Context and Activity Recognition (aka “Sensors + Inference”)

Varun Manjunath
CMSC 818G
Three Papers

1) Zhou, et al. IODetector: A Generic Service for Indoor Outdoor Detection


3) Lu, et al. The Jigsaw Continuous Sensing Engine for Mobile Phone Applications
IODetector: A Generic Service for Indoor Outdoor Detection
Uses of an Indoor Outdoor Detector?
Uses of an Indoor Outdoor Detector

- Indoor/Outdoor localization - GPS or WiFi?
  - GPS poor inside. WiFi poor outside.
- To vet the performance of GPS

- Tag locations/annotate smartphone photos automatically [1]
- Activity and context recognition [2]

What sensors onboard a smartphone would you use to build an IO detector?
What sensors onboard a smartphone would you use to build an IO detector?

- Light Detector
  - Intrinsic differences between natural (outdoor) and man-made (indoor) light

- Cellular Module
  - Cellular Signal strength varies between indoor and outdoor (dividing walls)

- Magnetism Sensor
  - Intensity of Magnetic Field varies due to presence of electrical appliances.

Also, proximity sensor, system clock and accelerometer. (Why?)

Most but not all Android smartphones are equipped with all five sensors.
Sensors

- Three sensors - each sensor isn’t perfect by itself. Then, aggregation

- Problems with light sensors
  - Light intensity changes with time, season, pose of phone, etc

- Problems with cellular tower
  - Varies significantly with location

- Problems with magnetism detector
  - Not accurate unless calibrated carefully
Some Requirements

- High Accuracy
  - duh!
- Prompt Response
  - no latency
- Shouldn’t drain the battery
- Universal applicability
  - sensors must be present on all devices
Three Indoor Outdoor Classes

Figure 1. Three indoor/outdoor environment types and the representative scenes.

Why bring in semi-outdoor?
Figure 2. System architecture of IODetector.
Light Detector

- During the day, irrespective of weather, $\text{Intensity}_{\text{indoor}} \ll \text{Intensity}_{\text{outdoor}}$

Figure 3. Mobile phone light sensor readings in different scenes.
Light Detector

- At night, \( \text{Intensity}_{\text{indoor}} > \text{Intensity}_{\text{outdoor}} \)
Light Detector

Thresholds are 2000 Lux and 50 Lux.

Confidence is calculated in the simplest possible way.

Also, man-made light flickers with frequency of alternating current.

Therefore, Fourier Transforms.
System Overview

Figure 2. System architecture of IODetector.
Cellular Detector

- Cellular Received Signal Strength (RSS)
- Dividing walls block signals
- When the user walks indoors from outdoors, RSS drops significantly.

- Why not WiFi RSS?
Cellular Detector

- Usually connect to one tower (with strongest RSS), but like [3], uses all visible cell towers.

- Issues
  - Handovers
  - Corner Effect

Cellular Detector

- Over multiple cell towers.
- Idea: If aggregate RSS drops - outdoor -> indoor, and vice versa.
System Overview

Figure 2. System architecture of IODetector.
Magnetism Detector

- Earth’s magnetic field is distorted due to electrical and steel appliances.
  - Eg : At equator - 0.25Gauss, At poles - 0.65Gauss
  - Fridge magnet - 100Gauss

- Magnetic Field shows greater variance when user is indoors (due to appliances), than outdoors, except when ___________.
Magnetism Detector

- Limitation?
  - What other sensor would help?

**Figure 8. The variation of magnetic field intensity.**
Advantages/Disadvantages

<table>
<thead>
<tr>
<th>Sensor</th>
<th>Advantage</th>
<th>Disadvantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Light</td>
<td>Rapid</td>
<td>Doesn’t work if phone is in pocket or bag.</td>
</tr>
<tr>
<td>Cellular</td>
<td>Natural function of phone, no battery drain</td>
<td>Slow, Requires sufficient cell tower coverage</td>
</tr>
<tr>
<td>Magnetism</td>
<td>It can distinguish between semi-outdoor and outdoor</td>
<td>Only works when user moves, not reliable.</td>
</tr>
</tbody>
</table>

- Aggregate them.
  - Sum up individual confidences (stateless)
  - HMMs (stateful)
Results

- Case study: GPS performs poorly indoors, but performs well outdoors.
  - Use IO-Detector to vet GPS performance. If indoors, use alternative means for localization.
Conclusion

- IODetector uses 3 sensors - light, cellular RSS and magnetism.
- Aggregates results of three.
- Tested on Android phones.

- Cast study to infer GPS availability.
Questions ?
Extended Presence

Aliener

Wut u do

New as of Wed Apr 29 2009 20:24:07 GMT-0400 (EDT)

Friends

Ronald Peterson

New as of Wed May 06 2009 01:30:37 GMT-0400 (EDT)

Location

Wed Apr 29 2009 20:24:07 GMT-0400
text: wut u do

Address

Powered by Google

Mao data ©2009 Tele Atlas - Terms of Use
Introduction
Introduction

- Automated answers to “where r u?” or “what r u doing?”

- Nokia N95, Symbian, Java Micro Edition

- sensing, classification + sharing on a social network
Some issues

- Continuous Sensing / Always on softwares drain battery

- Cannot disrupt normal functioning like phone calls
Split Level Classification

- Do “some” processing/classification on client, push it to server. Data to “Primitives”.
- Do “higher level” processing/classification on server. “Primitives” to “Facts”.

eg: Primitives - talking, loud noise, running. Fact - “party”
Primitives are:

1) Sound classification
2) Accelerometer classification
3) Nearby Bluetooth Mac Addresses
4) GPS readings
5) Random photos
Phone Software Architecture

- Run as a daemon
ClickStatus

- ClickStatus is the “front end”
Backend Architecture

Figure 3: Software architecture of the CenceMe backend.
Front-end Classification

- Data to Primitives
Sound Classification

- Features - Mean and Std. deviation of DFT
- Classes - talking or no-talking

Figure 5: Discriminant analysis clustering. The dashed line is determined by the discriminant analysis algorithm and represents the threshold between talking and not talking.

(a) DFT of a human voice sample registered by a Nokia N95 phone microphone.

(b) DFT of an audio sample from a noisy environment registered by a Nokia N95 phone microphone.
Accelerometers

- Not particularly different from a spring mass system
Accelerometer Classification

- features - mean, std dev, number of peaks along three axes
- Decision tree classifier
- classes = {sitting, standing, walking, running}
- For training, phone in belt, bag, pocket

Figure 6: Accelerometer data collected by the N95 on board accelerometer when the person carrying the phone performs different activities: sitting, standing, walking, and running.
Backend Classification

- Primitives to Facts
- Eg. facts are : {Meeting, Partying, Dancing}, “do you go to the gym more”, “do you spend time in museums”, etc.
- Rules like “if you are in a conversation, and background noise is high” → party
- Party + “running” → dancing
Backend Classification

- Social Context - Scans bluetooth addresses to find nearby friends.

Eg: Are you alone, or are you with other “Cenceme buddies”

- Mobility (using accelerometer - vehicle or not)
- Location (using GPS)
Backend Classification - Am I Hot?

- Nerdy
- Party Animal
- Cultured
- