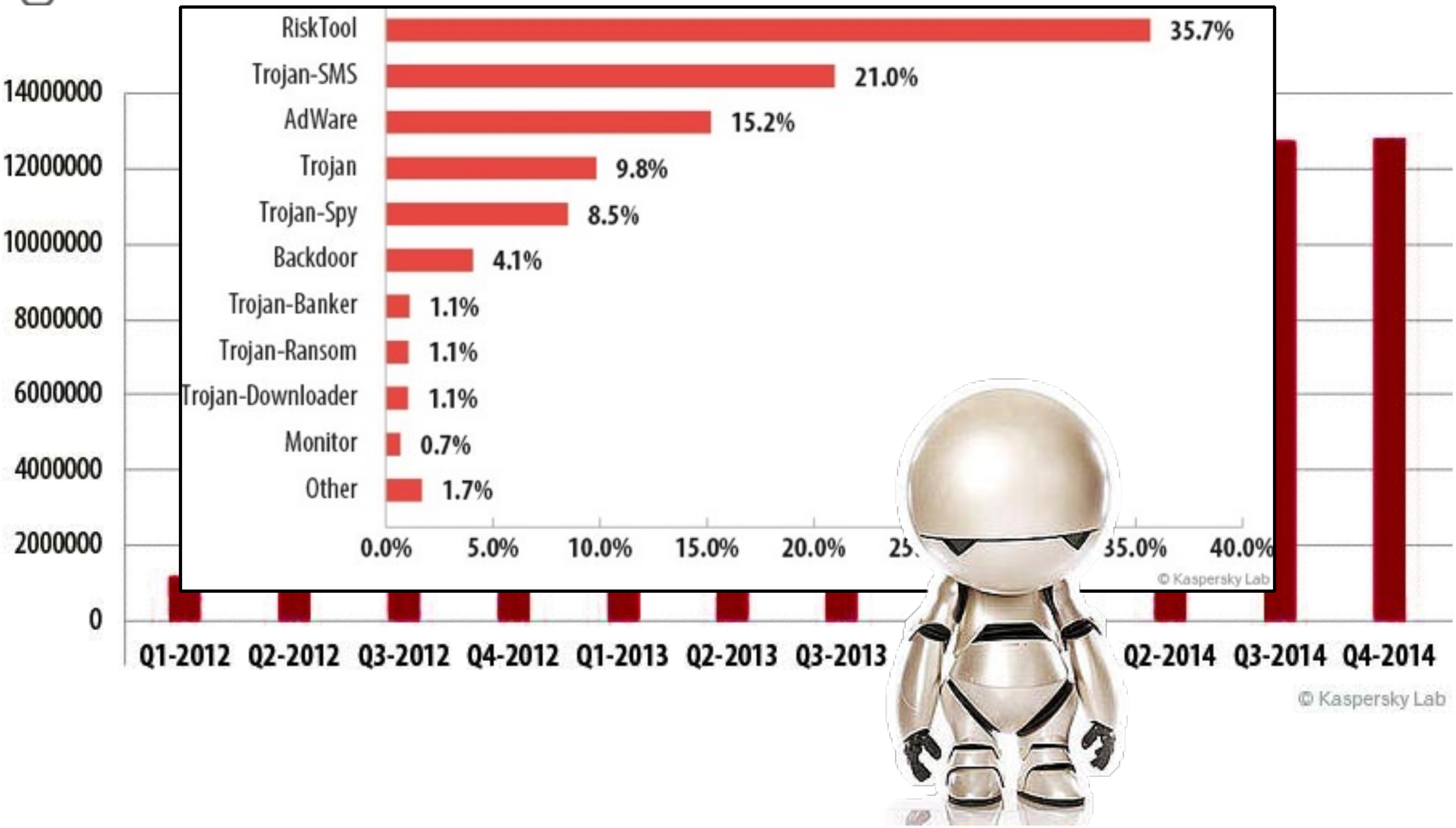
MARVIN: Efficient And Comprehensive Mobile App Classification
Through Static and Dynamic Analysis

