OS Fingerprinting and Tethering Detection in Mobile Networks

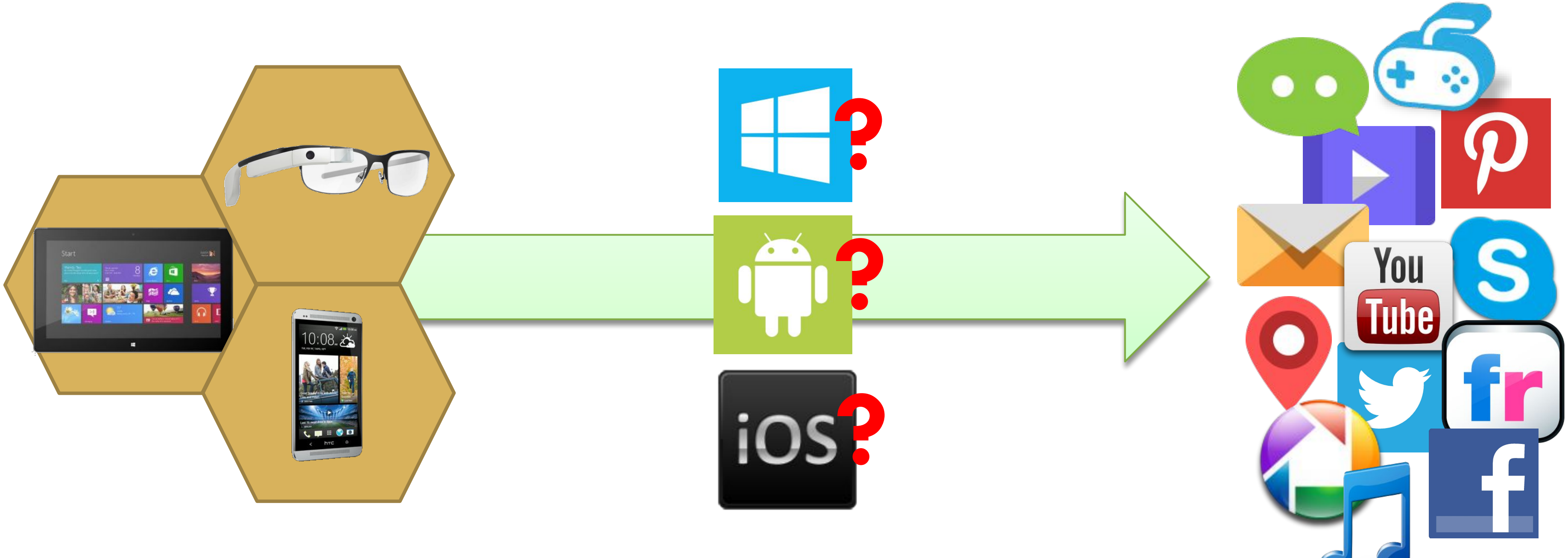
Yi-Chao Chen
The University of Texas at Austin

Joint work:
Yong Liao[‡], Mario Baldi[‡], Sung-Ju Lee[‡], Lili Qiu[†]
Narus Inc. [‡], The University of Texas at Austin[†]

IMC 2014

Mobile OS Fingerprinting

- Problem statement
  - Infer what operating system a device is running by analyzing the packets it's generating.
  - Tethering detection: identify mobile devices which are sharing the Internet access



Importance

- Tethering detection
  - Billing for shared access in mobile networks
- Security
  - Policy enforcement in enterprise networks

Existing Works

Application	• HTTP user agent [P0f], DHCP options [Satori]
Transport	• TCP handshake, timeout, MTU, flags, init seq. number [P0f, NMap, VEYSET02, PAM04, RAID03], TCP Timestamp [INFOCOM99, IMW02]
Network	• IP TTL, ID, dest address [P0f, PAM04]
Link	• 802.11 MAC fields, SSID, frame size [MOBICOM07]

Limitation of Existing Works

- Existing works focus on the Internet traffic
- Mobile networks impose new challenges:
  - Dynamic frequency due to power saving
    - Clock skew, boot time estimation, …
  - Short connections
    - TCP flavors, initial sequence number, …
  - Features might have changed in mobile OSes
    - TCP MTU, IP flags, …

Approach

- Identify features to fingerprint mobile device OSes
- Detect tethering
  - **Clock frequency stability**, boot time estimation
  - IP Time-to-Live, ID Monotonicity
  - TCP timestamp option, **window size scale option, timestamp monotonicity**
- Combine multiple features
- Quantify the performance
  - Individual and combined features
  - OS fingerprinting and tethering detection

Dataset

- Lab trace
  - 56 mobile user traces
    - 14 Android phones and tablets traces
      - Samsung Galaxy S5, HTC Ones, HTC Inspire phones, Google Nexus 10 tablet
    - 10 iOS traces
      - iPhone 4s, iPhone5s, iPad 2, iPod Touch
      - iOS 5.1.1, iOS 6.1
    - 32 Windows laptops traces
      - running Windows XP or Windows 7
  - Each capture lasts 10~30 minutes

Other Datasets

Trace	Time	Duration	# IPs
Lab Trace	Oct. 2013	2 hours	56
SIGCOMM08 Trace	Aug. 2008	1 day	223
OSDI06 trace	Nov. 2006	1 day	292

