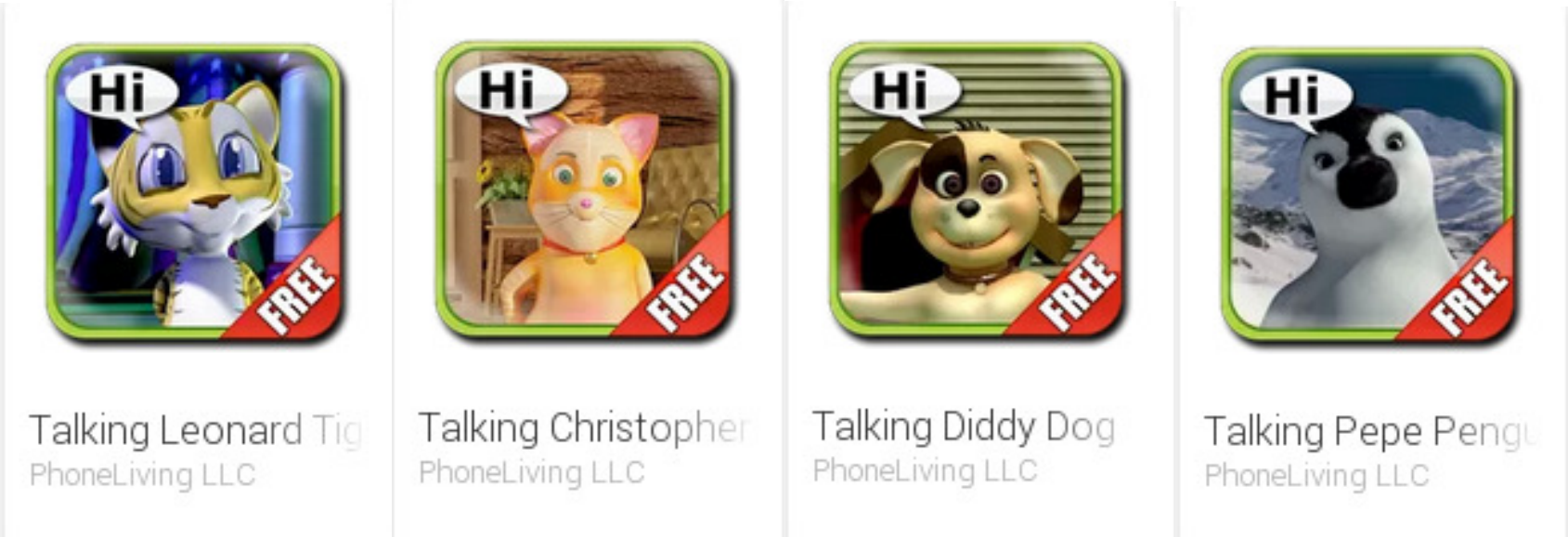
AnDarwin: Scalable Detection of Semantically Similar Android Applications

Jon Crussell, Clint Gibler, Hao Chen

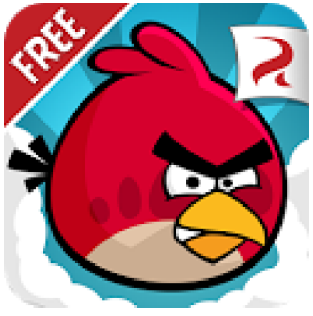
UC Davis Computer Security Lab

September 9th, 2013



Smartphones Abound







App Plagiarism

Prior work[1] showed apps are often plagiarized to siphon ad revenue

- Leveraged mobile ad traffic from major US provider

On average, developers lost:

- 14% Ad revenue
- 10% User base

[1] Gibler et. al, MobiSys 2013

Plagiarism Harms the App Ecosystem

Plagiarist Goals:

- Siphon ad revenue from legitimate apps
- Infect users with malware
- Phish users

Developers:

- Lose revenue and incentive to develop apps

Markets:

- Polluted search results

Users:

- Lose sensitive information
- Difficult to find useful, well-supported apps

Differences of App Plagiarism

- High-level bytecode
  - Significant modification of apps feasible[2]
  - Static analysis tractable
- Large scale
  - Previous work often focuses on localized problems:
    - Students' homeworks[3]
    - Synthetically created software "clones"
- Libraries embedded with app code
  - Difficult to differentiate without external knowledge

[2] Davis et. al, MobiSys 2013

[3] MOSS: `http://theory.stanford.edu/~aiken/moss/`

Previous Approaches: Not Robust

DroidMOSS (Zhou et. al, CODASPY 2012)

- Compares fuzzy hashes computed over the bytecode

Juxapp (Hanna et. al, DIMVA 2012)

- Compares k-grams of opcodes in disassembled bytecode

PiggyApp (Zhou et. al, CODASPY 2013)

- Compares APKs using vectors of used API methods

Previous Approaches: Not Scalable

DNADroid[4]

- Pair-wise app comparison based on PDGs
- Uses subgraph isomorphism to compare PDGs (NP-C)
- Relies on meta data to pick candidate app pairs

[4] Crussell et. al, ESORICS 2012

Definitions

App Cloning:

- Taking an existing app, modifying it and republishing it

Definitions

App Cloning:

