90.98	8.68	32.0794
	Neural Net	93.19	8.41	28.4913
Scaling	RF	97.49	1.54	0.5799
	SVM	96.11	2.52	108.3715
	KNN	96.47	3.15	34.3693
	Neural Net	96.09	4.30	37.0156

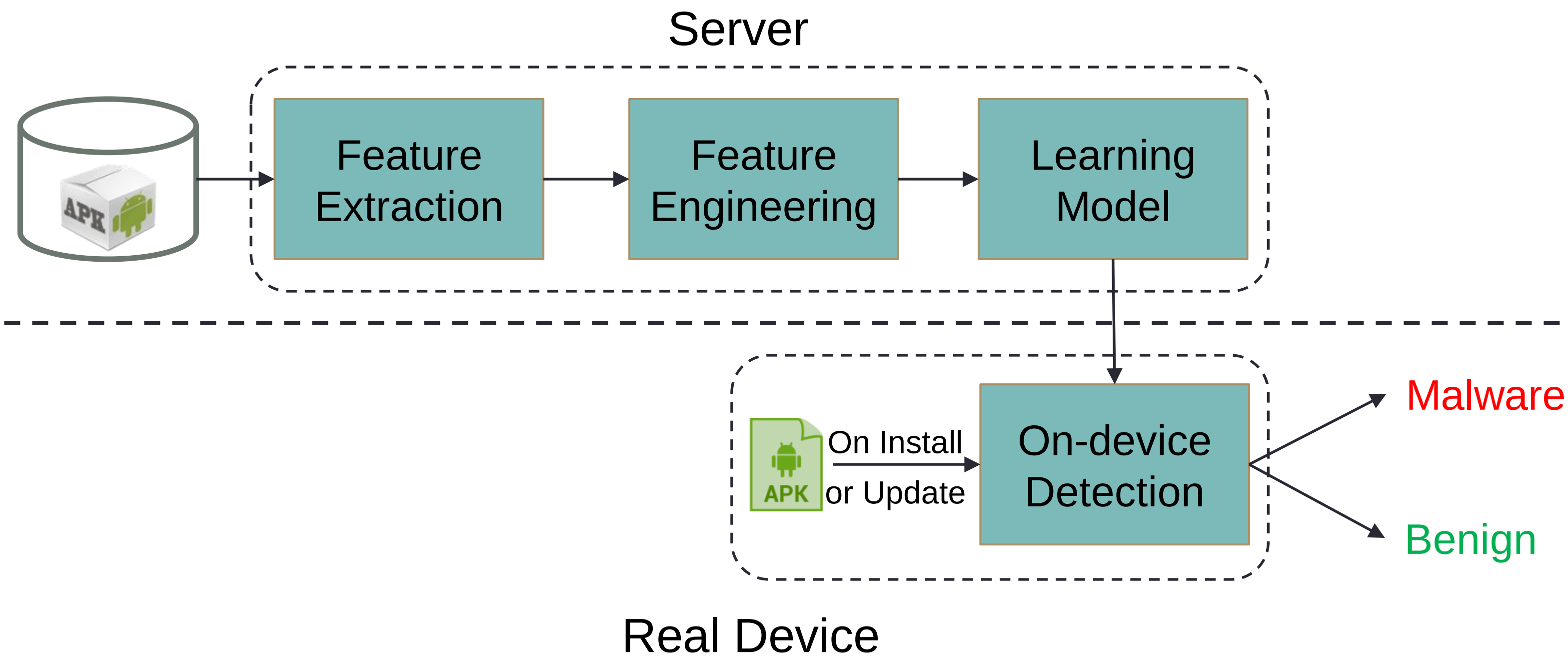
Building The System



On-device Detection

- Check when an App gets installed/updated
- Generate feature vector
- Pass to detection model
  - Benign:
  - Malware:
    - Notify to user
    - Option to uninstall App

Building The System



Evaluation

- Evaluation metrics
  - Precision
  - Recall
  - F1-Score
  - False Positive Rate
- Runtime performance
  - Execution time
  - Device energy consumption