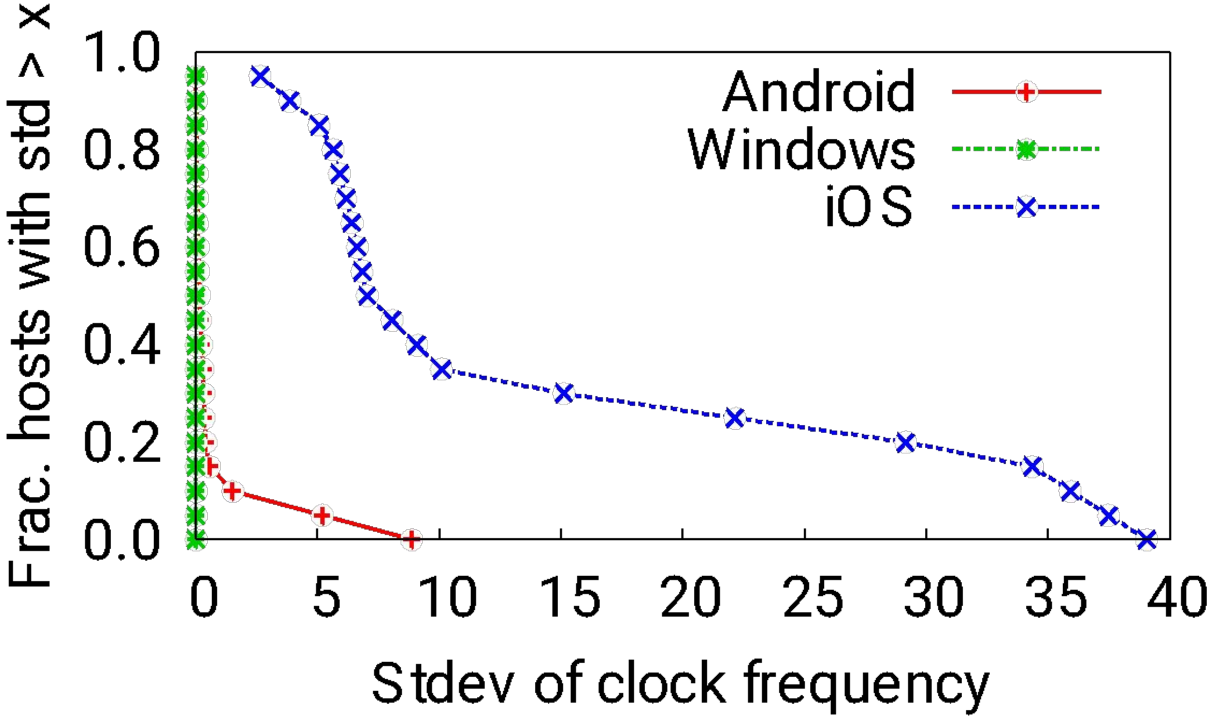
Features

- **Clock Frequency**

$$freq = \frac{timestamp_i - timestamp_1}{rcv\_time_i - rcv\_time_1}$$

  - Th dows,
  - bu

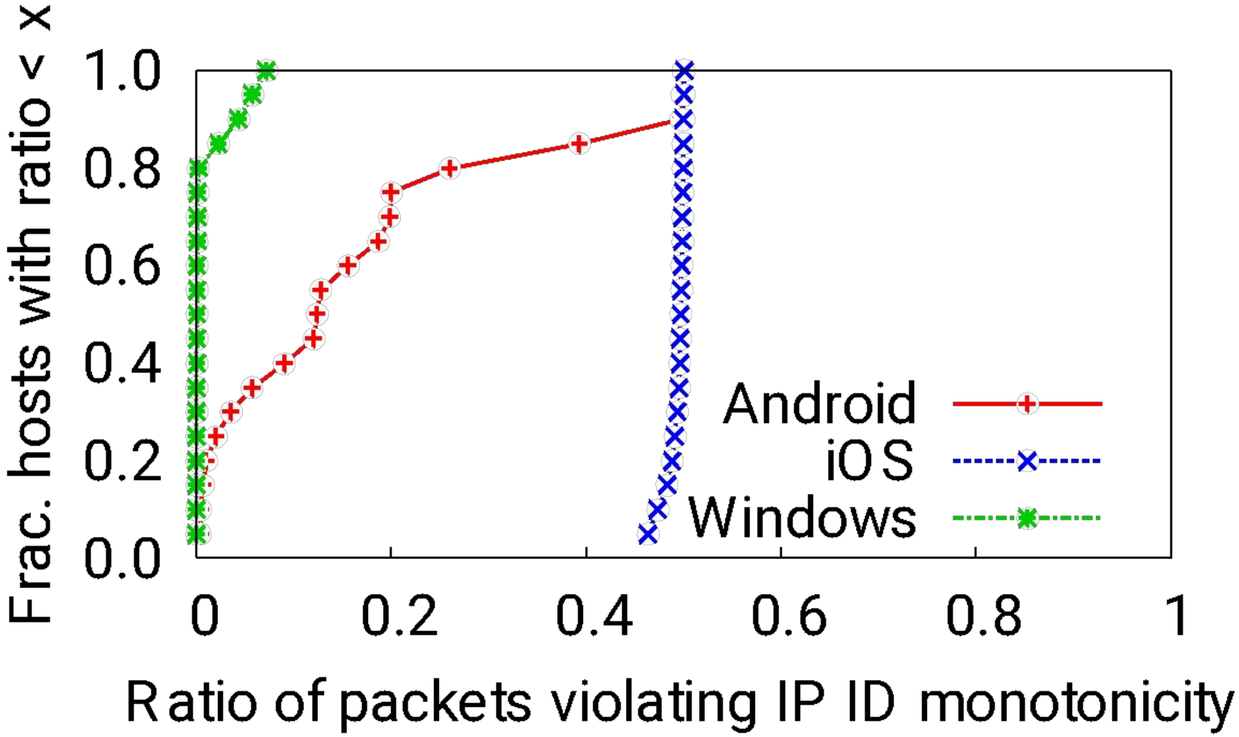
High clock frequency std. suggests **iOS**