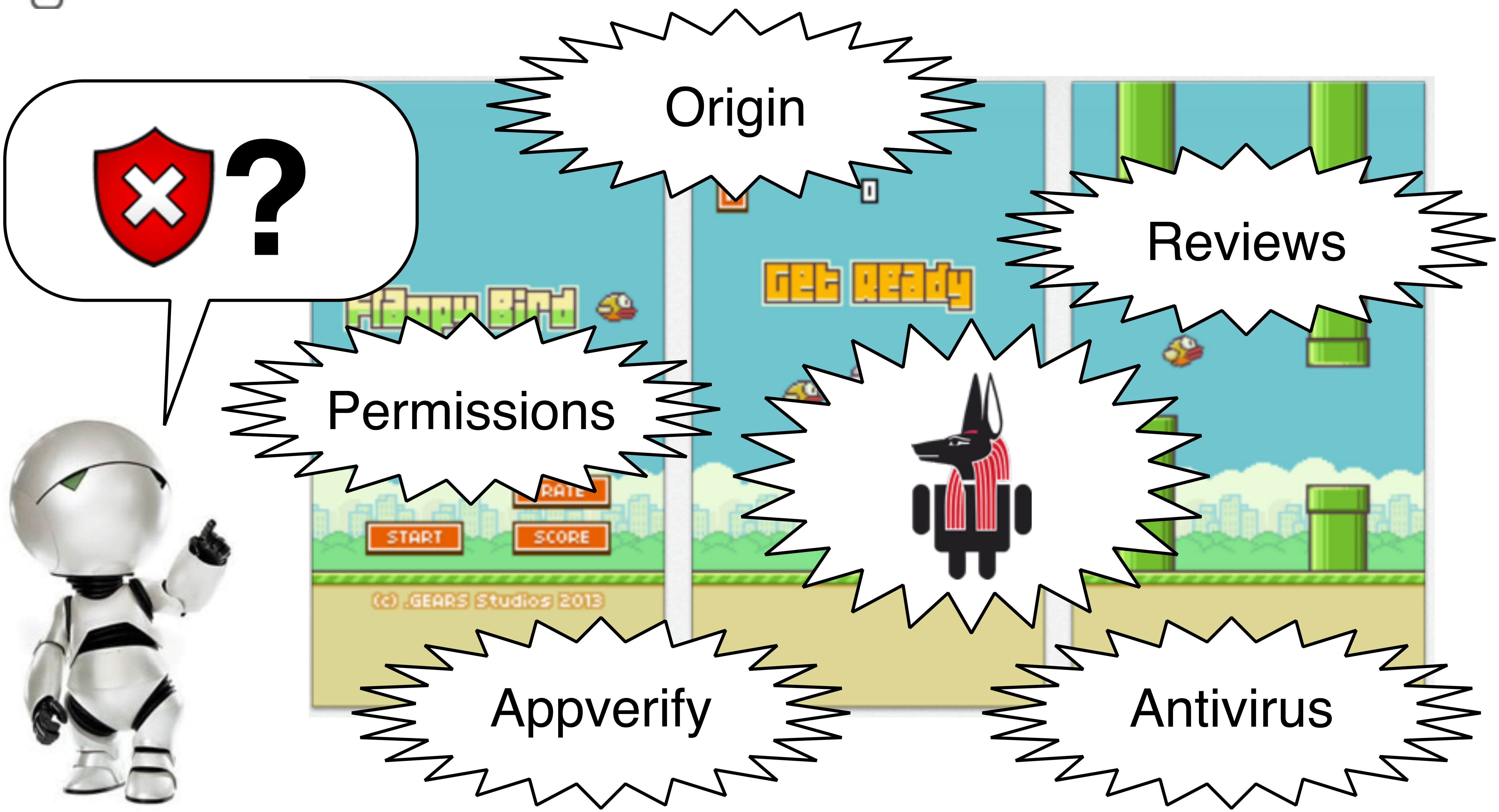
Martina Lindorfer, Matthias Neugschwandtner, Christian Platzer

SBA Research, Vienna, Austria
IBM Research, Zurich, Switzerland
International Secure Systems Lab, Vienna University of Technology, Austria

State of Mobile Malware

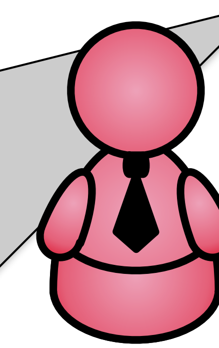


Real or Fake Flappy Bird App?



Use Cases


```
SELECT * FROM
   apps
 WHERE
   malice_score > 5.0
 AND
   has_nw_traffic = True
...
```



