Ubiquitous and Mobile Computing CS 525M: Fast App Launching for Mobile Devices Using Predictive User Context

XIAOCHEN HUANG

Computer Science Dept.

Worcester Polytechnic Institute (WPI)

AUTHORS

Tingxin Yan

David Chu

Deepak Ganesan

Aman Kansal

Jie Liu

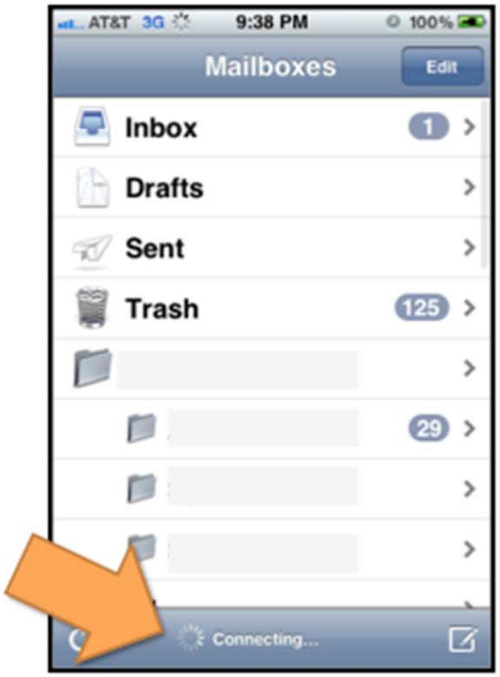
University of Massachusetts Amherst, Microsoft Research
{yan, gamesman}@cs.umass.edu, {davidchu, kansal, jie.liu}@microsoft.com

Mobile Apps … Loading Slowly

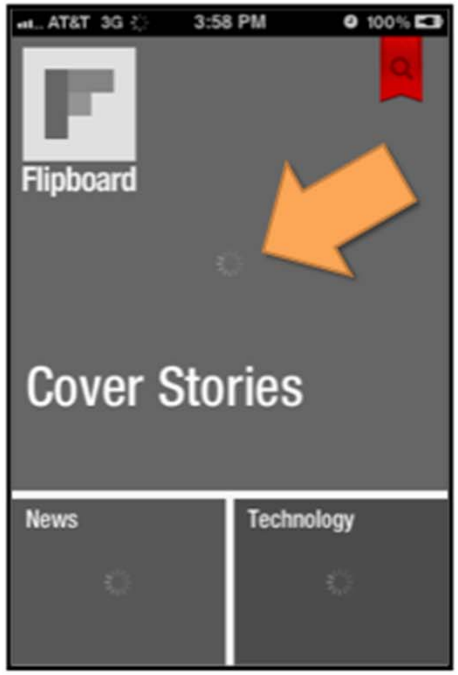
Average loading time >12s

Measured on Sansung Focus Windows Phone

Slow network Content Fetching



Email



News

Social

Loading time > 10s in 3G

Measured on iPhone 3GS

Two approaches …

Cache apps in memory	Push notification
Decrease launch time	Address stale content
• Demand large amounts of memory, overwhelming memory real estate for other apps • Content may become stale by the time the user interacts with the app	• The energy cost of push communication can be prohibitively high

FALCON

- What is Prelaunch?

Prelaunch: Schedule an app to run before the user launches it

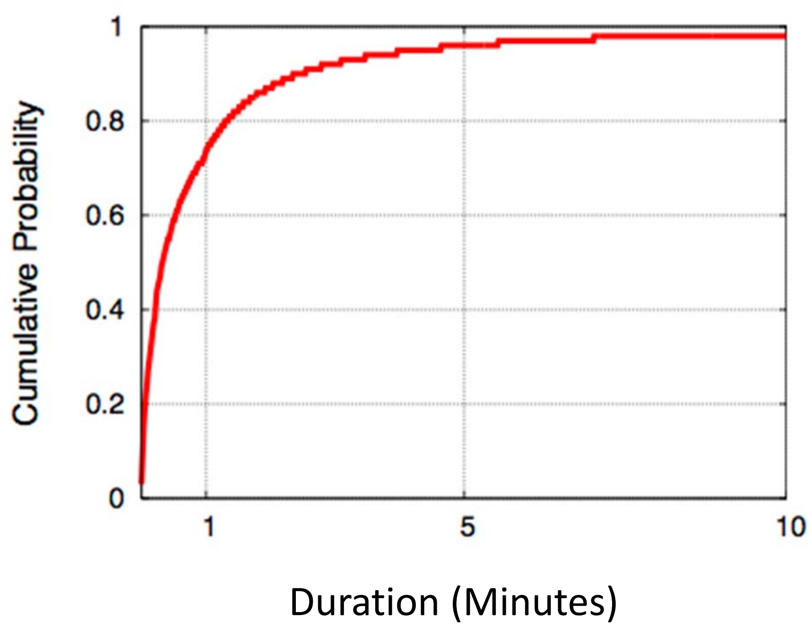
- Challenges and Approaches

inference	optimization	systems
Which app to prelaunch?	At what cost and benefit	Requiring what systems support?
Context clues indicate *predictable* app usage patterns Data insights shed light on informative context	Cost: energy, memory Benefit: latency reduction Problem formulation efficiently provides optimal solution	FALCON' s scheduling and memory management complement kernel's Windows Phone OS mod prototype implementation