- Taking an existing app, modifying it and republishing it

App Author:

- Developer account that publishes the app

Definitions

App Cloning:

- Taking an existing app, modifying it and republishing it

App Author:

- Developer account that publishes the app

App Plagiarism:

- App clones with different authors

Definitions

App Cloning:

- Taking an existing app, modifying it and republishing it

App Author:

- Developer account that publishes the app

App Plagiarism:

- App clones with different authors

Our goal: find clones of Android apps based on code similarity

Android

Android

- Smartphone operating system
- Open-source, developed primarily by Google
- Official and third-party markets for apps
- Over 900 million devices and 975,000 apps[5]

Android apps

- Developed primarily in Java, run on Dalvik VM
- Some ($< 10\%$) include native code

[5] `http://www.android.com`

AnDarwin

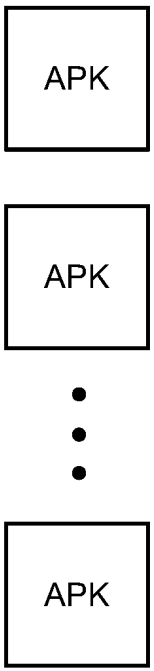
Goals:

- Scalably find clusters of apps with similar byte code
- Robust against plagiarist evasion techniques
- Low false-positive rate

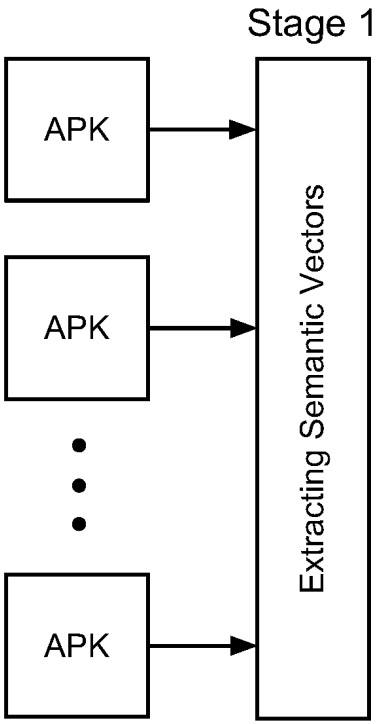
Non-goals:

- Determine similarity of native code
- Determine which app is the original

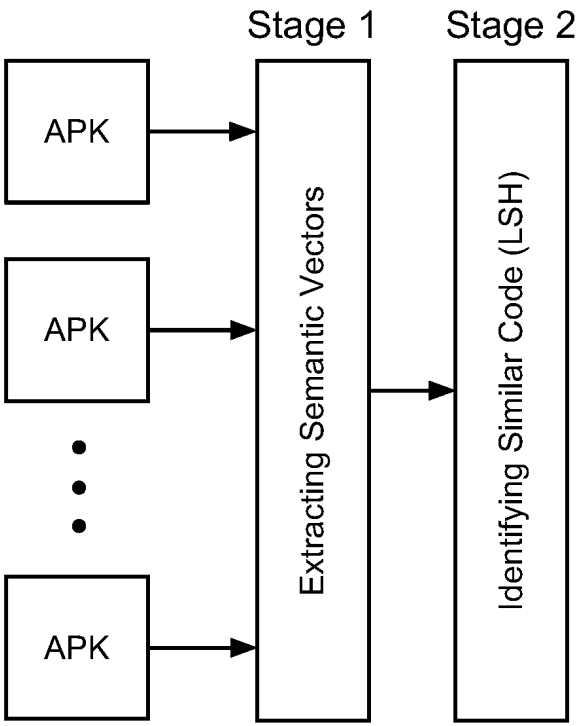
AnDarwin: Overview



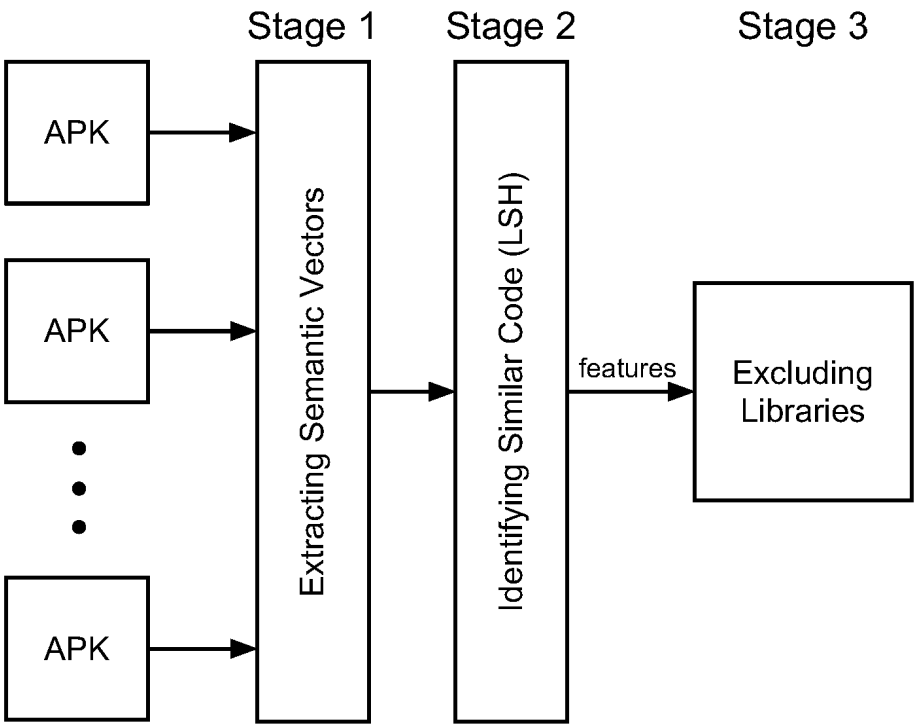
AnDarwin: Overview



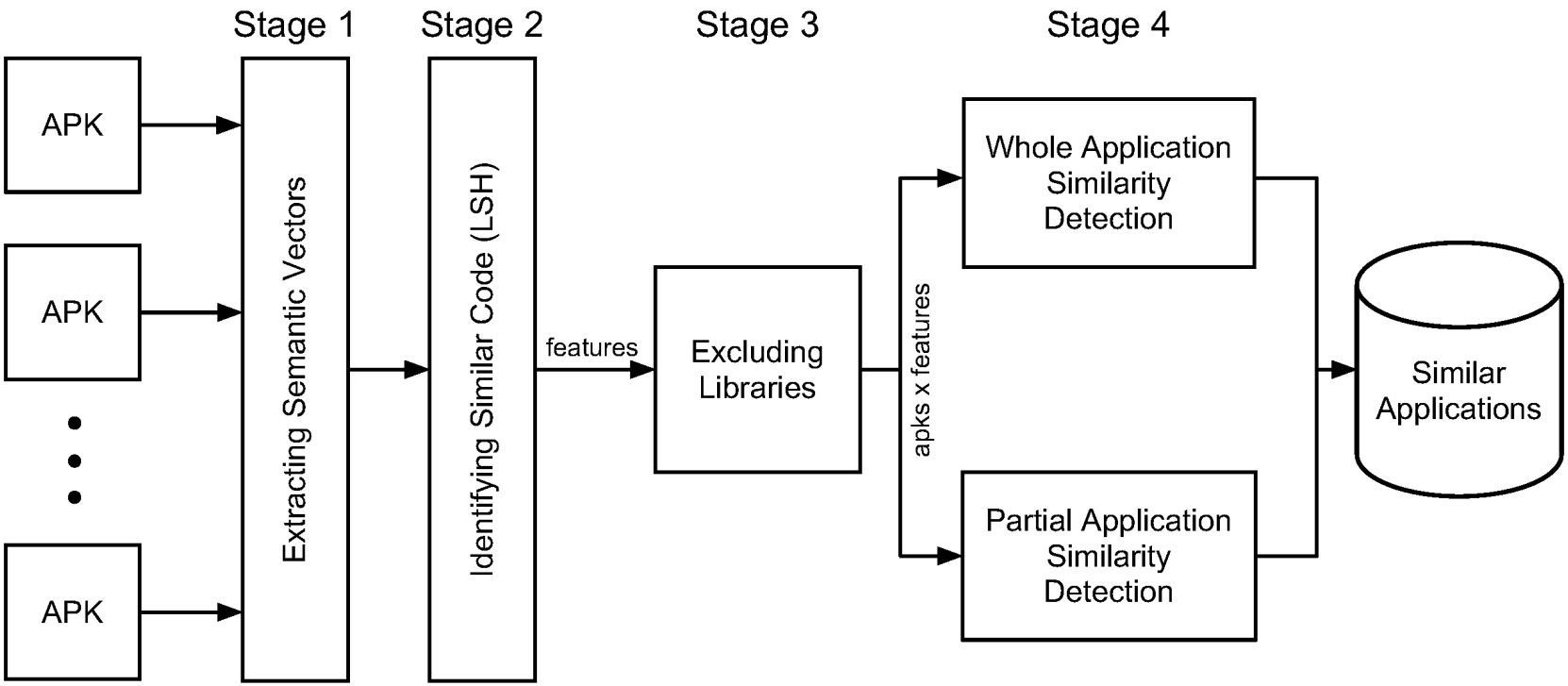
AnDarwin: Overview



AnDarwin: Overview



AnDarwin: Overview

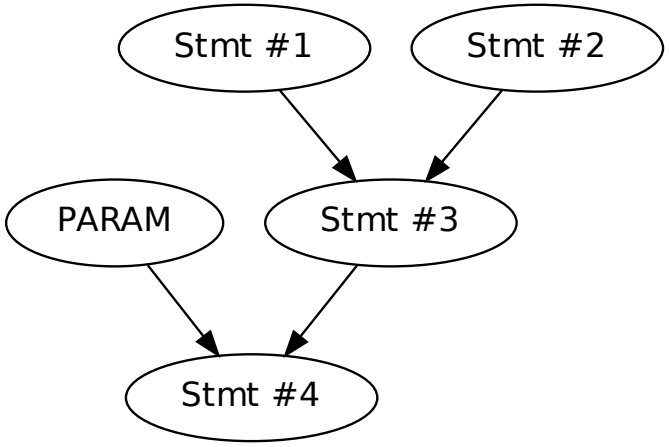


Program Dependency Graph (PDG)