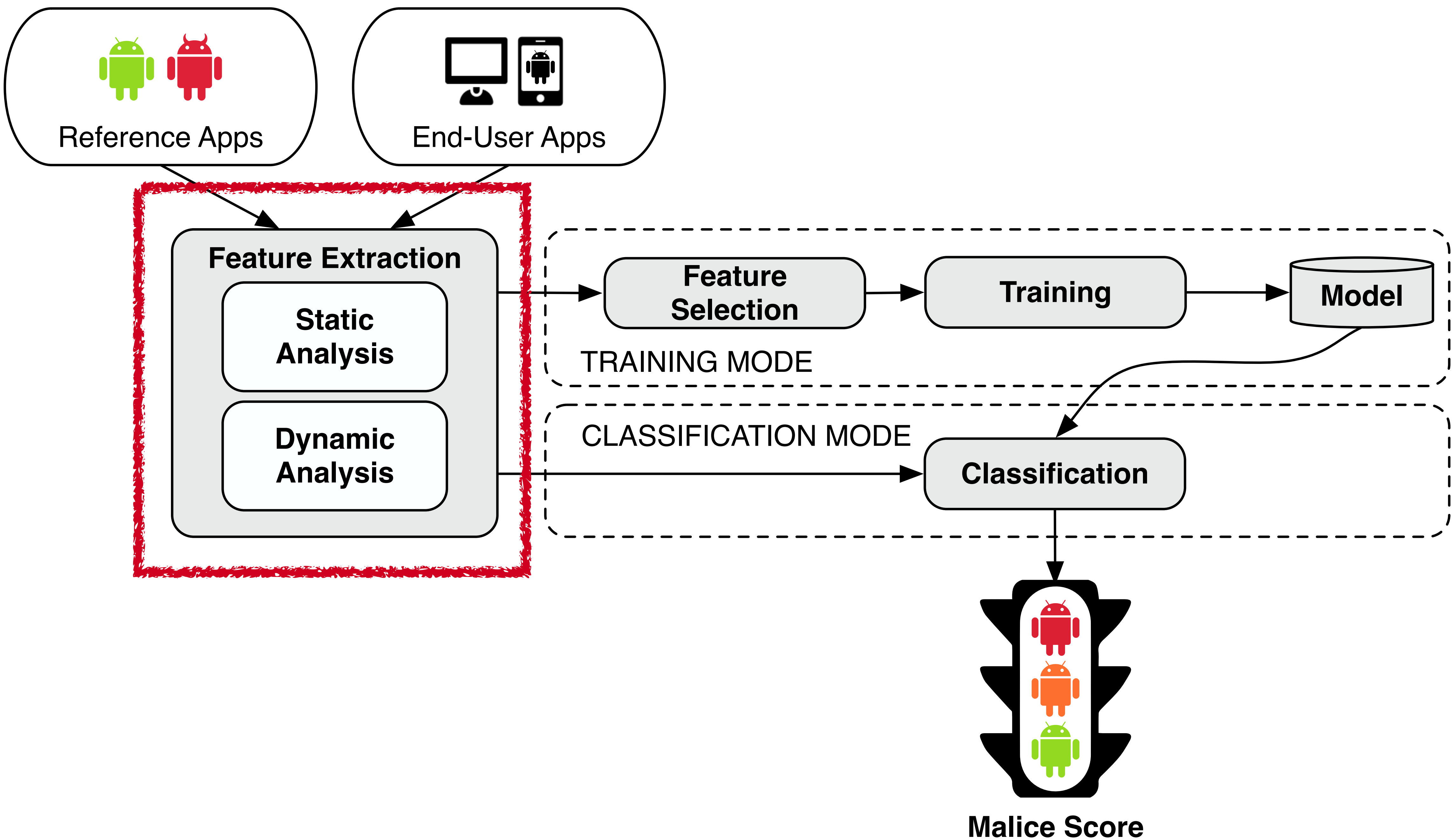
Outline

- **App Classification**
- Evaluation
- Future Work and Conclusion

Classification Goals

- Use machine learning to classify Android apps
- Address grey area between malware and goodware
  - Provide user with a malice score from 0 to 10
- Address drawbacks of related work
  - Only consider static features
  - Trained and evaluated on very small dataset
  - Do not account for history of dataset
- Long-term practicality through efficient retraining

System Overview



Static vs. Dynamic Analysis

- Static analysis…
  - code is not executed
  - all possible branches can be examined (in theory)
  - quite fast

- Problems of static analysis…
  - undecidable in general case, approximations necessary
  - obfuscated & packed code
  - self-modifying code
  - code (down)loaded at runtime

Static vs. Dynamic Analysis

- Dynamic analysis…
  - code is executed
  - sees behavior that is actually executed
  - sees dynamically loaded code

- Problems of dynamic analysis…
  - in general, single path is examined
  - analysis environment possibly not *invisible*
  - scalability issues

