Ubiquitous and Mobile Computing
CS 528: Attelia: Reducing User's Cognitive Load due to Interruptive Notifications on Smart Phones

Kuang Xiong

Tengyang Jia

Computer Science Dept.

Worcester Polytechnic Institute (WPI)

Outline

- Introduction
- Interruption overload
- Adaptive notification scheduling on smart phones
- Design of Attelia
- Attelia System Architecture
- Controlled User Study
- In-The-Wild User Study
- Further Challenges
- Related Work
- Conclusion

Introduction

- Given this background of information overload, the limited resource of human attention is the new bottleneck in computing
- Focus on interruption overload
- Contribution
  - the design and implementation of breakpoint detection system
  - present the results from both a controlled and an “in-the-wild” field user study

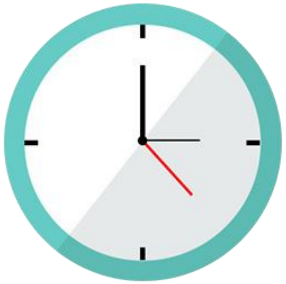
Interruption Overload

Importance

Interruption overload caused by large numbers of ill-timed notifications is one piece of the larger problem of information overload, and is increasing in frequency.

Solutions

scheduling



Defer the notification

mitigation

changing the modality

Adaptive Notification Scheduling on Smart Phones

Recent Trends of Notifications

- Increasing diversity in types and sources of notifications
- Multiple mobile devices as targets
- Wider range of urgency level
- Increasing length of interruptive periods

Principles for Attention Status Sensing

- Feasibility for mobile devices
- Real-time sensing
- Applicability to diverse types of notification sources
- All-day-long use

Design of Attelia

Breakpoint detection will satisfy these three features:

1. “Breakpoint” as a Temporal Target for Interruption

Attempts to sense more coarse-grained, but easier to sense signals

Design of Attelia

Breakpoint detection will satisfy these three features:

2. Application Usage as a Sensor

Focus on a user's application usage and use that information to detect a user's breakpoints.

Reason: simplicity of implementation andreducing the reliance on a sensor that may not exist on all target mobile devices

TABLE I. APPROACHES OF KNOWLEDGE COLLECTION FOR BREAKPOINT DETECTION

Approaches on Knowledge Source of Breakpoint	Examples of Data Types
Application-specific breakpoint knowledge	explicit breakpoint declaration inside application, explicit future breakpoint forecast inside application
Runtime status/event of systems and applications	stack trace, number of threads, thread names, memory consumption Android API invocation, system call invocation, rendered screen image, Low-level GUI events, switches between applications

TABLE II. TIMINGS OF KNOWLEDGE INPUT AND DATA COLLECTION

Approaches on Knowledge Source of Breakpoint	Knowledge on Breakpoints: When? By Who? and How?		Data Collection at Application Run-Time
	Application Development Phase	System Training Phase	
Application-specific breakpoint knowledge	Embedding additional API calls to provide explicit breakpoint knowledge (by application developer)	None	From API calls embedded inside running applications
Runtime status/event of systems and applications	None	Ground truth annotation of collected status/event information (by application users)	From the middleware and operating system

Design of Attelia

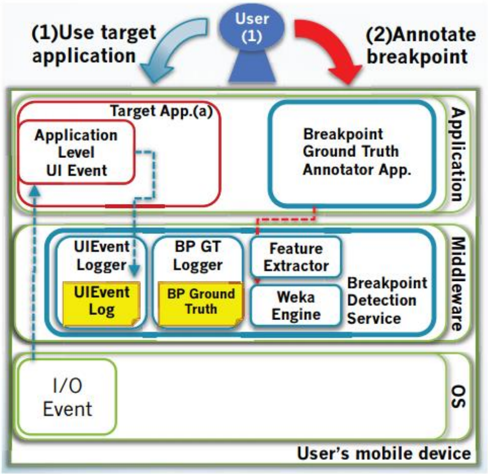
Breakpoint detection will satisfy these three features:

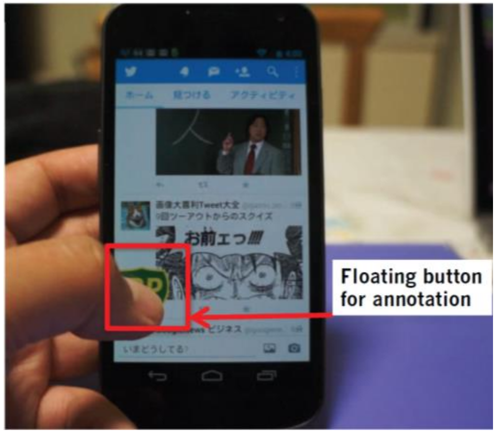
3. Real-Time Detection with Machine Learning techniques

J48 classifier

For each time frame Tf, a feature vector V is extracted from the sensed data, and a trained classifier identifies the time frame as a user breakpoint or not.