PDG represents a method in a program

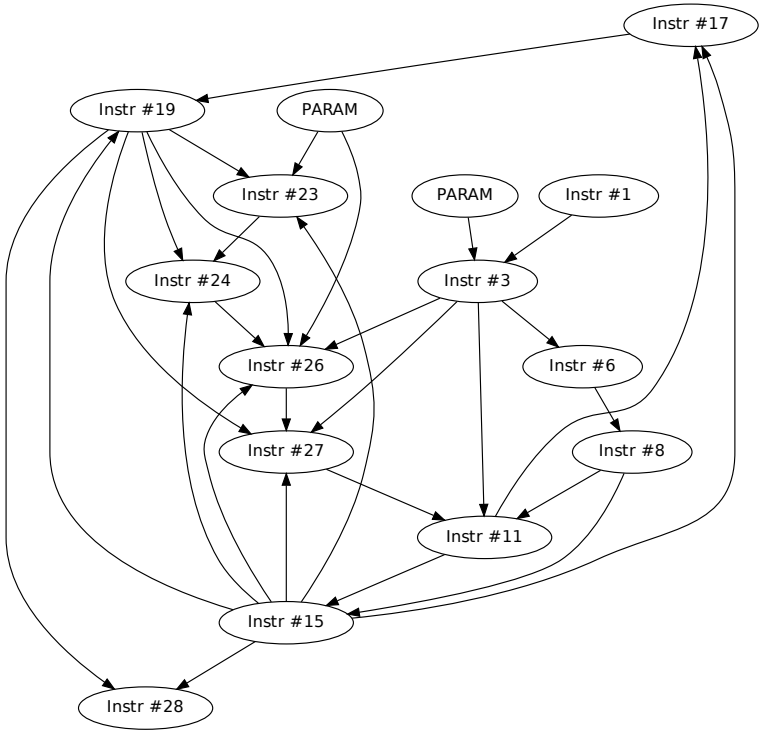
- Nodes: statement in the method, many different types:
  - Arithmetic operations
  - Branch, return, call
  - ...
- Edges: show data dependencies between statements

```
0: def foo(arg0)
1:   var x = 0
2:   var y = 1
3:   var z = x + y
4:   return z + arg0
```



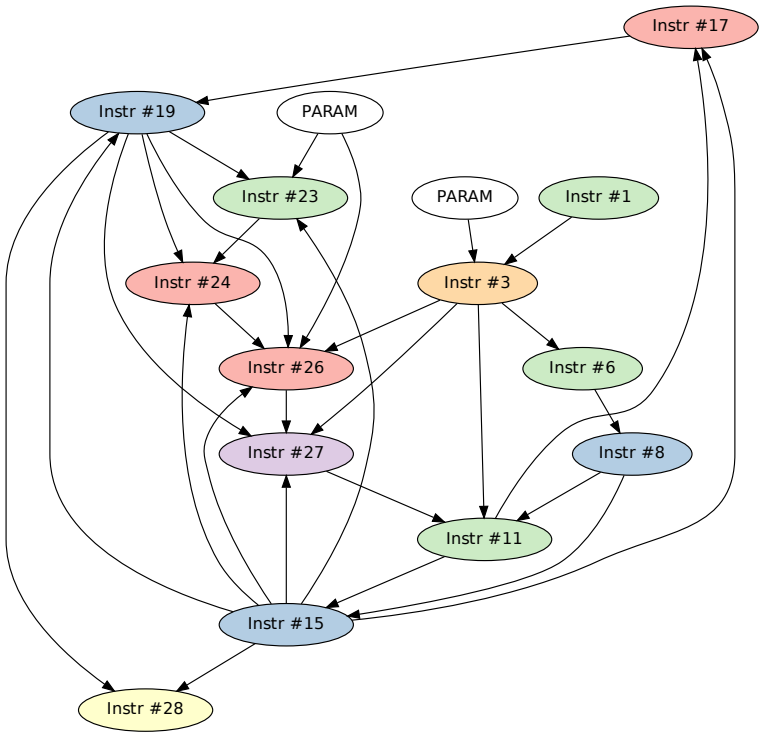
Semantic Vectors

PDG:



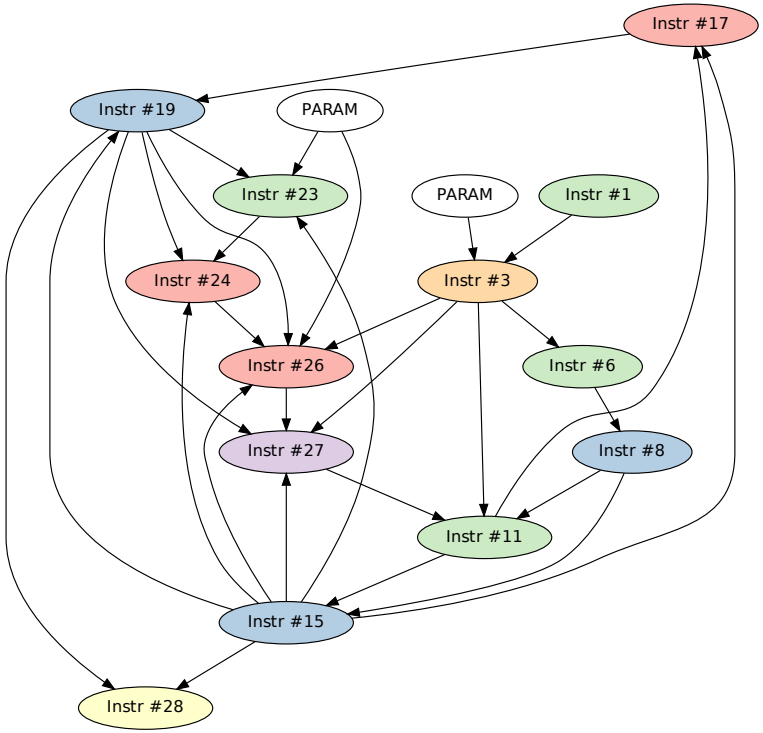
Semantic Vectors

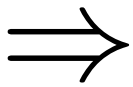
PDG:



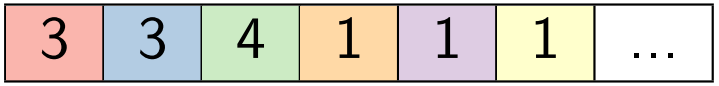
Semantic Vectors

PDG:





Semantic Vector:



Clustering Semantic Vectors

Methodology:

- Extract semantic vectors from all apps
  - Use dex2jar to convert Dalvik byte code to Java byte code
  - Use WALA[6] to construct PDGs
- Cluster vectors using Locality Sensitive Hashing

[6] T.J. Watson Libraries for Analysis

Clustering Semantic Vectors

Methodology:

- Extract semantic vectors from all apps
  - Use dex2jar to convert Dalvik byte code to Java byte code
  - Use WALA[6] to construct PDGs
- Cluster vectors using Locality Sensitive Hashing

Clusters of semantic vectors used as features to detect similar apps

- PDGs within clusters are likely similar

[6] T.J. Watson Libraries for Analysis

Locality Sensitive Hashing (LSH)[7]

A family of hash functions such that for any two points, p and q:

$$\text{if } ||p - q|| \leq R \text{ then } Pr[h(p) = h(q)] \geq P_1$$

$$\text{if } ||p - q|| \geq cR \text{ then } Pr[h(p) = h(q)] \leq P_2$$

$$P_1 > P_2$$

Used as approximation algorithm to find nearest-neighbors (NN):

- Approximate $\Rightarrow$ probability some NNs will be missed
- Preprocessing: compute hash values for all points
- Time complexity for single NN search: sublinear
- Form clusters by querying NNs of each point

[7] Andoni et. al, FOCS 2006

App Feature Matrix

		Apps						
		app_1	app_2	app_3	app_4	app_5	...	app_N
Features	f_1	✓			✓		...	
	f_2		✓				...	✓
	f_3	✓			✓		...	
	f_4			✓		✓	...	
	f_5		✓			✓	...	
	f_6		✓				...	✓
	f_7			✓		✓	...	
	f_8						...	✓
	...	...	...	...	...	...	...	...
	f_M	✓			✓		...	

App Feature Matrix

		Apps						
		app_1	app_2	app_3	app_4	app_5	...	app_N
Features	f_1	✓			✓		...	
	f_2		✓				...	✓
	f_3	✓			✓		...	
	f_4			✓		✓	...	
	f_5		✓			✓	...	
	f_6		✓				...	✓
	f_7			✓		✓	...	
	f_8						...	✓
	...	...	...	...	...	...	...	...
	f_M	✓			✓		...	

Observation: Matrix is sparse

Full App Similarity Detection

Goal: Find apps with many shared features

Full App Similarity Detection

Goal: Find apps with many shared features

		Apps						
		app_1	app_2	app_3	app_4	app_5	...	app_N
Features	f_1	✓			✓		...	
	f_2		✓				...	✓
	f_3	✓			✓		...	
	f_4			✓		✓	...	
	f_5		✓			✓	...	
	f_6		✓				...	✓
	f_7			✓		✓	...	
	f_8						...	✓
	...	...	...	...	...	...	...	...
	f_M	✓			✓		...	

Full App Similarity Detection

Goal: Find apps with many shared features

		Apps						
		app_1	app_2	app_3	app_4	app_5	...	app_N
Features	f_1	✓			✓		...	
	f_2		✓				...	✓
	f_3	✓			✓		...	
	f_4			✓		✓	...	
	f_5		✓			✓	...	
	f_6		✓				...	✓
	f_7			✓		✓	...	
	f_8						...	✓
	...	...	...	...	...	...	...	...
	f_M	✓			✓		...	