➡ Combine features from static AND dynamic analysis

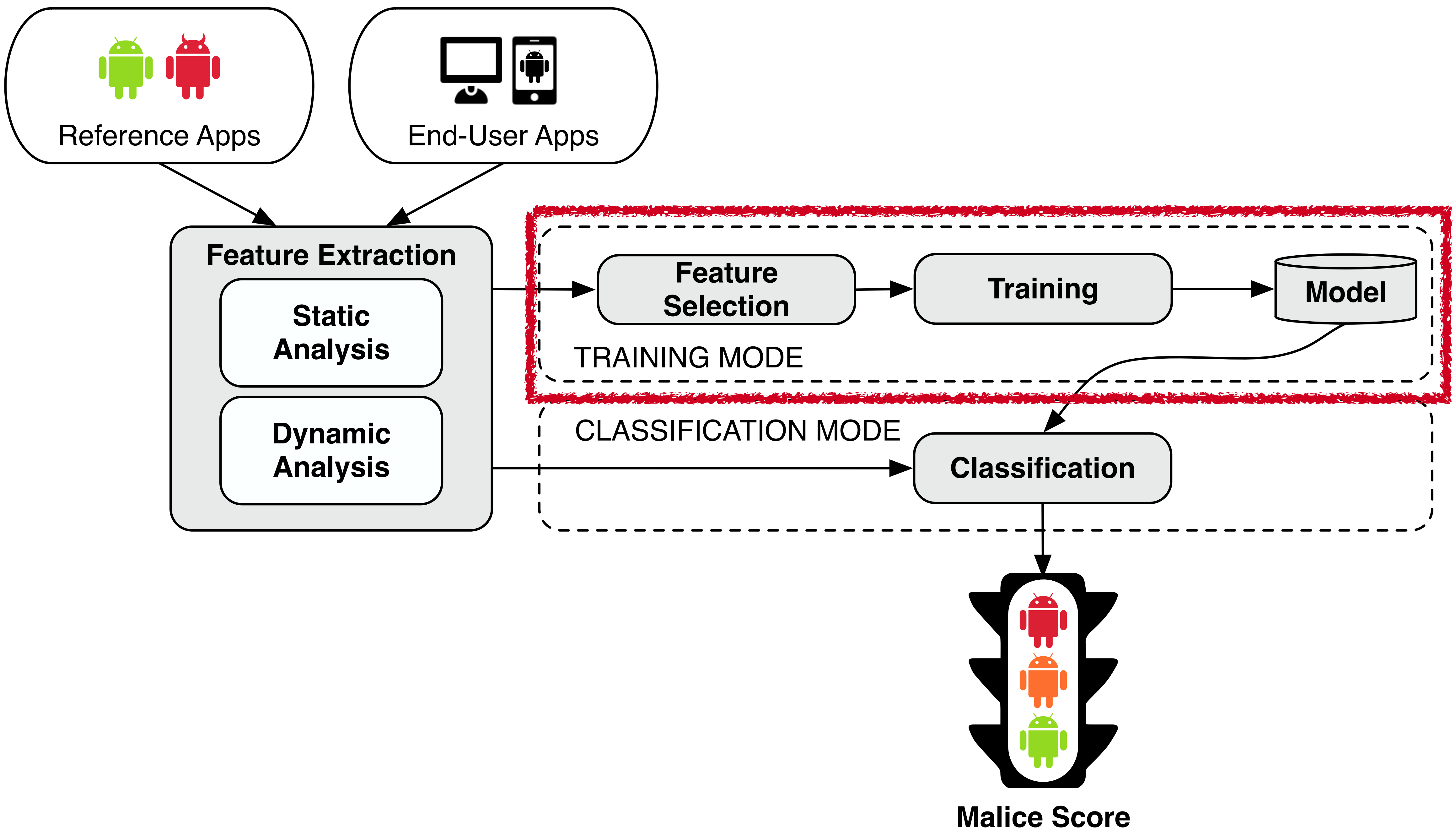
Feature Extraction in ANDRUBIS

- Extended ANDRUBIS app analysis sandbox [BADGERS2014]

- Static Analysis
  - Required/Used permissions, Activities, Services, Receivers, …
  - Certificate metadata (owner, validity, …)
  - Included libraries
- Dynamic Analysis
  - File/network/phone activities
  - Cryptographic operations
  - Leaked data
  - Loading of dynamic code (DEX and native code)

- Output: Sparse feature vector of binary features

System Overview



Classification Challenges

- High-dimensional feature space
  - Explicit feature selection:
    Order features by discriminative power (F-Score)
  - Implicit feature selection:
    Order features by weights from classifier
- Sparse data
- Grey area between malware and goodware
  - Classifier outputs probability that sample belongs to class
  - Scale probability in interval [0,10]
- Performance

➡ Experiments with SVM and linear classifier with different regularization methods