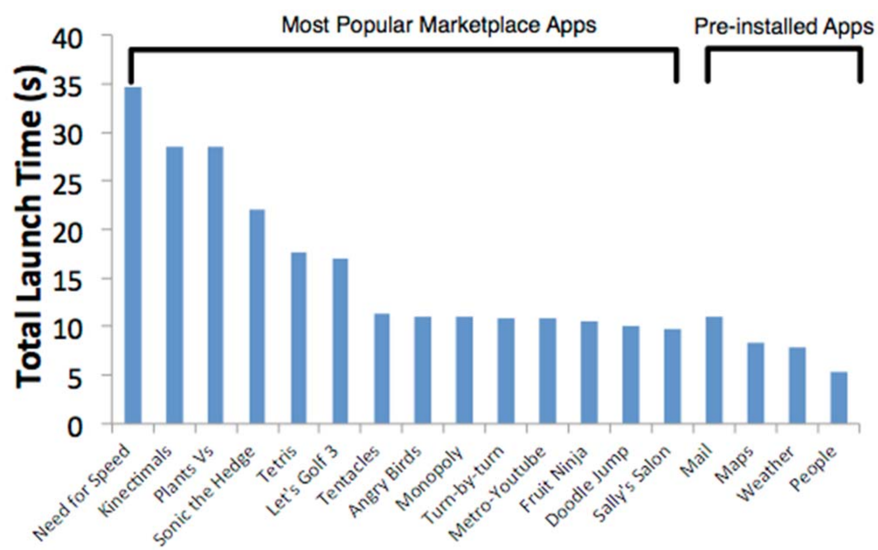
BACKGOURND AND PROBLEM SCOPE

Slow launch times & Brief usage durations

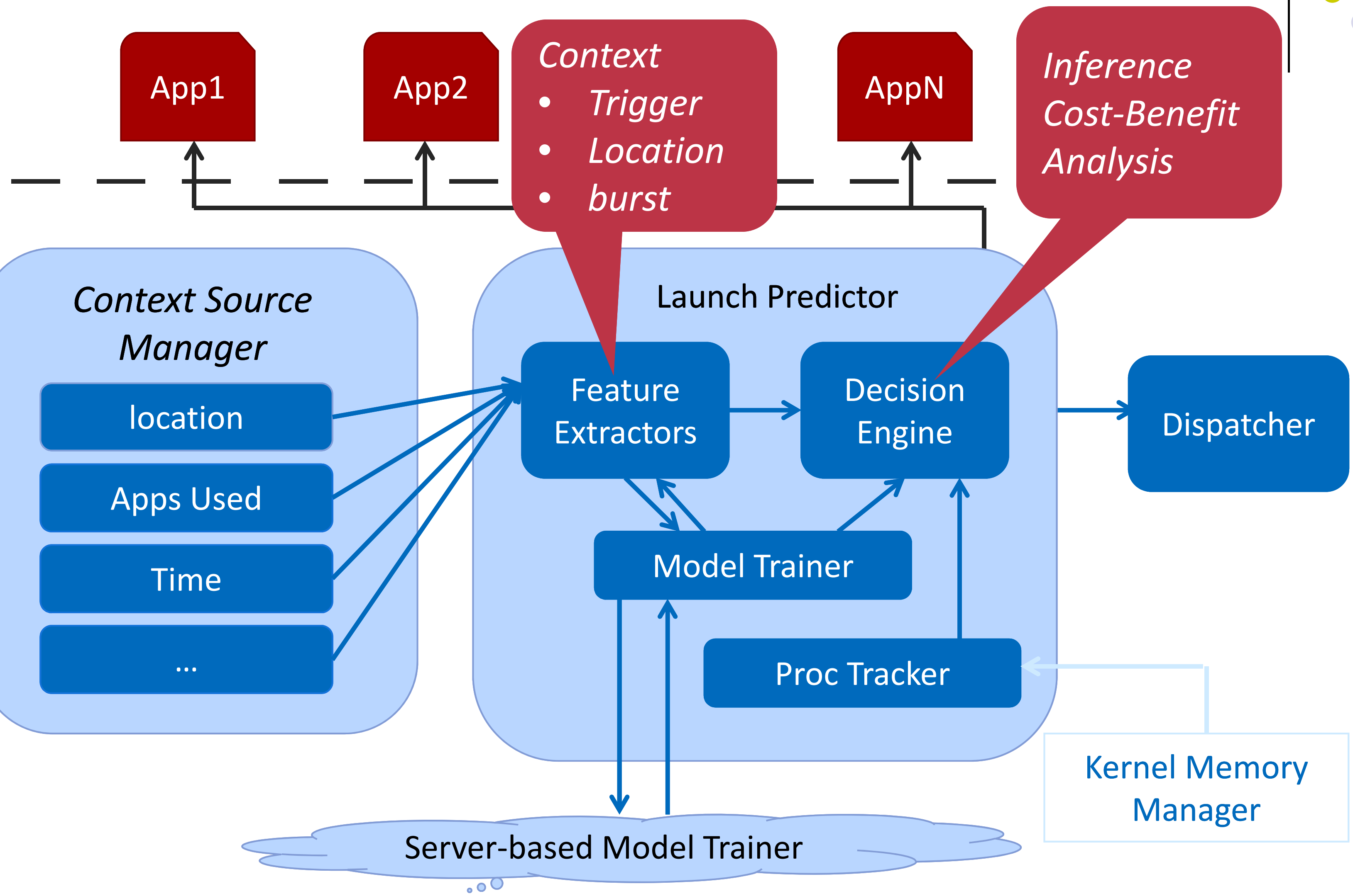
- Examine the thesis that mobile interactions are inherently brief