Attelia System Architecture





Execution Modes

- Ground truth collection

  users manually provide ground truth about breakpoints during application usage

- Off-line model training

  feature extraction and classifier training is executed off-line.

- Real-time mobile breakpoint detection

  Sensing, feature extraction, and classification with a previously trained model.

Attelia System Architecture

Sensing Data and Features

- Using Android Accessibility Framework.
- Collecting UI events and data about the UI components the user is interacting with.

TABLE III. UIEvent Collected in Attelia

Event Types	Events
View	`View clicked`, `View long clicked`, `View selected`, `View focused`, `View text changed`, `View selection changed`, `View text traversed at movement granularity`, `View scrolled`
Transition	`Window state changed`, `Window content changed`
Notification	`Notification state changed`

TABLE IV. Features used in Attelia

Feature Types	Features
Rate of occurrence of each UI Event type inside the frame	snipped (one for each event type presented in Table III)
Statistics on the status of the event source UI component	*rate*(`isEnabled`), *rate*(`isChecked`), *rate*(`isPassword`)
Statistics on the events' timings in the frame	*min_timegap*, *mean_timegap*, *max_timegap*, *stdev_timegap*
Statistics on the location of the event source UI components	*min., mean., max., stdev., the value of the smallest rectangle, the value of the biggest rectangle of* `X-left`, `X-right`, `X-width`, `Y-top`, `Y-bottom`, `Y-height`

Attelia System Architecture

Frame Length

- choice of time frame length Tf will affect our ability to perform breakpoint detection.

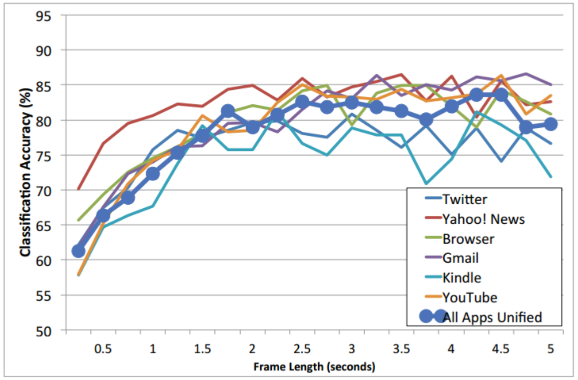


Fig. 3. Classification Accuracy and Frame Length

Around 2 to 2.5 seconds, the accuracy begins to stabilize. At the 2.5-second setting, accuracy was 82.6%, precision was 82.7% and recall was 82.3%

Attelia System Architecture

Power Saving

TABLE V. COMPARISONS OF POWER CONSUMPTION OVERHEAD

Sensor Type	Frequency (Hz)	Overhead (mW)
UI Events	10	51.70
Accelerometer	120	102.90
	60	48.76
	15	12.08
Gyroscope	100	158.88
	50	129.24
	15	74.04

Portable Implementation

- Attelia is implemented as a “Service” inside the Android platform.

Controlled User Study

- Participants
  - 37 users
  - ages 19 to 54
  - not paid and not told the purpose
- Experiment Setup
  - Disabled the android notification
  - 6 representative application
  - a trained J48 classifier
  - 4 strategies: disabled, random timing, breakpoint timing, and non-breakpoint timing

Controlled User Study

- Interruptive Task
  - a full screen pop-up
  - contain a question for user to answer

- Experiment Procedure
  - 2 parts: send 5 emails and use other 5 applications for 5 minutes each
  - use Latin Square to eliminate ordering effects
  - see one strategy twice randomly

Controlled User Study

- Measurements
  - NASA-Task Load Index questionaire
  - http://humansystems.arc.nasa.gov/groups/tlx/
- Results
  - Clustering users by NASA-TLX weighted workload scores

Cluster name	Users	Mean WWL Stdev.
“sensitive”	19	23.11
“insensitive”	18	9.92

Controlled User Study

- Results for sensitive user

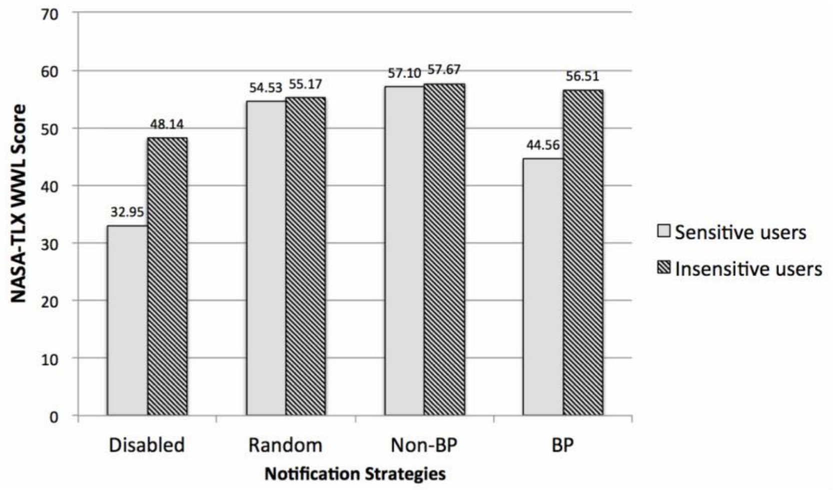


Fig. 5. NASA-TLX WWL Scores for Each Cluster

- Baseline: Disabled
- Increase of workload:
  - Breakpoint(35%) < Random(66%) < Non Breakpoint(73%)
- Breakpoint have 46% reduction in workload compared to Random

Controlled User Study

- Results for Insensitive cluster
  - Only significant difference between “Disabled” and other strategies

In-The-Wild User Study

- Participants
  - 30 users (20 male and 10 female)
  - ages 18 to 29
  - pay $60 for participation

- Experiment Setup
  - same J48 classifier
  - 3 strategies: no notification, random timing, and breakpoint timing
  - randomly choose strategy each day
  - interval: 15 ~ 30 minutes
  - maximum interruptive task: 12
  - Test time: 8AM ~9PM

In-The-Wild User Study

- Interruptive Task
  - two full screen pop-up
  - ask if it is a breakpoint
  - contain a question for user to answer

- Experiment Procedure
  - 16 days experiment
  - evaluate through NASA-TLX survey everyday
  - post-experiment survey

In-The-Wild User Study

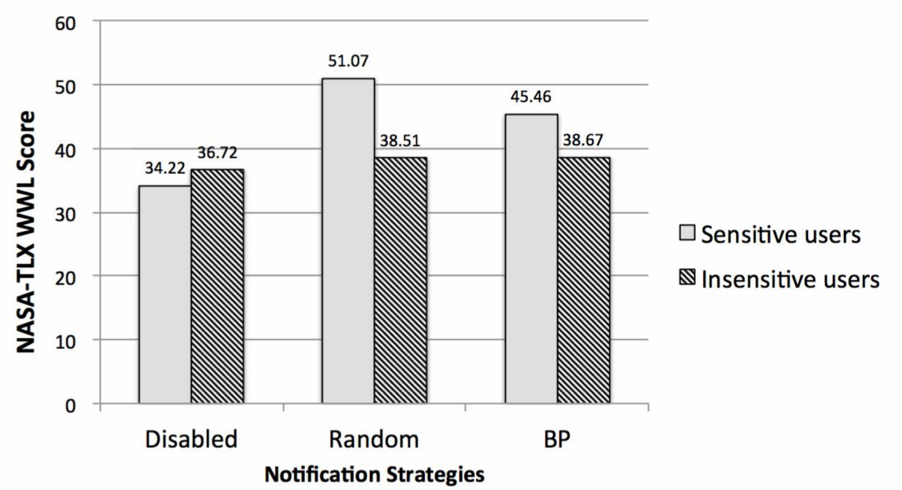
- Measurements
  - NASA-TLX everyday
  - response time, time to answer question, quiz answer

- Results
  - Clustered by NASA-TLX weighted workload scores
  - 27 valid users

Cluster name	Users	Mean WWL Stdev.
"sensitive"	13	21.38
"insensitive"	14	8.19

In-The-Wild User Study

- Results for sensitive user



- Baseline: Disabled
- Increase of workload:
  - Breakpoint(33%) < Random(49%)
- Response time for first pop-up
  - Breakpoint(2.77s) < Random(3.18s)

Further challenges

- Insensitive users study
- Real android notification study
- Support for multiple mobile devices

Related Work

- Desktop environments interruptibility inference
- Interruptibility research based on external on-body sensor
- Interruptibility research based on smartphone sensor data

Conclusion

- Proposed a novel middleware to identifies when to deliver notification
- Detect breakpoint in real-time
- Without any additional devices or any modification to applications
- Evaluated the design by controlled user study and “In the wild” user study

References

- *http://humansystems.arc.nasa.gov/groups/tlx/*
- *https://en.wikipedia.org/wiki/Friedman_test*
- *http://infolab.stanford.edu/~ullman/mmds/ch7.pdf*
- E. Haapalainen, S. Kim, J. F. Forlizzi, and A. K. Dey, “Psychophysiological measures for assessing cognitive load,” in *Proceedings of the 12th ACM international conference on Ubiquitous computing*, 2010, pp. 301–310.