Features

- **IP ID Monotonicity**
  - ically

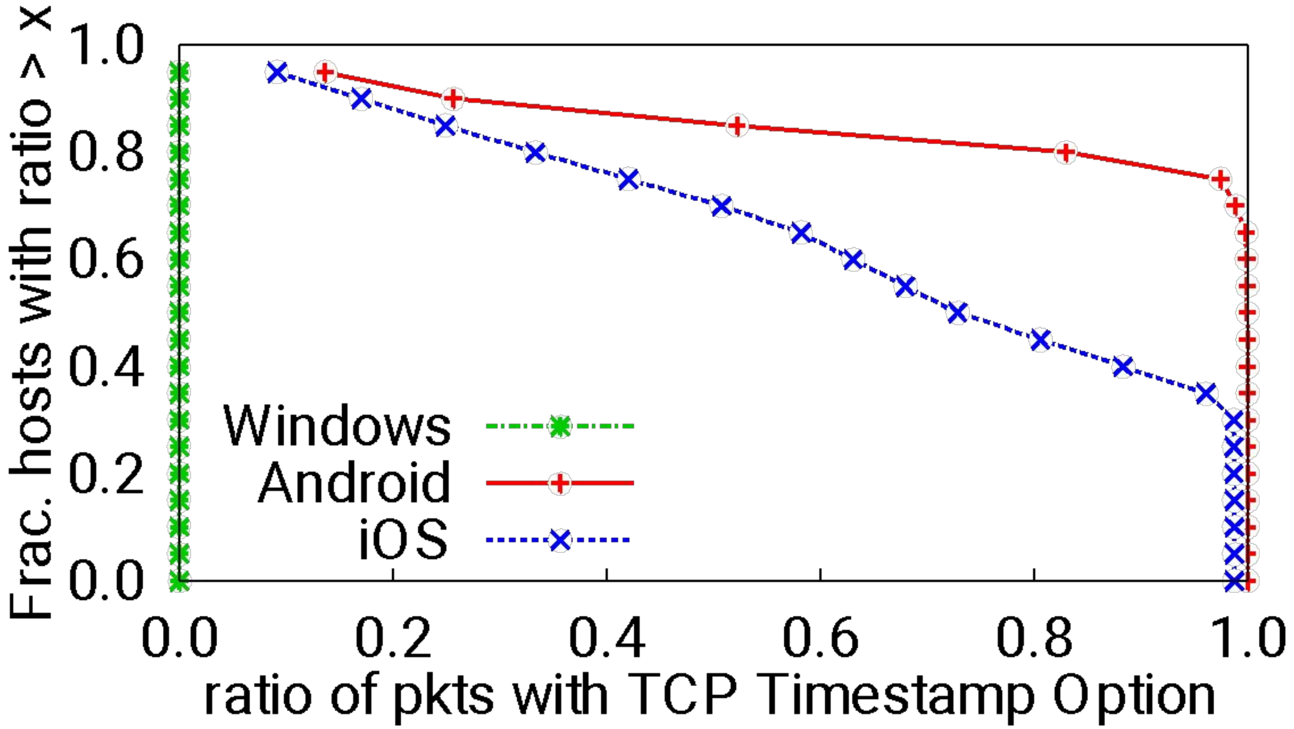
High violation ratio suggests **iOS;**
low violation ratio suggests **Windows.**