- Slow app launch is a substantial drag on use experience

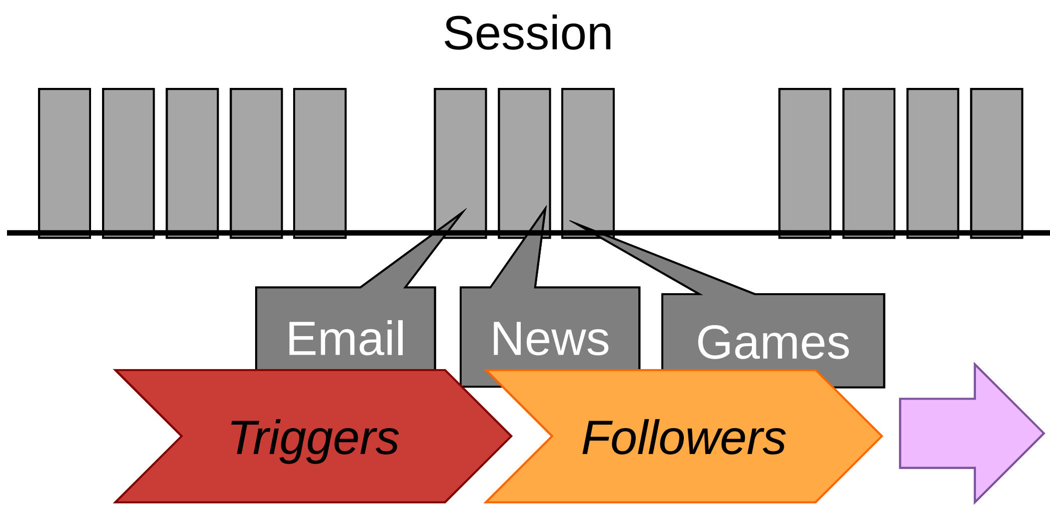


FALCON Architecture & Prototype



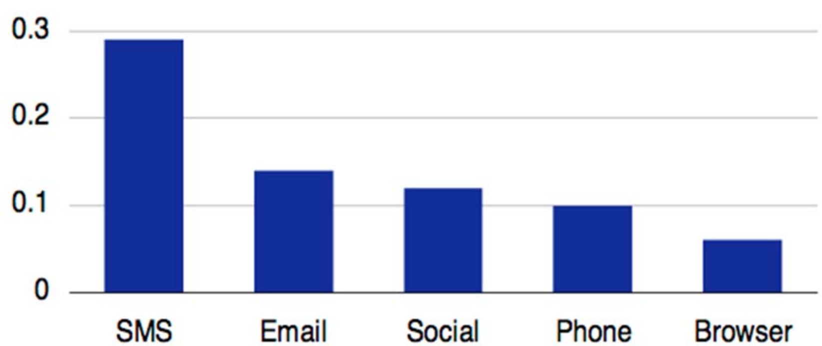
Personalized Features

- Triggers Context



Trigger Context: Given a trigger app, identify most likely follower apps

[interpretation] *Interruptions lead us to habitual routines*



Most popular triggers overall

Personalized Features

- Triggers Context
- ***Problem***

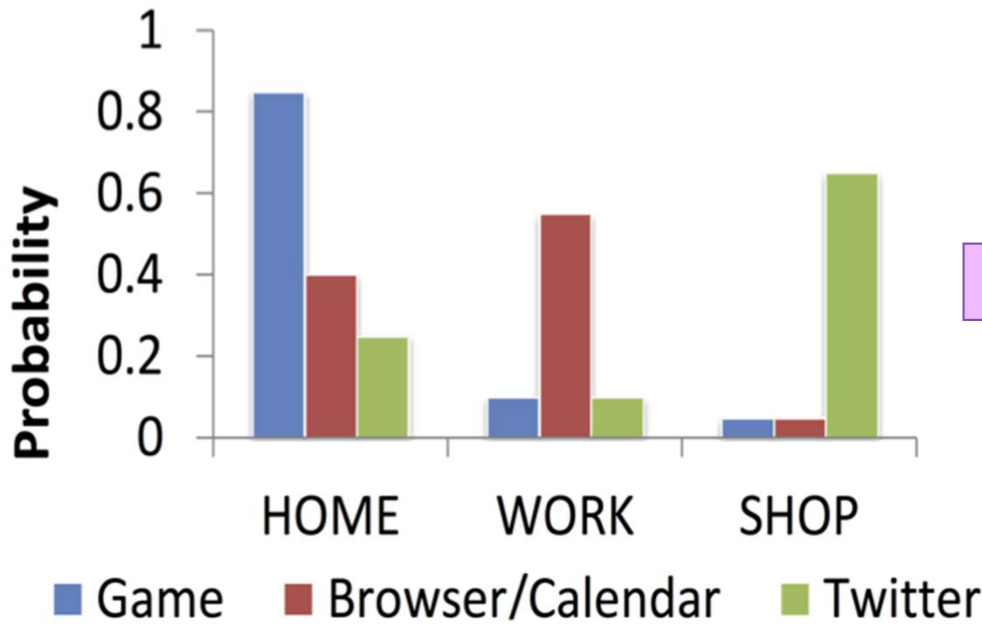
  The best triggers are different for different applications

  Significant variability across users
- ***Dynamic triggers***

  Be calculated on a prelaunch-candidate basis as the set of top-k triggers most likely to lead to the launch candidate as a follower

Personalized Features

- Location Context



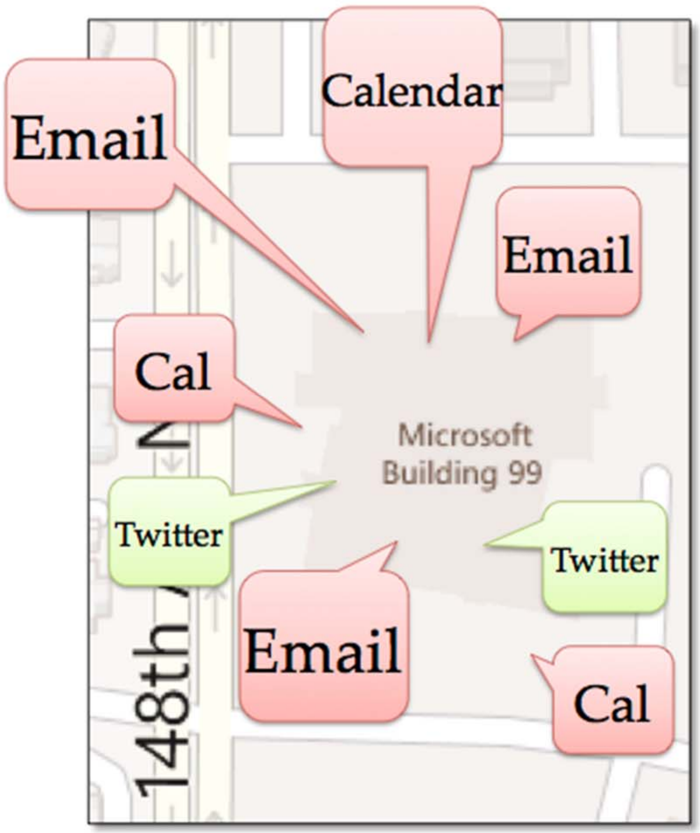
App usage is correlated with location

Location Context: Given a location, identify most likely apps

[interpretation] What we are doing depends on Where we are

Personalized Features

- Location Context



Productivity Apps
@ Work

Social Apps
@ Outdoors

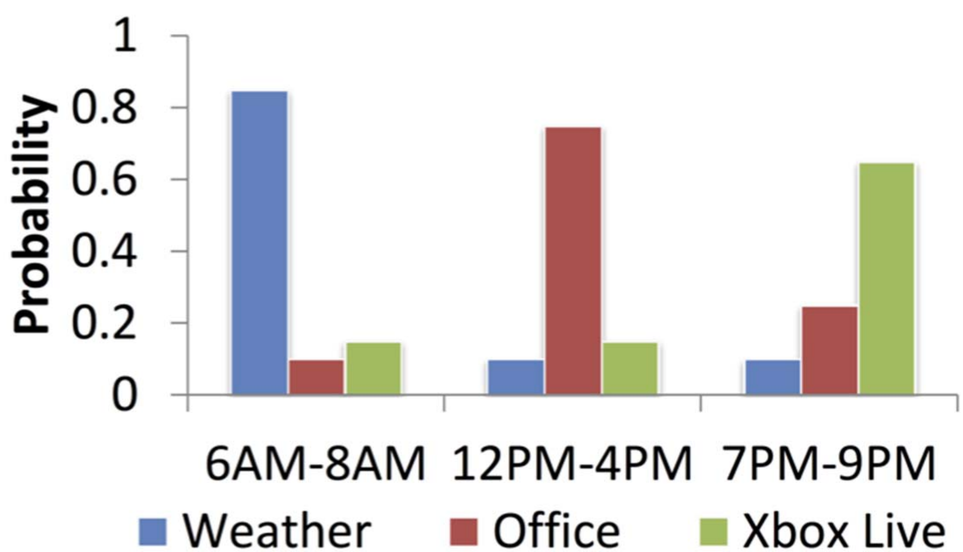
Personalized Features

- Burst Context

Angry birds

Burst Context: detect and predict whether an app is bursting

App usage likelihood changes throughout the day as well



Decision Engine

- Step1: Likelihood Estimation
  - Context

    Trigger, Location, Burst

  - App Likelihood Estimation

    Conditional probability on <Trigger, Location, Burst>

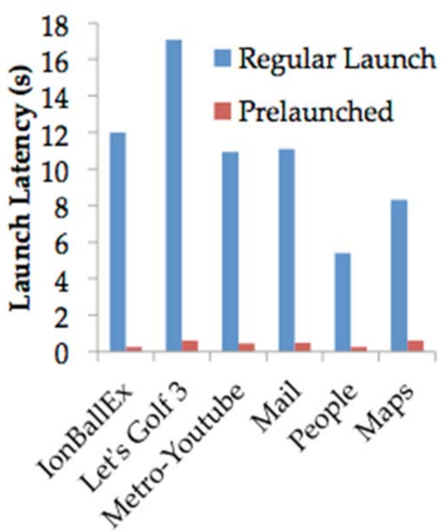
  - Likelihood Estimates used as ranking metric…

Decision Engine

- Step2: Cost-Benefit Analysis

Likelihood Estimates

Potential Benefit: Latency reduction

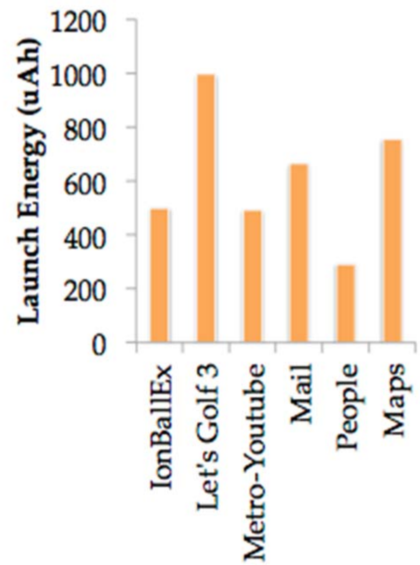


Optimization

- Maximize Latency Reduction
- Subject to energy budget

- Knapsack Problem
- Top-K Greedy approximation

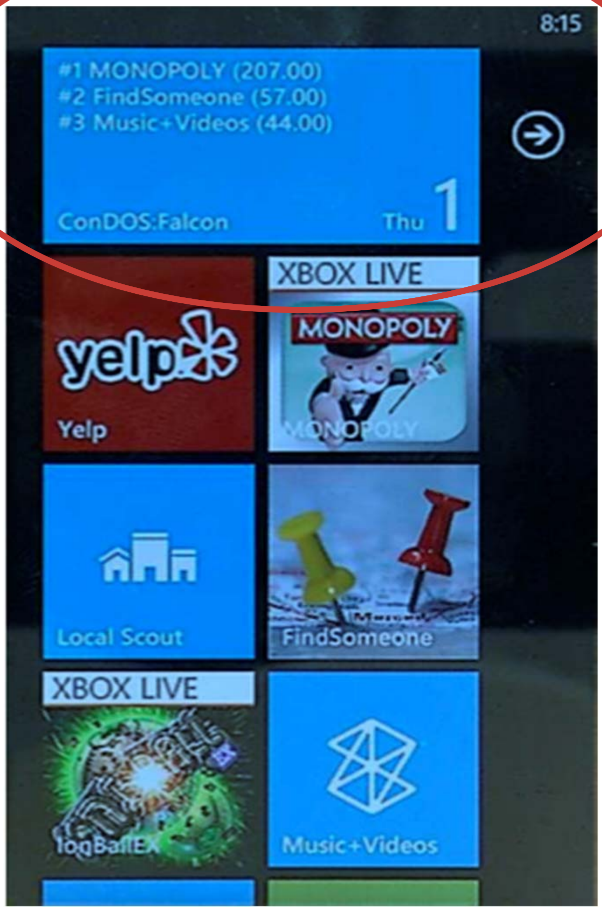
Potential Benefit: Energy wasted



Implementation

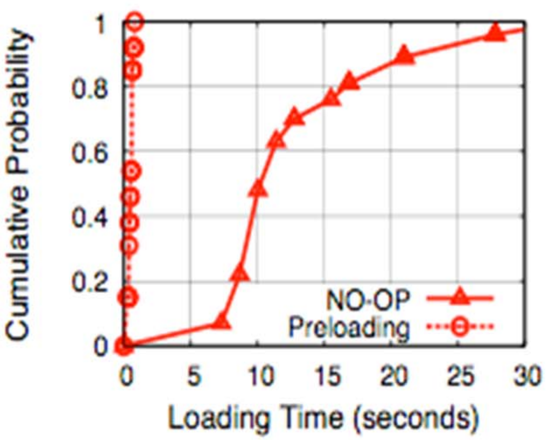
- An OS services
- New application event upcall
- An ambient display widget

FALCON UI

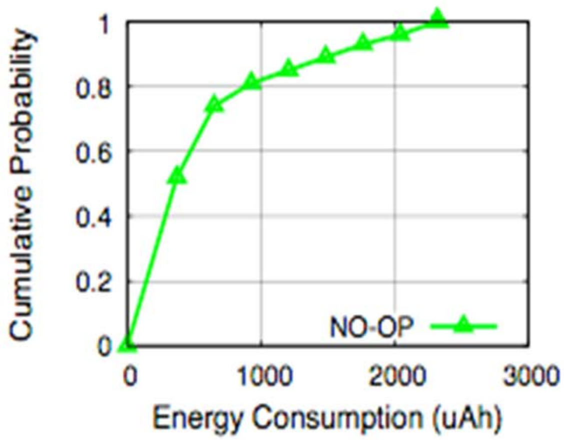


Evaluation: Micro-benchmarks

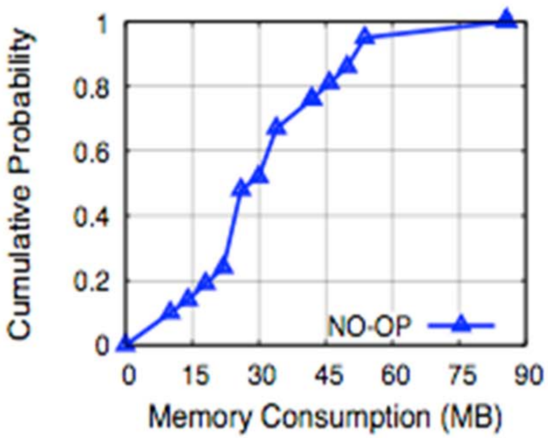
- Three critical parameters
  - the time taken for launching an app
  - the amount of energy consumed during app launch
  - the memory consumed by the app at launch time