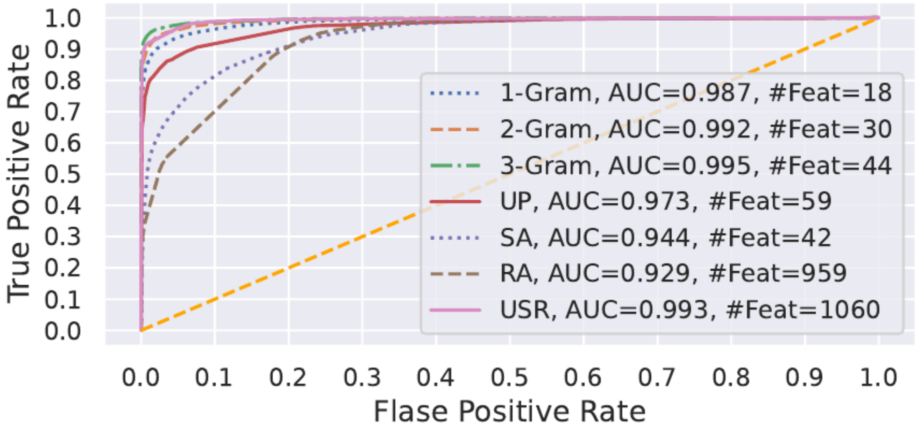
Dataset

- Multiple Datasets
- Training (Known): 80% samples of AMD, VirusShare and Play Store
- Evaluation (Unseen): 20% of AMD, VirusShare and Play Store
- New: AndroZoo (2019) and Pegasus samples
- Obfuscated by obfuscating Androzoo-2019

Dataset/source	Duration	Malware	Benign
AMD	Till 2016	24553	--
VirusShare	Till 2018	20976	--
Pay store	Till 2018	--	56346
AndroZoo-2019	2019	5380	5380
Pegasus (CloudSek)	Pre 2019	5	--
Obfuscated	2019	4993	--

Performance Comparison of Features

- Extracted 7 different features from Dex code
- RandomForest classifier is trained on training set
- Evaluated against evaluation set
- Observes
  - ROC curve and AUC value

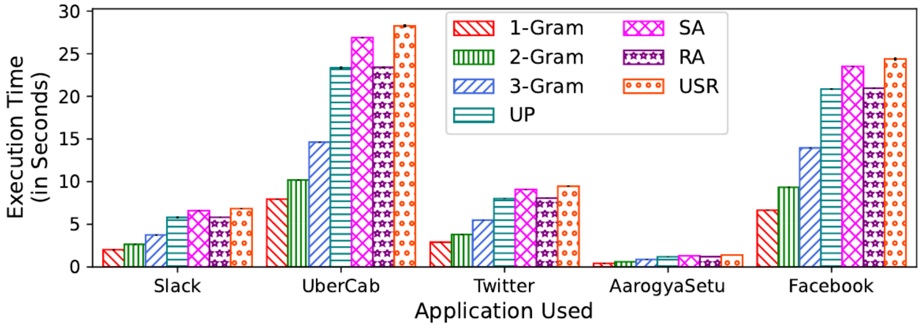


Three feature set (2-Gram, 3-Gram, USR) with more 99% AUC

Feature set size of USR is relatively large compared to 2-Gram and 3-Gram

Runtime Efficiency of Features

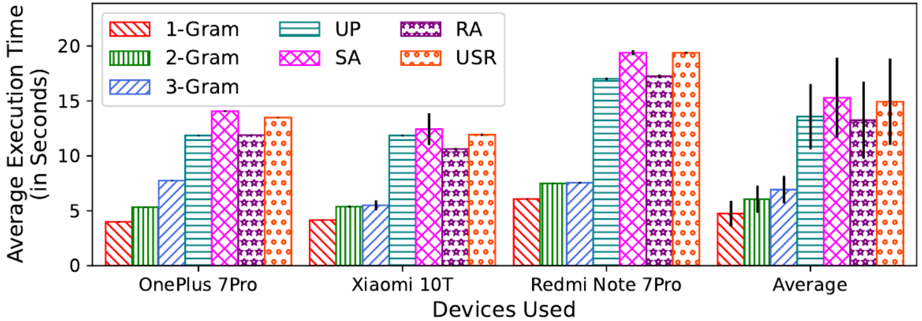
- ❑ Use five different Apps
- ❑ Extracts features on three different smartphones
- ❑ Measures execution time
  - ➢ Per App