Outline

- App Classification
- **Evaluation**
- Future Work and Conclusion

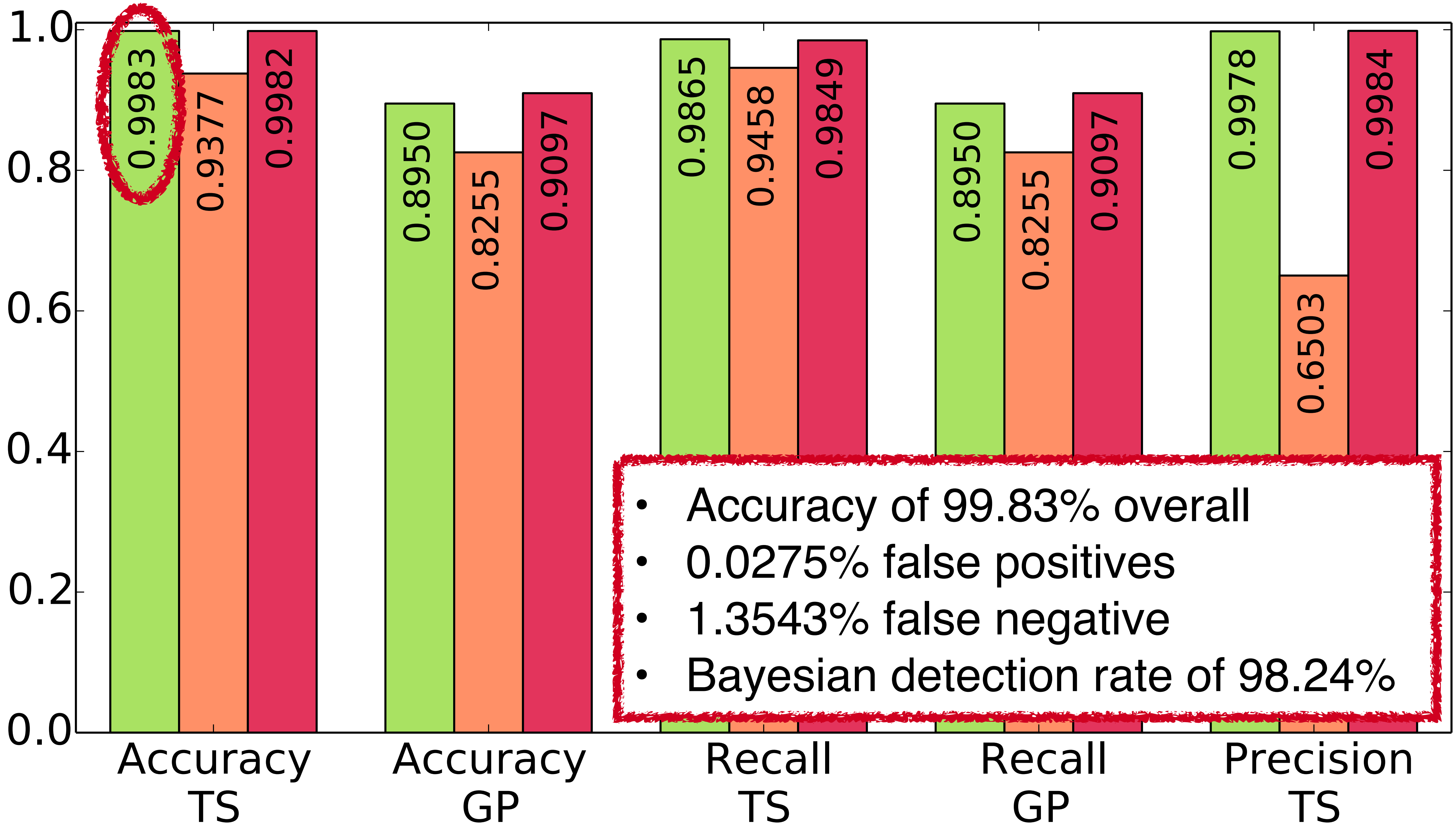
Evaluation Overview

- Large training and testing sets
  - Set of goodware apps from Google Play Store
  - Set of known malware with AV labels from VirusTotal
  - 135,823 unique Android applications (15,741 known malware)

Goals:

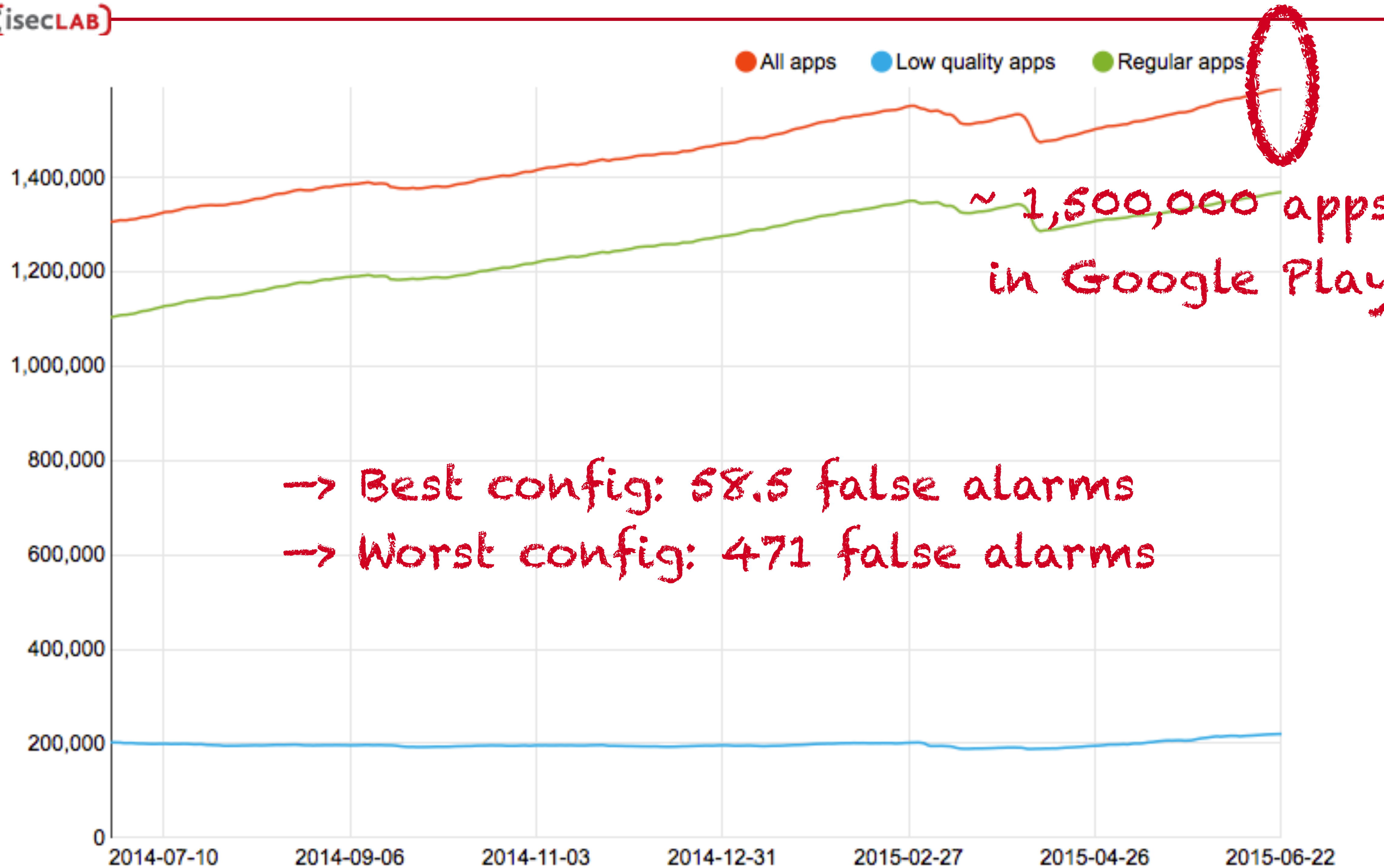
1. Evaluate accuracy of different classifiers
2. Evaluate performance (market-scale classification)
3. Evaluate long-term practicality
   - History of samples in dataset matters [ESSoS2015]
   - Estimate retraining intervals and efficiency
4. Evaluate most distinguishing features