(a) CDF of launch time



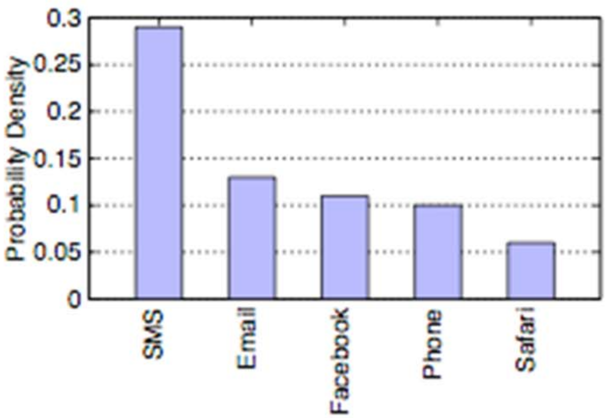
(b) CDF of energy consumption



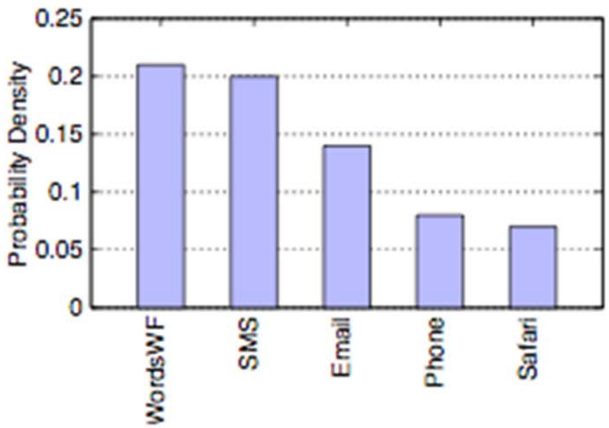
(c) CDF of memory consumption

Evaluation: Benefits of Individual Features

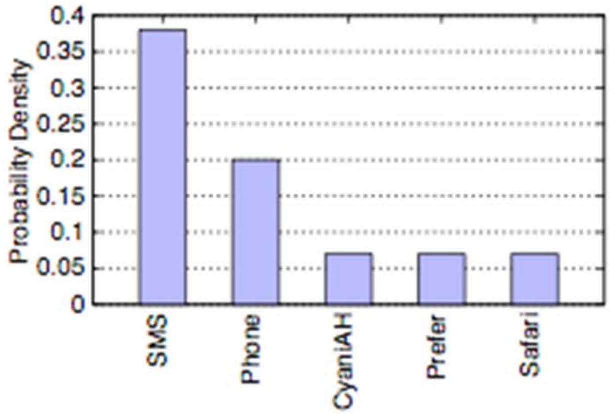
- Session Triggers and Followers
  - Triggers need to be both user- and app-specific
  - Make the case for Dynamic personalized triggers



(a) Top 5 trigger across all users



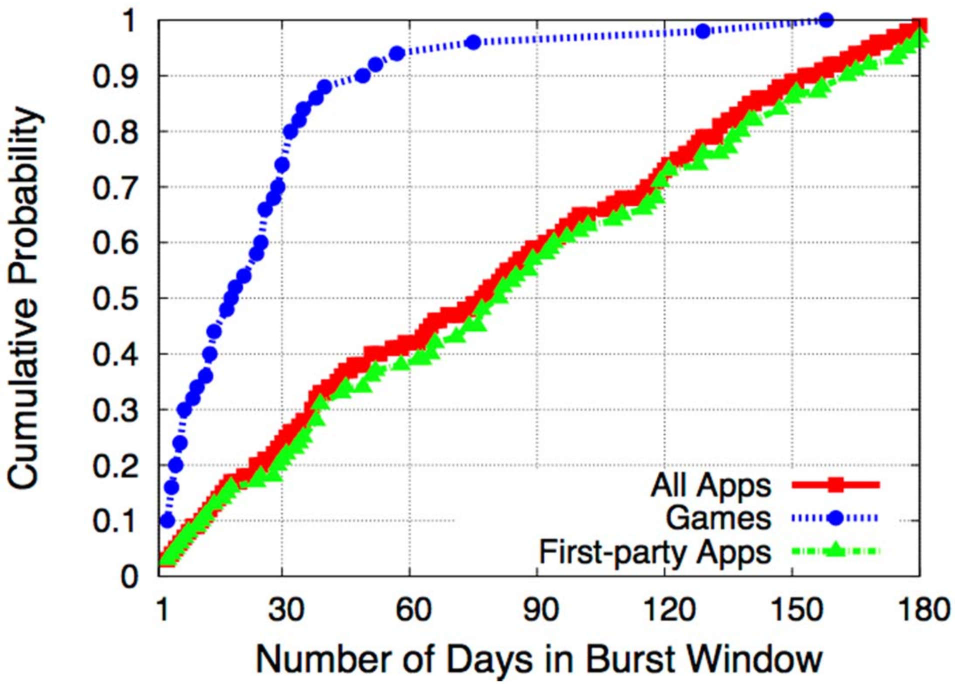
(b) Top 5 triggers for a specific user



(c) Top 5 triggers for “Angry Birds”

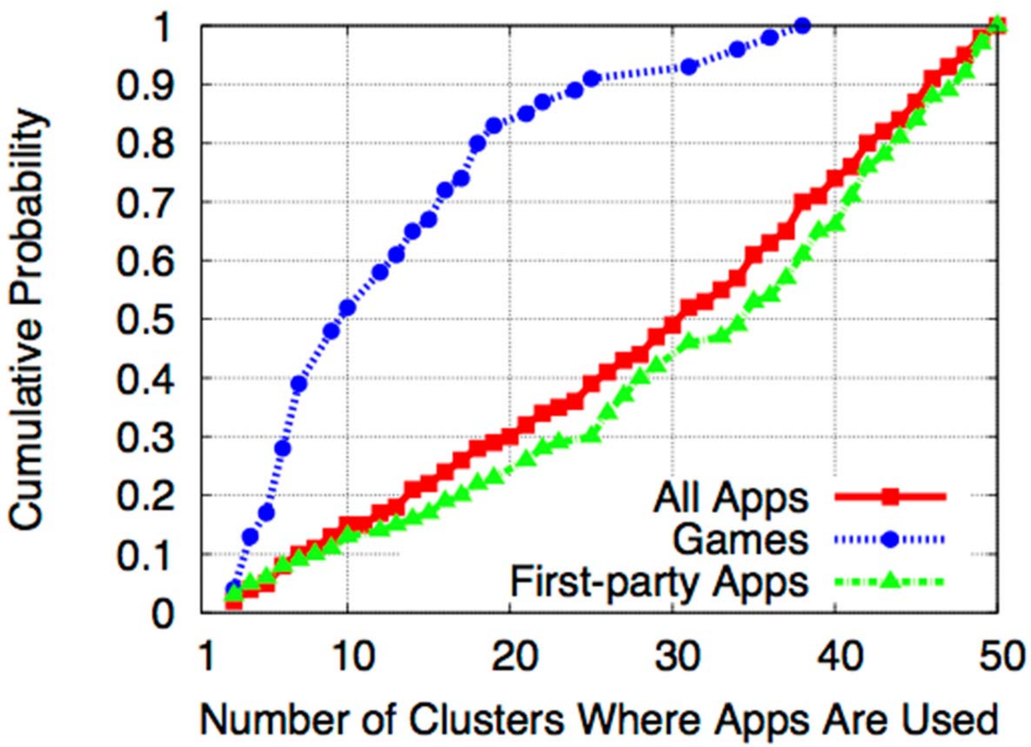
Evaluation: Benefits of Individual Features