Features

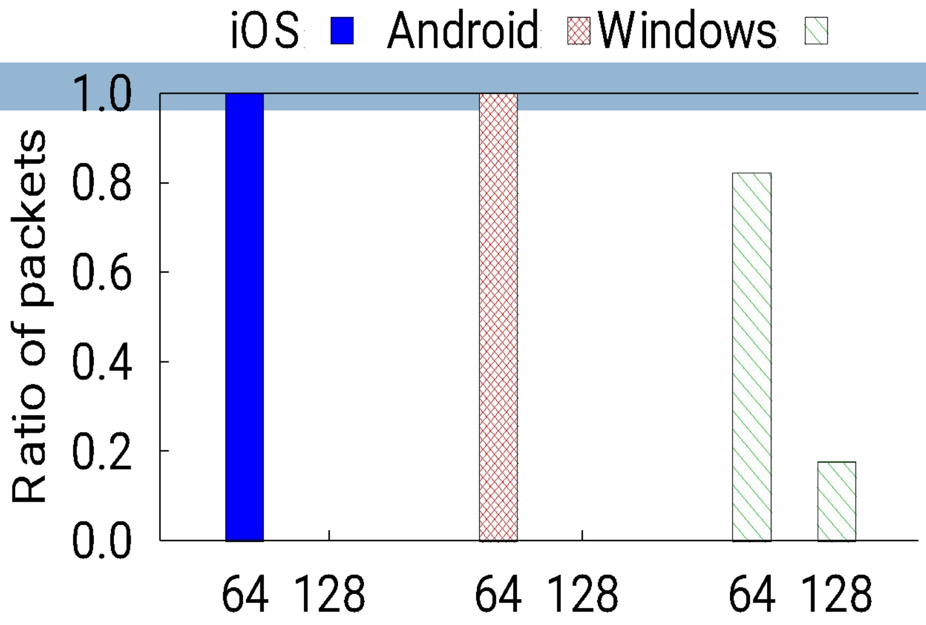
- **TCP Timestamp Option**

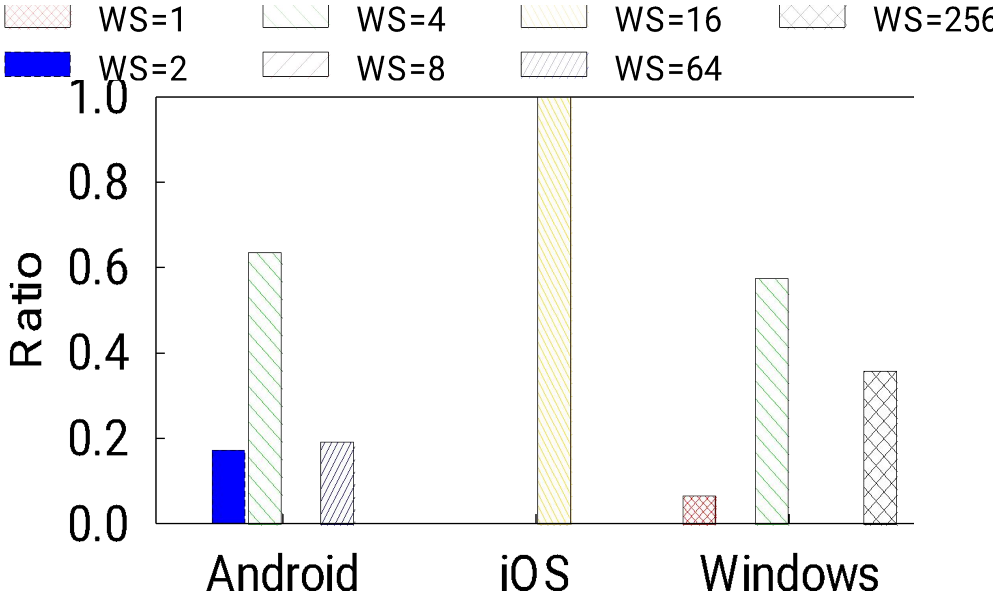
Low ratio of TCP TS option suggests **Windows.**



Features

- □ IP Time-To-Live
- □ TCP Window Size Scale Option
- □ Boot time estimation





Combining Features

- **No single feature** works in all scenarios
- **Naïve Bayes classifier**

Probability of being **OS**$_x$

Probability of finding feature f_i in **OS**$_x$'s traffic

Probability of finding feature f_i in all traffic

Tethering Detection

- Apply the same technique for tethering detection.
- Features which identify mobile devices
  - IP Time-To-Live
  - TCP timestamp monotonicity
  - Clock frequency
  - Boot time estimation
  - Multiple OSes

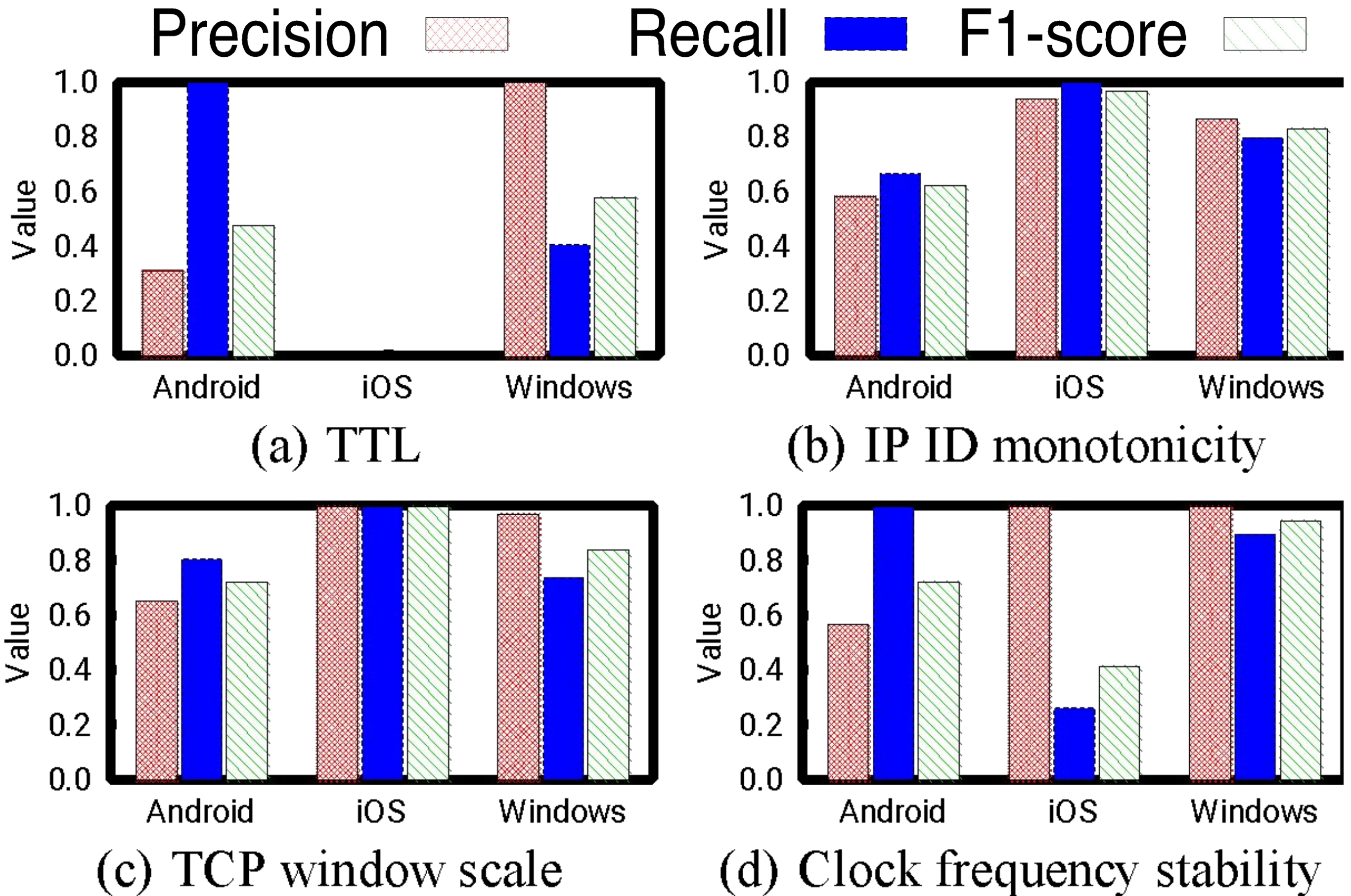
Evaluation – Single Feature

No single feature identifies all OSes accurately.