MinHash

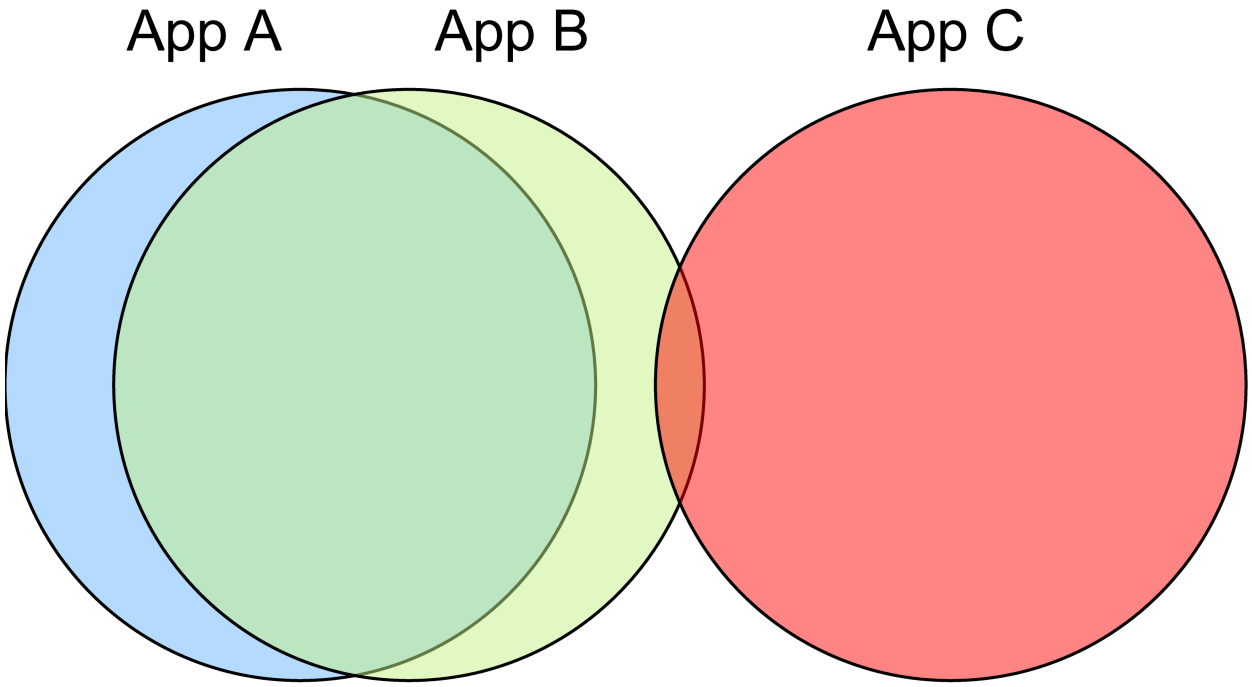
Broder et. al present MinHash to find similar sets[8]

- Member of LSH family
- Designed to estimate Jaccard coefficient of sets
- Requires a sparse feature matrix
- Time complexity: $O(N \log N)$

[8] Compression and Complexity of Sequences 1997

Problem:

Jaccard coefficient $\equiv J(A, B) = \frac{|A \cap B|}{|A \cup B|}$



Partial App Similarity Detection

Goal: Find apps that share common module of code

Partial App Similarity Detection

Goal: Find apps that share common module of code

		Apps						
		app_1	app_2	app_3	app_4	app_5	...	app_N
Features	f_1	✓			✓		...	
	f_2		✓				...	✓
	f_3	✓			✓		...	
	f_4			✓		✓	...	
	f_5		✓			✓	...	
	f_6		✓				...	✓
	f_7			✓		✓	...	
	f_8						...	✓
	...	...	...	...	...	...	...	...
	f_M	✓			✓		...	

Partial App Similarity Detection

Goal: Find apps that share common module of code

		Apps						
		app_1	app_2	app_3	app_4	app_5	...	app_N
Features	f_1	✓			✓		...	
	f_2		✓				...	✓
	f_3	✓			✓		...	
	f_4			✓		✓	...	
	f_5		✓			✓	...	
	f_6		✓				...	✓
	f_7			✓		✓	...	
	f_8						...	✓
	...	...	...	...	...	...	...	...
	f_M	✓			✓		...	

Library Code

Apps may contain one or more libraries

- Common library code should **not** be considered for clone detection

Library Code

Apps may contain one or more libraries

- Common library code should **not** be considered for clone detection

Methodology:

- Calculate number of apps each feature appears in
- Exclude features (rows of matrix) using threshold
  - Must be larger than expected number of clones of single app
  - Must not be too large to avoid missing less common libraries