- Temporal Bursts
  - An effective feature for improving prelaunch accuracy
  - Three categories: all applications, Games, and First-party apps
  - A discriminating feature that can improve prelaunching performance, especially for games



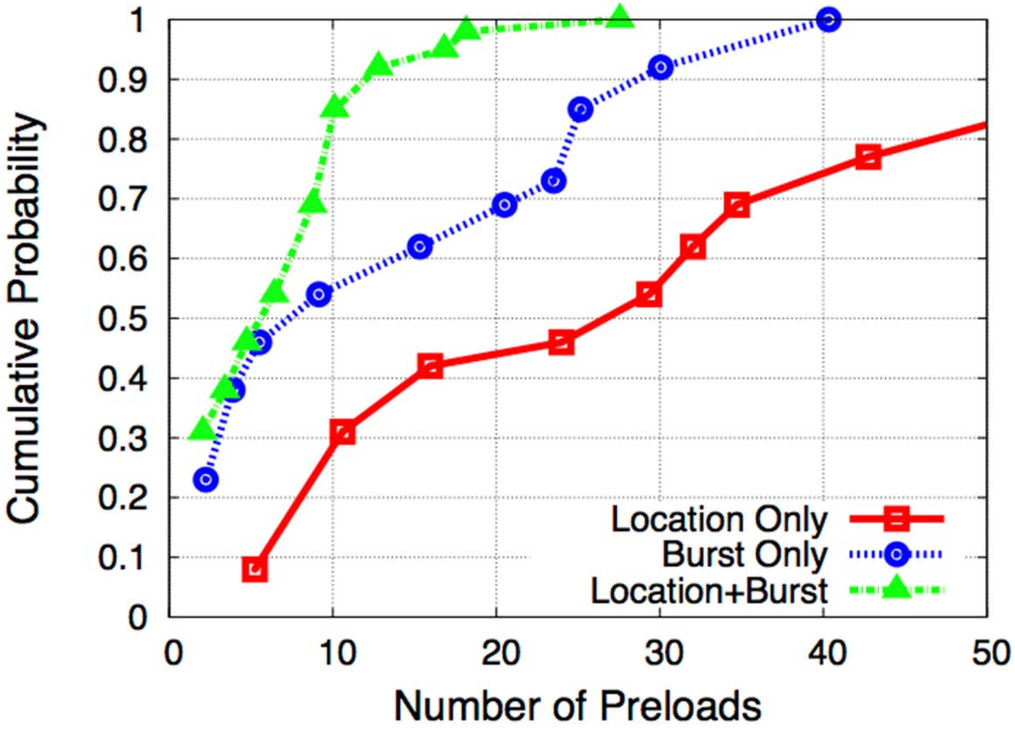
Evaluation: Benefits of Individual Features

- Location Clusters
  - Three categories: all applications, Games, and First-party apps
  - Also a discriminating feature, particularly for games



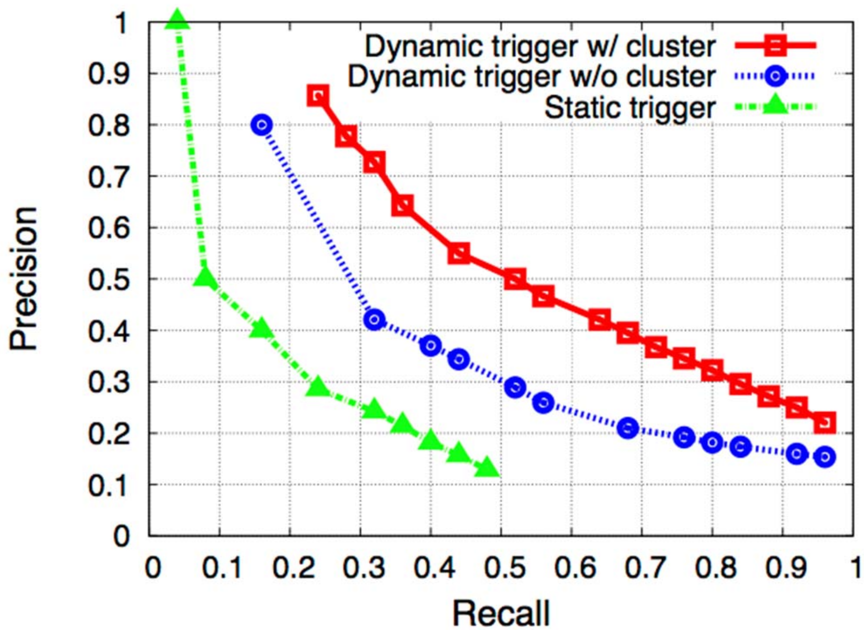
Evaluation: Combining Features

- Benefits of location + temporal features
  - Combining the two features gives better performance

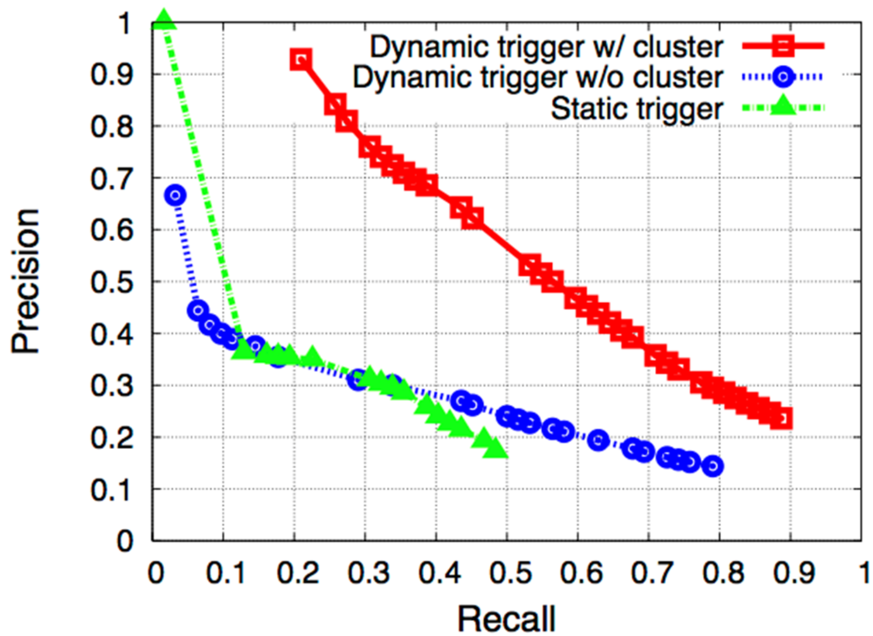


Evaluation: Combining Features

- Benefits of dynamic triggers
  - The performance of dynamic triggers for each location is stable for different users