$$precision = \frac{tp}{tp + fp}$$

$$recall = \frac{tp}{tp + fn}$$

$$F1 = 2\frac{prec \times recall}{prec + recall}$$



(a) TTL

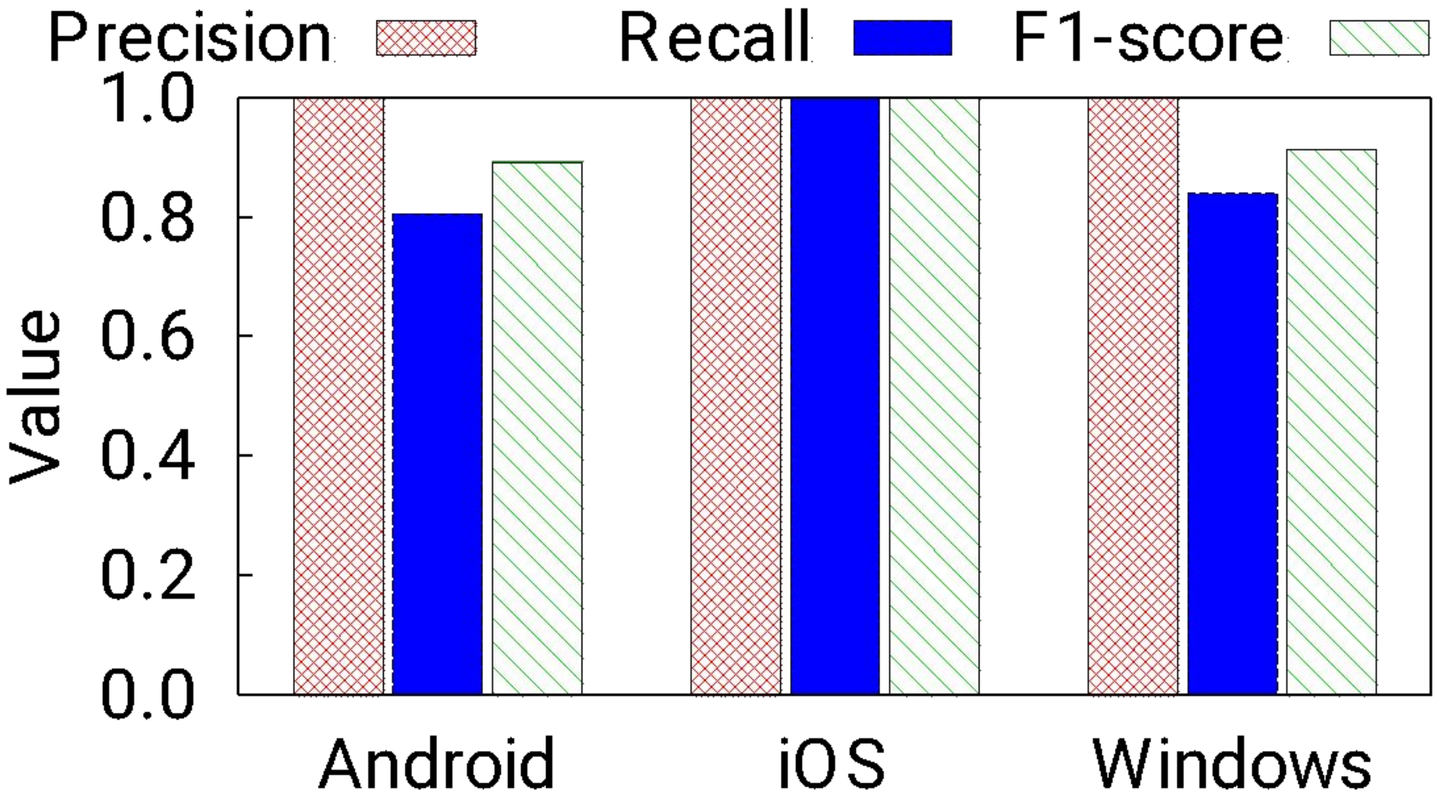
(b) IP ID monotonicity

(c) TCP window scale

(d) Clock frequency stability

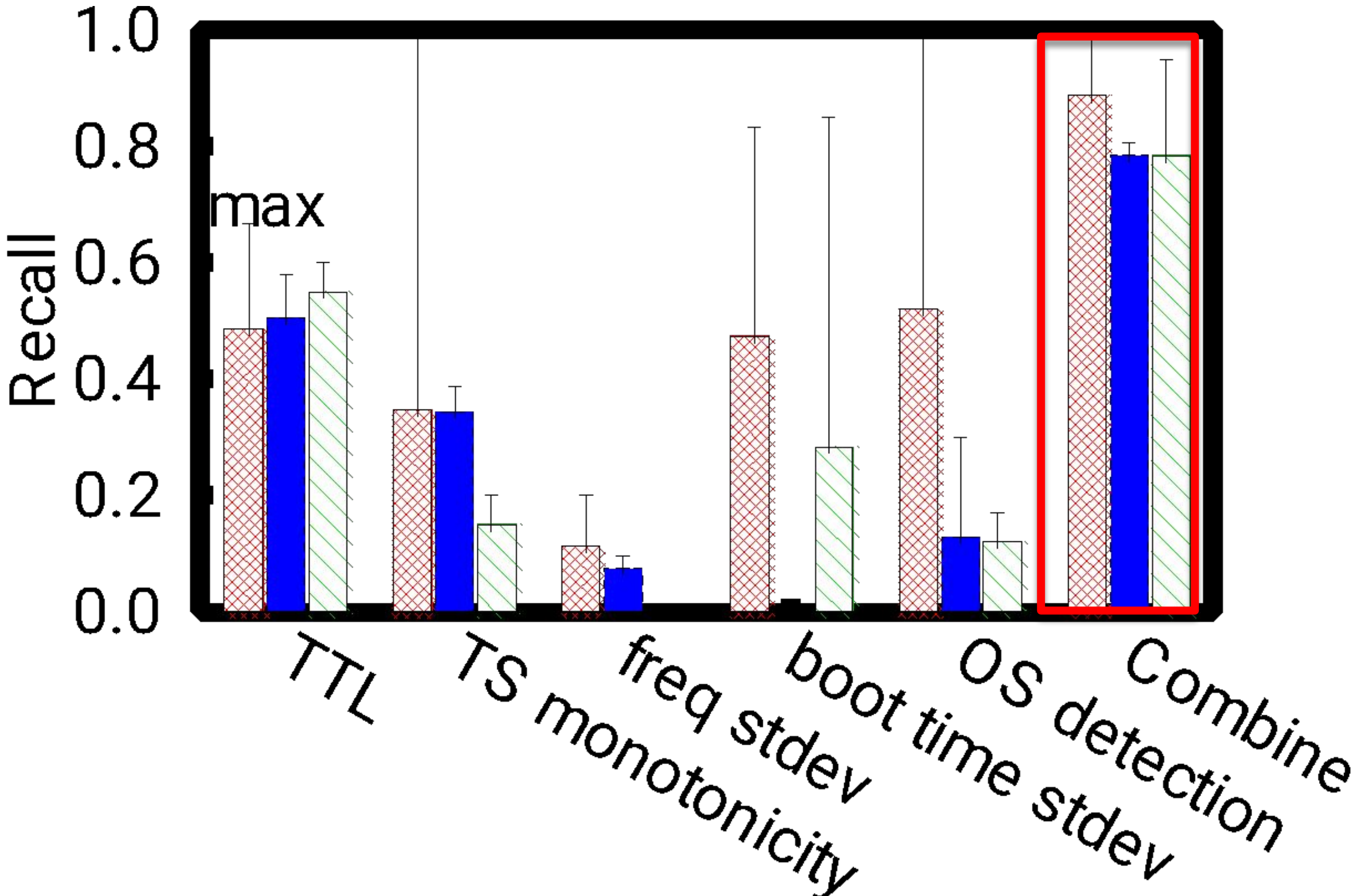
Evaluation – Combing Features

Combining all features yields the best result.



Evaluation – Tethering Detection

Combining all features also yields the best result in **tethering detection.**



Conclusion

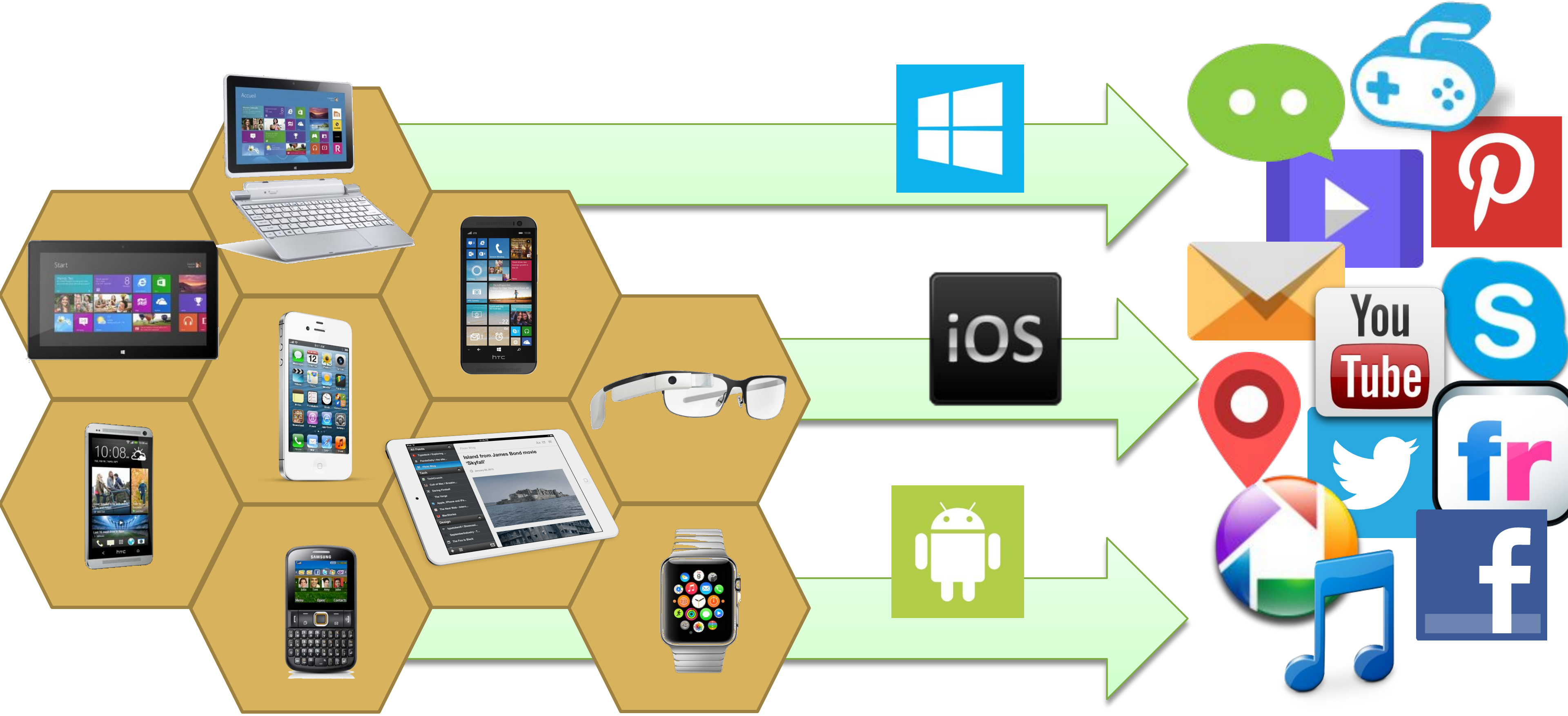
- Contributions
  - Identify new features for mobile OS fingerprinting and tethering detection
  - Develop a probabilistic scheme that combines multiple features
- Evaluate the individual and combined features
  - Combing multiple features yields the best performance
  - OS fingerprinting: 100% precision, 80% recall
  - Tethering detection: 79%-89% recall when targeting 80% precision