Per App execution time on OnePlus 7Pro

Runtime Efficiency of Features

- ❑Use five different Aps
- ❑Extracts features on three different smartphones
- ❑Measures execution time
  - ➢ Per App
  - ➢ Average execution time



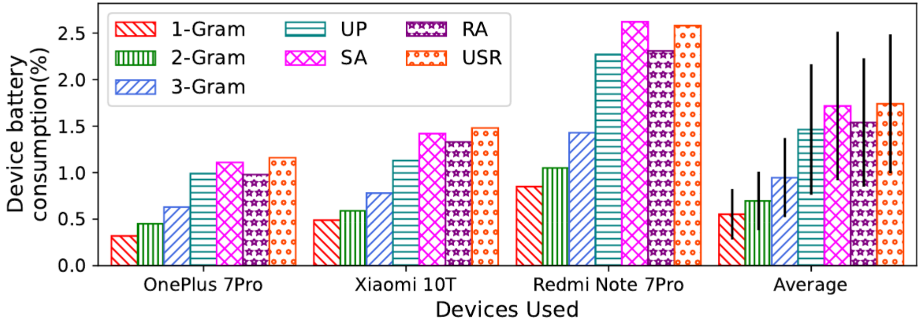
Average execution time on different device

2-Gram take ~5.32 seconds on OnePlus 7Pro, which is 2.13X and 2.53X faster than RA and SA, respectively

With 6.06 seconds of average execution time on all devices

Runtime Efficiency of Features

- ❑Use five different Apps
- ❑Extracts features on three different smartphones
- ❑Measures execution time
  - ➢ Per App
  - ➢ Average execution time
- ❑Device battery consumption



Device energy estimation

In OnePlus 7Pro, 2-Gram consumes 0.45% battery, which improves device energy by more than 2.1X against non Opcode based features

However, average energy consumption across all devices is 0.7%

Robustness Against Unseen/New

- Evaluated against known (training), unseen (evaluation set), new (Androzoo-2019)
- Also evaluated against 5 samples of Pegasus malware

Dataset	Precision	Recall	F1	FPR
Training Set	99.98	99.95	99.95	0.01
Evaluation Set	98.05	97.50	97.69	1.51
AndroZoo-2019	97.70	97.12	97.69	1.73
Pegasus (5)	--	100	--	--

Detect more than 97% new malware with FPR of 1.73%

Detect all samples of Pegasus malware

Evaluation Against Obfuscated Samples

- New obfuscated malware sample
  - Obfuscated AndroZoo-2019
  - Utilized Obfusapk Tool
  - 4993 unique sample in 6 category
- Evaluated on same samples
  - non-obfuscated (original)
  - and obfuscated

Category	#Samples	#Sample Detected		Drop (%)
		Original	Obfuscated	
Trivial	160	156	156	0
Renaming	570	554	554	0
Encryption	1135	1102	1092	0.53
Reflection	252	241	239	0.79
Code	2429	2358	2298	2.47
Mix	447	438	429	2.01
Overall	**4993**	**4849**	**4849**	**1.55**

No drop for trivial and renaming category

Maximum drop is 2.47% for code with average detection rate of 95.57%

Limitations

- Cannot detect malware
  - Malicious behavior in native code
  - Packed malware
  - Download malicious code from external source at runtime

Conclusion

With effective feature engineering, we have designed DeepDetect

Effectively detect more than 97% new malware with an FPR of 1.73%

Detect 95.57% of obfuscated malware

Analyze an App in ~5.32 seconds while consuming 0.45% of total device battery for 50 Apps

Thank You