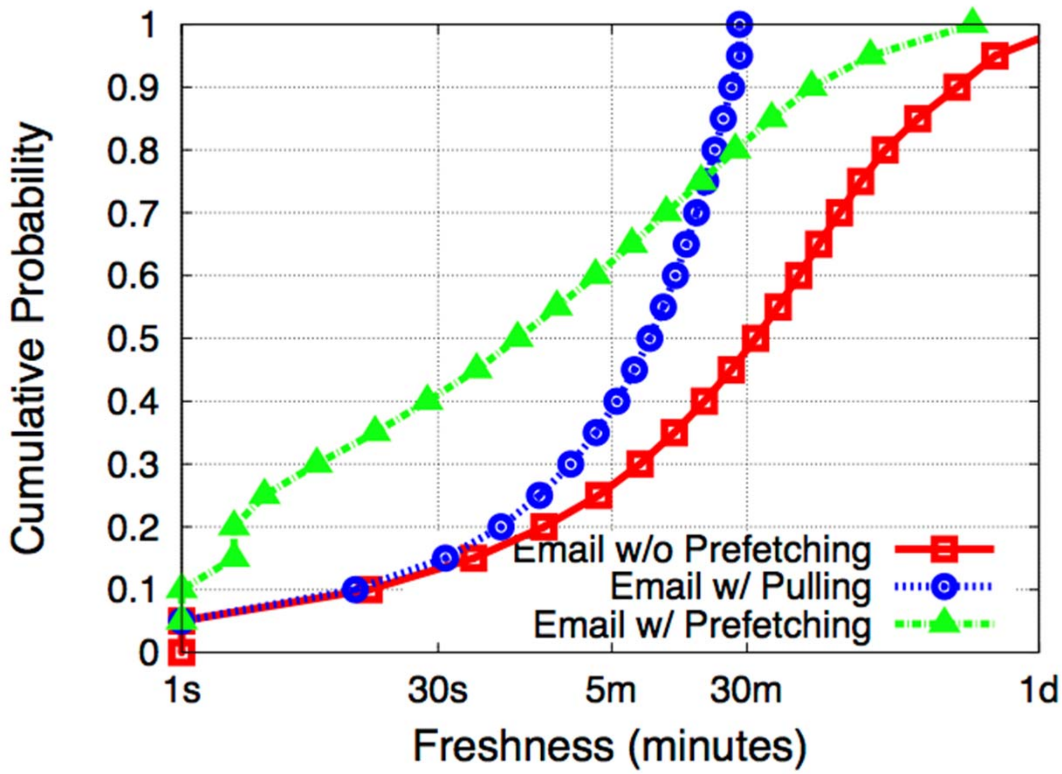
a. User 1



b. User 2

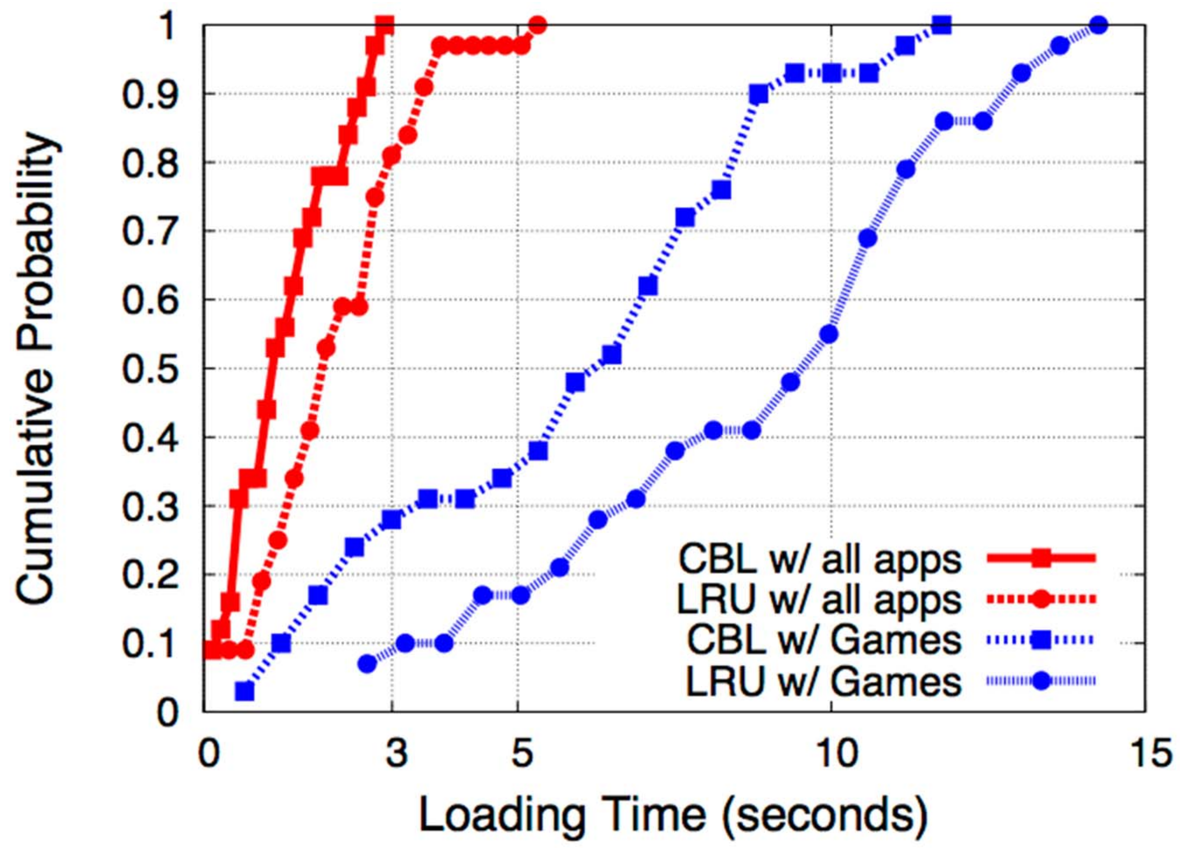
Evaluation: Evaluation of cost-benefit learner

- Performance of Prefetching
  - Look at the benefit for apps that require fetching new content during the launch process