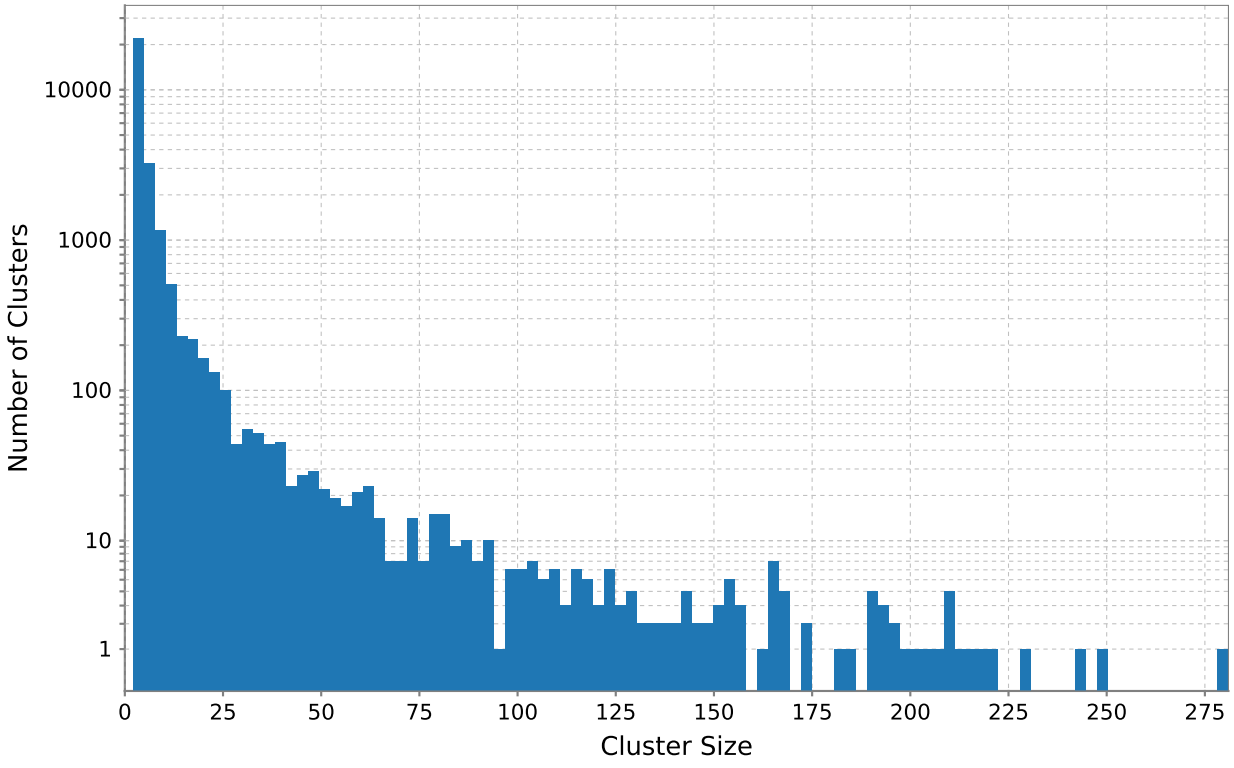
Dataset

We developed crawlers for Google Play and 16 Third-Party Markets

We downloaded a total of 265,000 free apps:

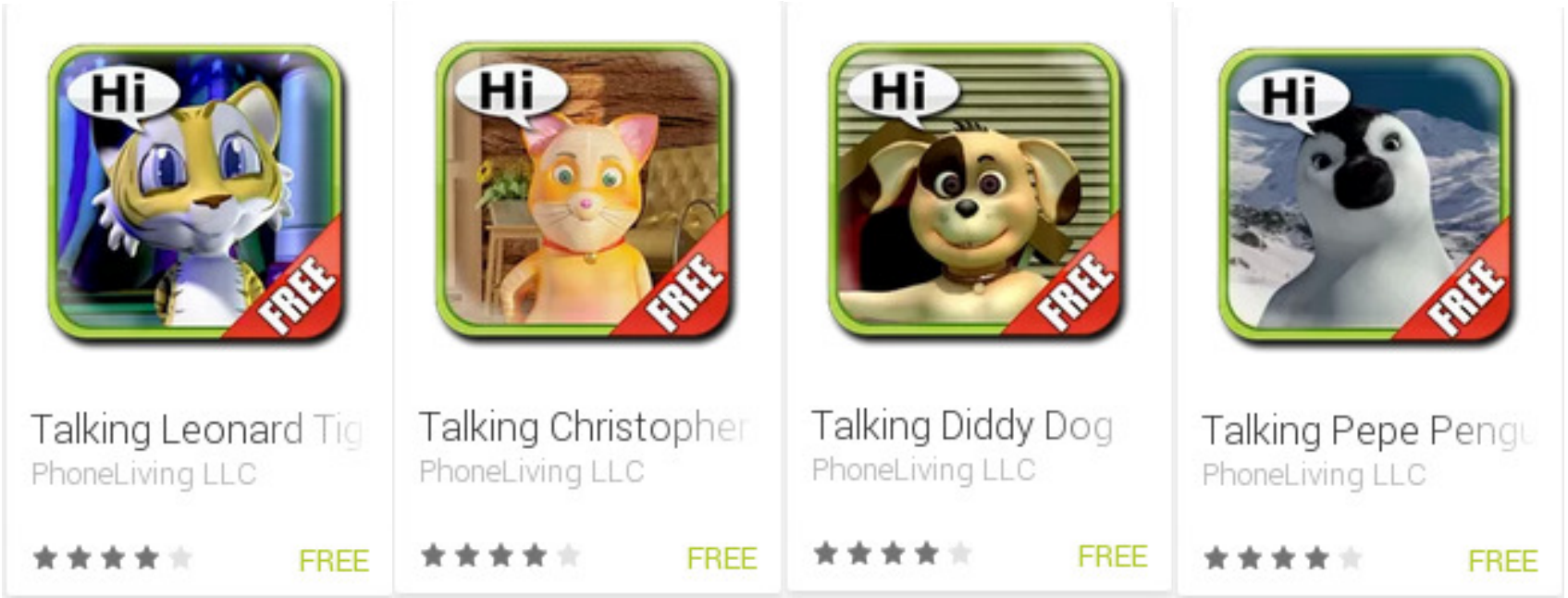
Full App Similarity Detection Results

28,000 clusters consisting of 150,000 apps in ten hours



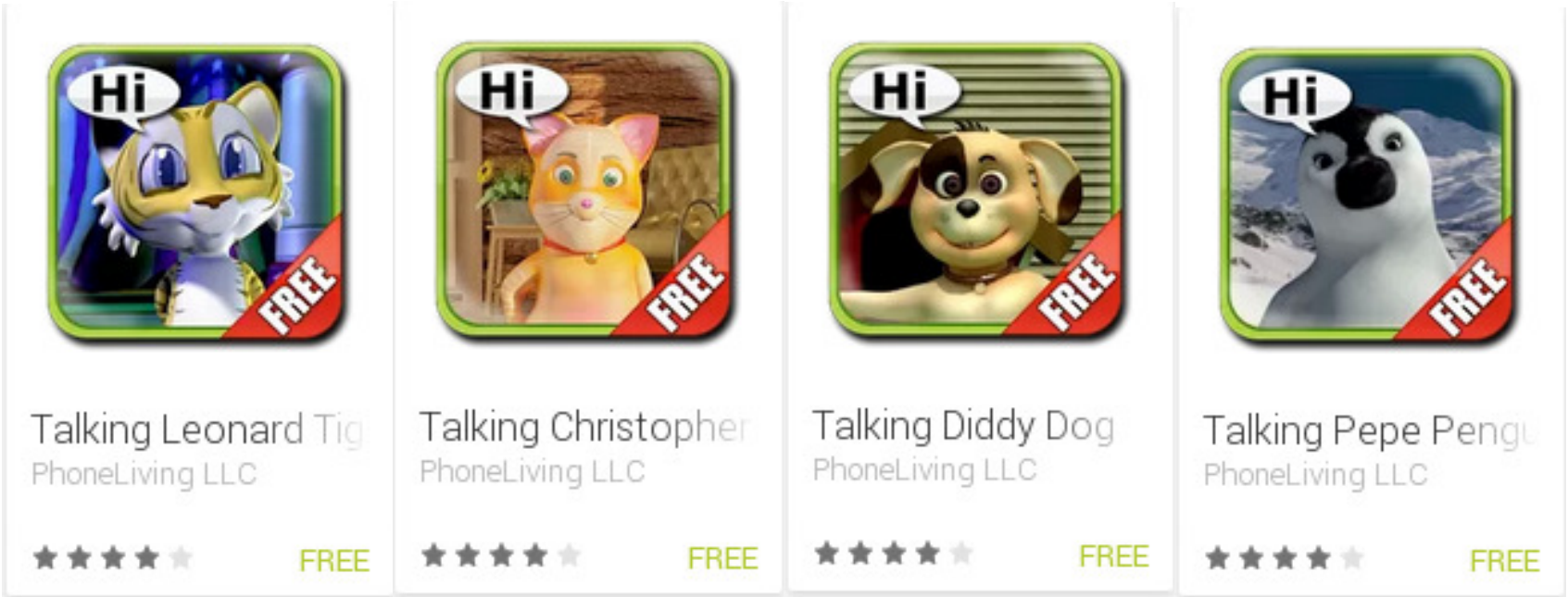
Finding: Rebranded Apps

Authors cloning their own apps to cater to different audiences:



Finding: Rebranded Apps

Authors cloning their own apps to cater to different audiences:



Total Number of Rebranded Apps: 36,106

Finding: Plagiarism Victims

Clusters of apps with multiple authors:

Market	**Developer**	**Package**	**Key**	**Ad Library**
Play	Rhythm Software	com.rhythm	fce850	admob
AndroidOnline	Rhythm Software	com.rhythm	fce850	admob
Freeware Lovers	Rhythm Software	com.rhythm	fce850	admob
Play	useful tools	com.uninstall	92cced	admob, airpush
Play	Android Tools	com.soft	92cced	admob

Finding: Plagiarism Victims

Clusters of apps with multiple authors:

Market	**Developer**	**Package**	**Key**	**Ad Library**
Play	Rhythm Software	com.rhythm	fce850	admob
AndroidOnline	Rhythm Software	com.rhythm	fce850	admob
Freeware Lovers	Rhythm Software	com.rhythm	fce850	admob
Play	useful tools	com.uninstall	92cced	admob, airpush
Play	Android Tools	com.soft	92cced	admob

Total Number of Plagiarism Victims: 4,295

Verification

Leverage DNADroid

- Robust tool based on PDGs
- Scalability limited
- Experimental False Positive Rate: 0%
    - Manually compared 140 app pairs

Tested 25,000 pairs from 6,000 random clusters

- Experimental False Positive Rate: 3.73%

Conclusion

Contributions:

- Developed approach for detecting similar apps
  - Analyzes apps at an unprecedented scale
  - Considers both full and partial similarity
  - Uses only code similarity without relying on metadata
  - Low false positive rate
- Use cases:
  - Single author: rebranded apps
  - Multiple authors: plagiarized apps

Coming soon: `sherlockdroid.com`