Classification Accuracy

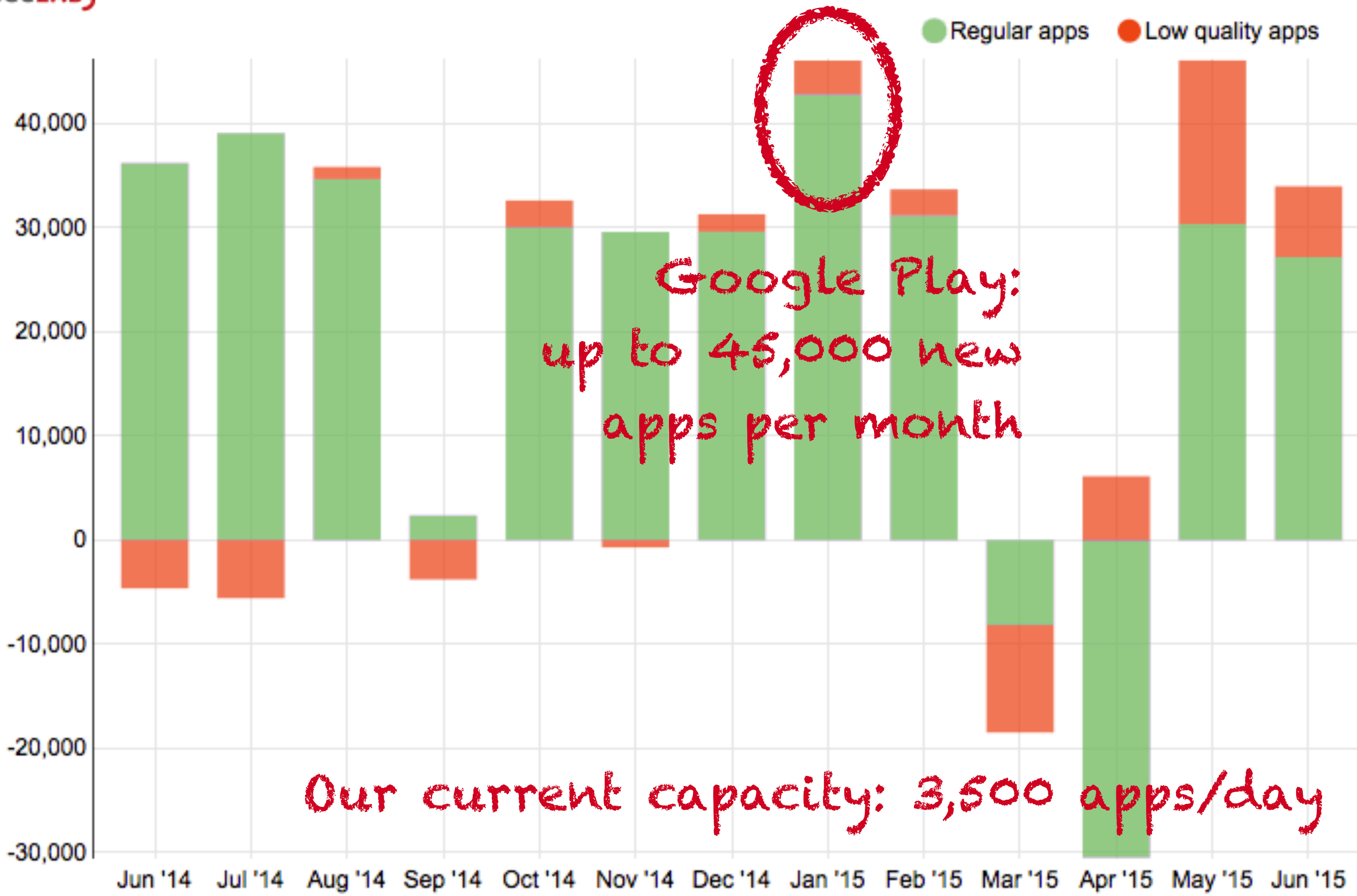


- Accuracy of 99.83% overall
- 0.0275% false positives
- 1.3543% false negative
- Bayesian detection rate of 98.24%