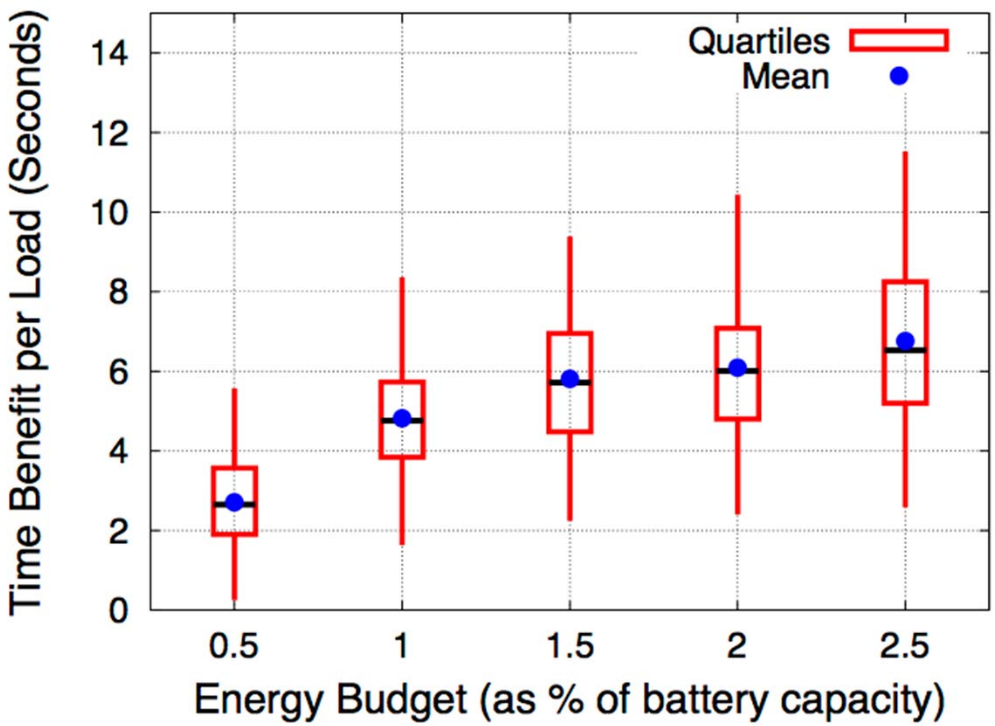
Evaluation: Evaluation of cost-benefit learner

- Performance of Preloading
  - Look at how app loading time can be improved by CBL learner, compared with LRU caching



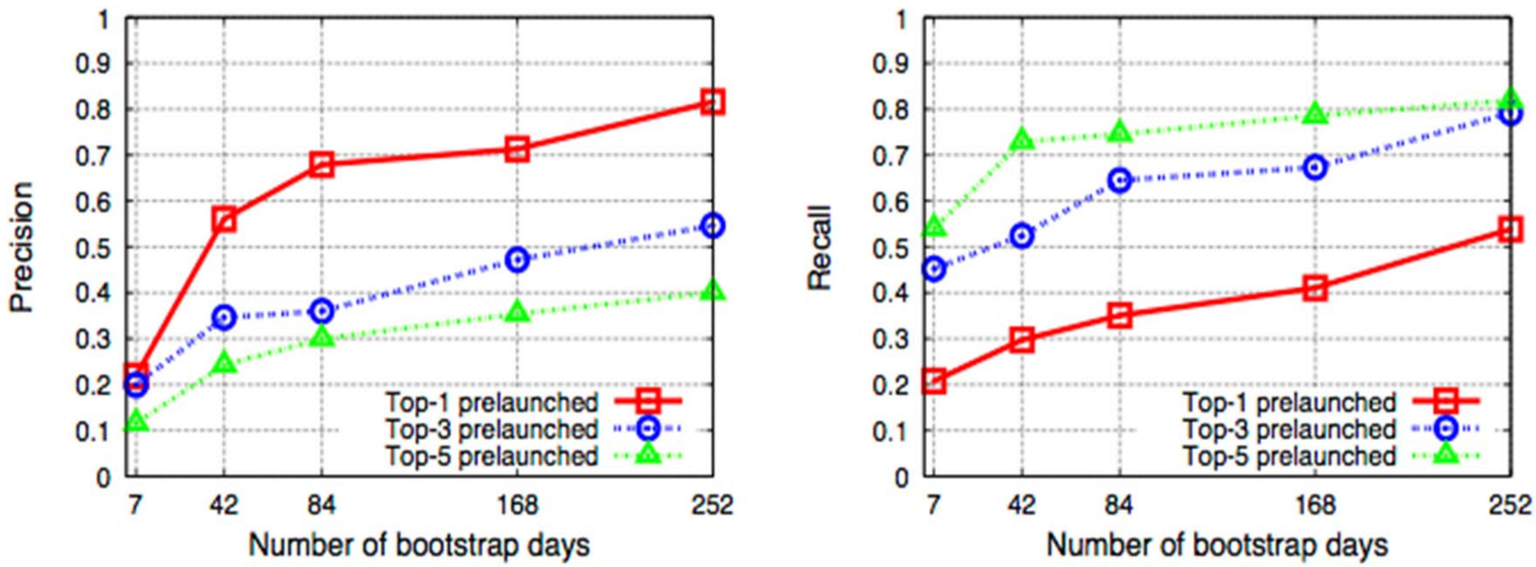
Evaluation: Evaluation of cost-benefit learner

- Overall benefits
  - X-axis represents the energy budget provided to CBL
  - Y-axis represents the benefit of loading time which includes both app loading time and content fetching time



Evaluation: Bootstrapping FALCON

- Look at how fast the cost-benefit learner can learn from history of app usage to make accurate prelaunch decisions.
- The performance of FALCON grows as the training data size increases



(a) Precision (b) Recall

Precision and recall of bootstrapping

Evaluation: System Overhead

- The resource consumption profile as follows

Binary Size	129 KB
Memory (stable state)	1840 KB
Processor utilization (stable state)	<1%
Processor utilization per prediction	<3%
Energy cost per prediction	$< 3\ \mu$Ah

Table FALCON Overhead Profile

- Do not account for the periodic geolocation sampling costs
- Online feature extraction

  (a) Perform location clustering cost much on a phone, therefore need to be done in the cloud

  (b) Implement a light-weight online burst window detection algorithm

  (c) Online burst detection as a small performance loss

DISCUSSION AND CONCLUSION

- **Contribution**
  - From extensive data analysis, design spatial and temporal features that are highly indicative of mobile app access patterns
  - Design a cost-benefit learning algorithm
  - Prototype FALCON on a Windows Phone
- **Future work**
  - Eliminating reliance on external servers or cloud services for model training in FALCON
  - How users' expectations will change as the OS predictively prelaunches apps on their behalf

Ubiquitous and Mobile Computing CS 525M: Fast App Launching for Mobile Devices Using Predictive User Context

THE END

THANK YOU!

XIAOCHEN HUANG

Computer Science Dept.

Worcester Polytechnic Institute (WPI)