Thank You!

yichao@cs.utexas.edu

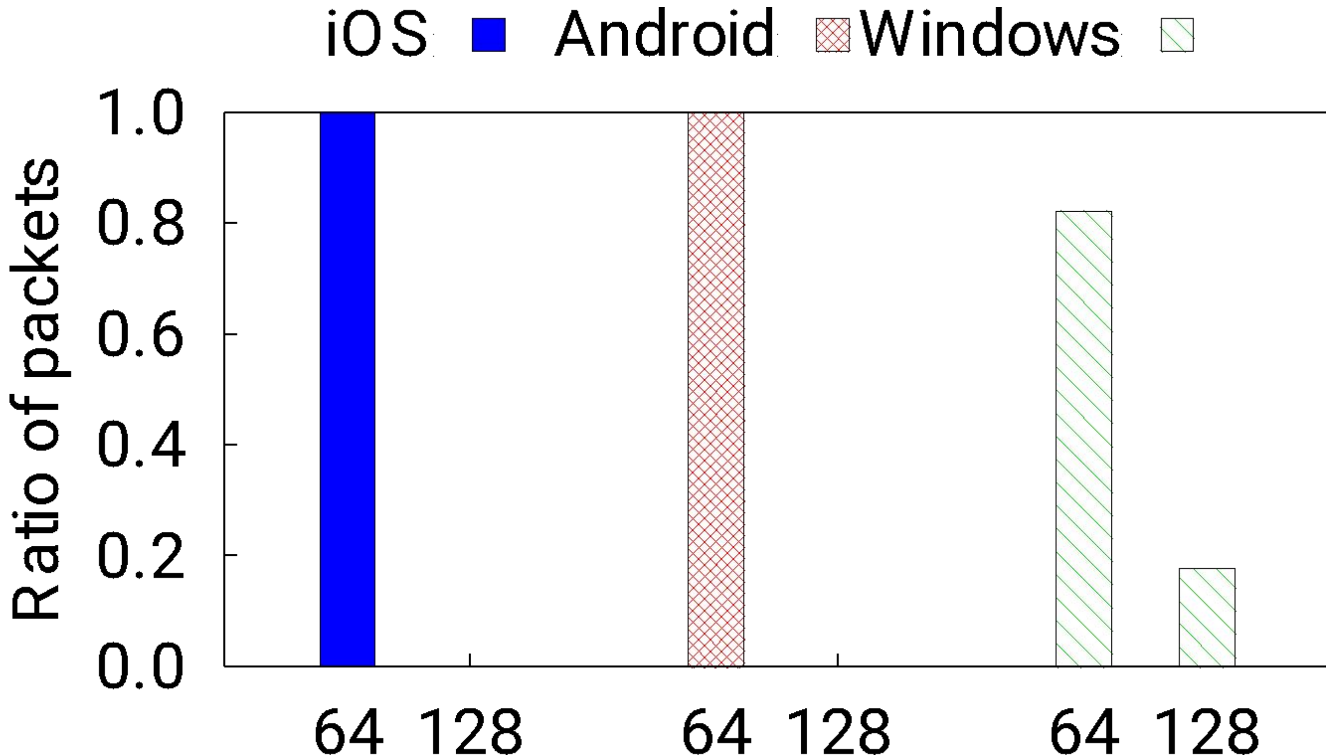
Backup Slides

Mobile OS Fingerprinting



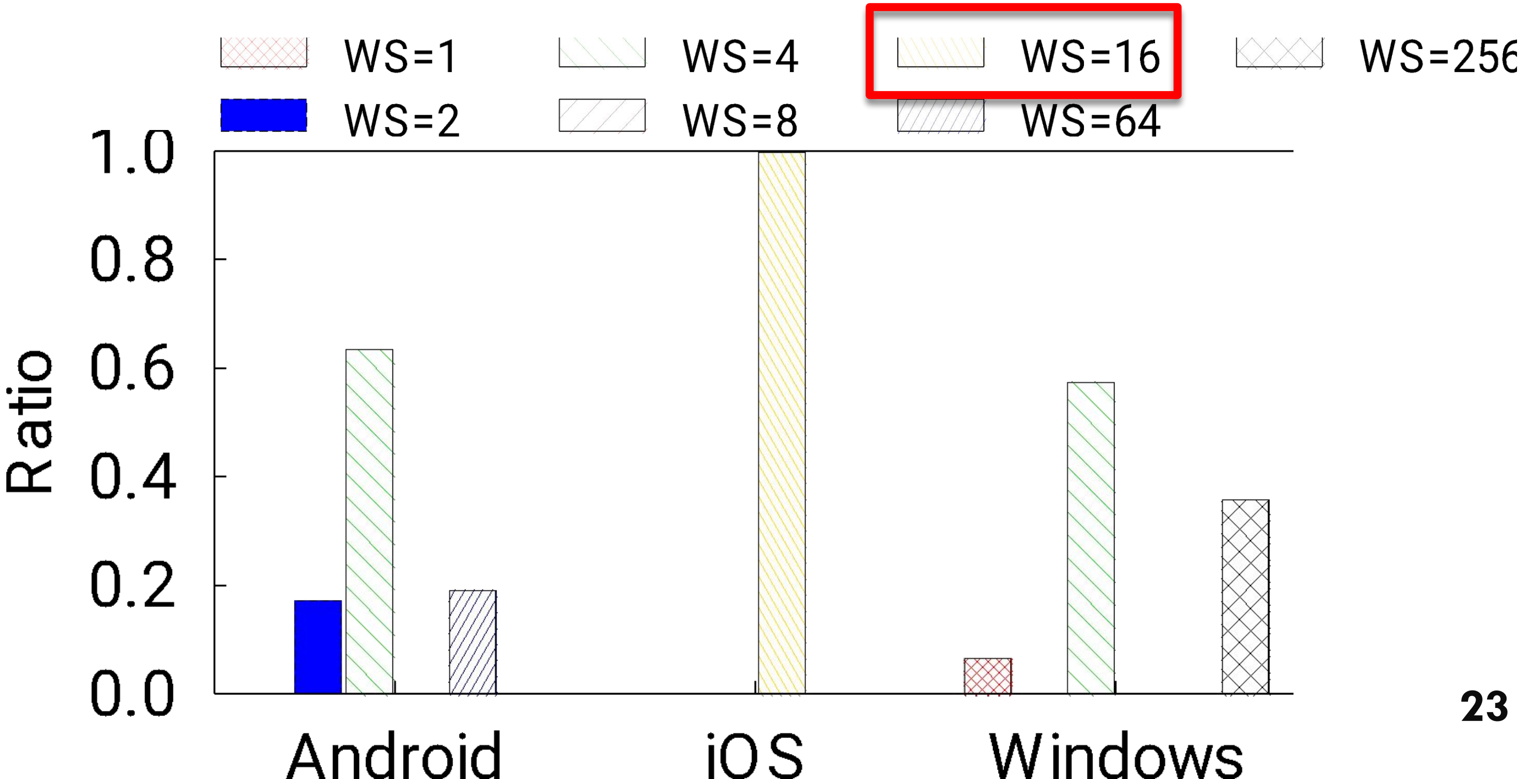
Features

- **IP Time-To-Live (TTL)**
  - **Windows:** *64* or *128*
  - **iOS** and **Android:** *64*



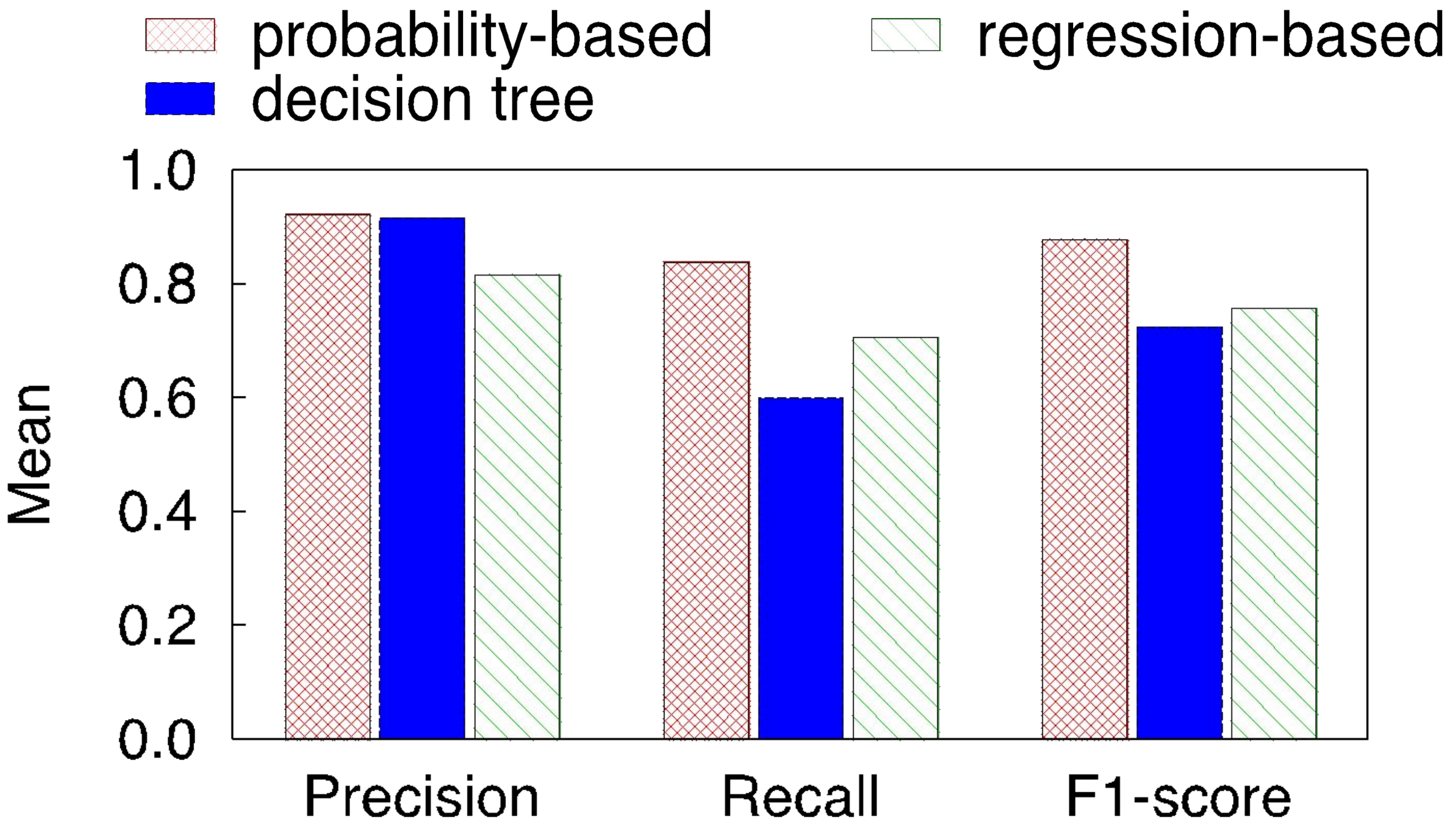
Features

- **TCP Window Size Scale Option**
  - **iOS:** *16*
  - **Windows** and **Android:** *2*, *4*, *64*, or *256*



Evaluation – Comparing Classifiers

Probability based classifier outperforms other classifiers by ***5~21%*** in F1-score measurement.


probability-based
regression-based
decision tree
Mean
Precision
Recall
F1-score