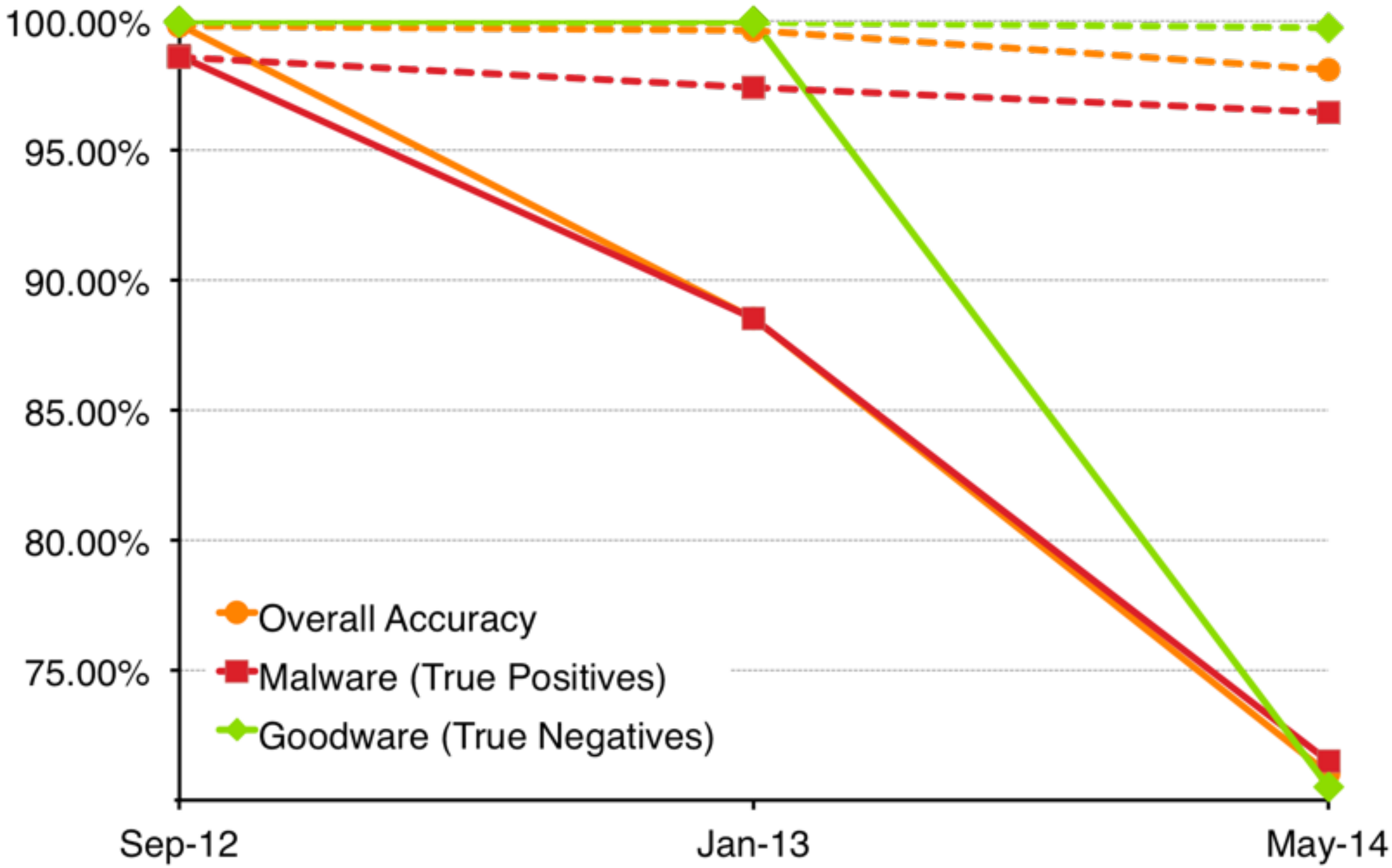
Market-Scale Classification



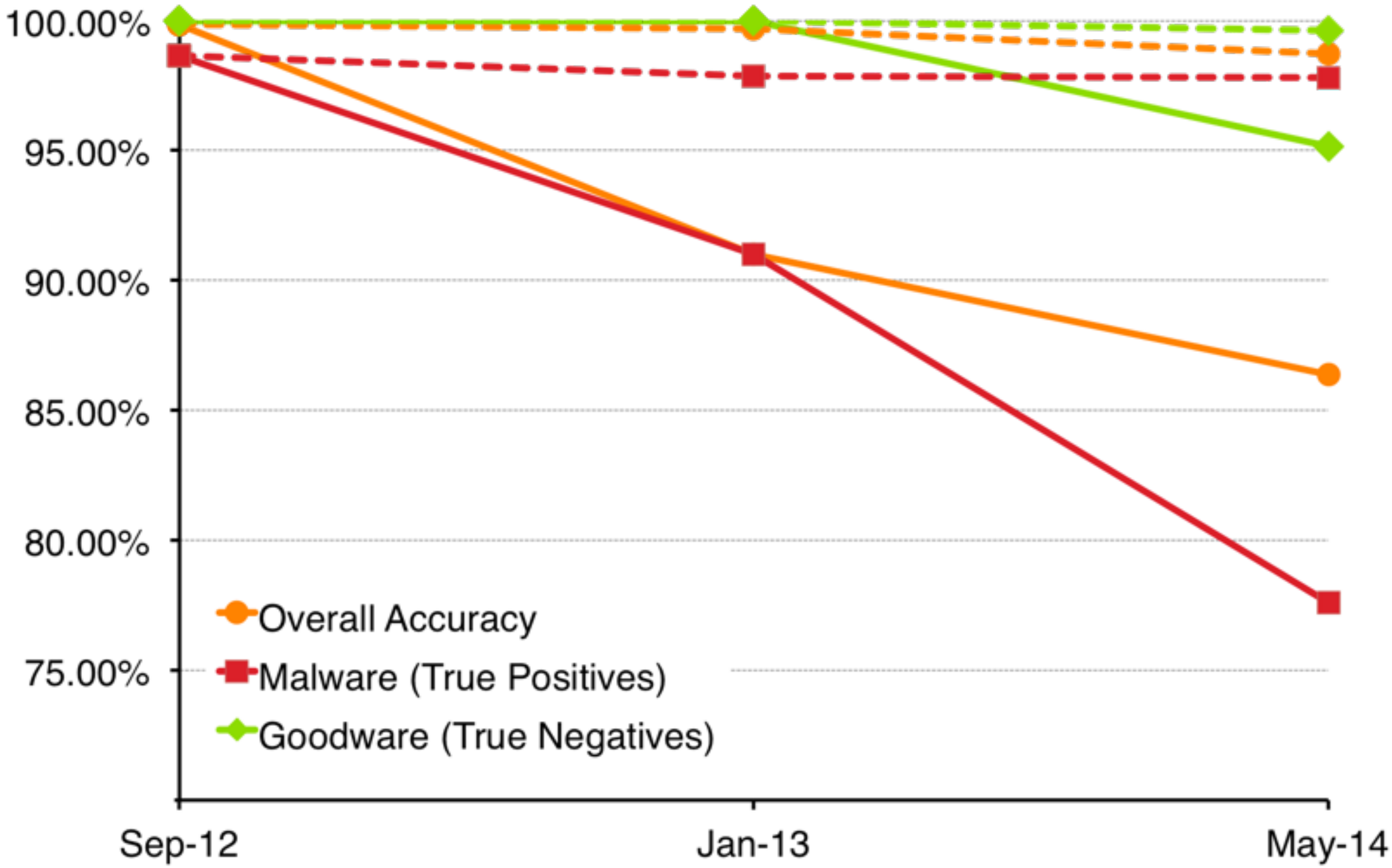
Market-Scale Classification



Long-Term Practicality (Less Features)



Long-Term Practicality (More Features)



Distinguishing Features

- Gain insights into classification through F-Score/feature weights

- Features most relevant for classification of malware:
  - Required/Used permissions
  - Certificates
  - SMS-related features
  - Information leaks
  - Dynamic code loading
  - Network activity and contacted hosts

Outline

- App Classification
- Evaluation
- **Future Work and Conclusion**

Future Work

- Dynamic features++
  - System-level events from native code analysis
  - More intelligent, user-like UI interactions
- Static features ++
  - Meta info in app markets from AndRadar [DIMVA2014]
- Interception of app installation process
- Defence against analysis evasion (arms race)

Conclusion

- Classification of Android apps using machine learning
  - Based on static AND dynamic features
  - Represented as a malice score

- Large-scale evaluation on over 135,000 apps
  - Correctly classifies 98.24% of malware samples
  - Very low positives of < 0.04%
  - Retraining to maintain accuracy

- Publicly available for submissions through web interface and dedicated mobile app

Questions?

email mlindorfer@iseclab.org
andrubis@iseclab.org

twitter @iseclaborg

http http://www.iseclab.org/people/mlindorfer
https://anubis.iseclab.org
https://play.google.com/store/apps/details?id=org.iseclab.andrubis

References

[BADGERS2014] Martina Lindorfer, Matthias Neugschwandtner, Lukas Weichselbaum, Yanick Fratantonio, Victor van der Veen, Christian Platzer
Andrubis - 1,000,000 Apps Later: A View on Current Android Malware Behaviors
International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS), 2014.

[ESSoS2015] Kevin Allix, Tegawendé F. Bissyandé, Jacques Klein, Yves Le Traon
Are Your Training Datasets Yet Relevant?
International Symposium on Engineering Secure Software and Systems (ESSoS), 2015.

[DIMVA2014] Martina Lindorfer, Stamatis Volanis, Alessandro Sisto, Matthias Neugschwandtner, Elias Athanasopoulos, Federico Maggi, Christian Platzer, Stefano Zanero, Sotiris Ioannidis
AndRadar: Fast Discovery of Android Applications in Alternative Markets
Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA), 2014.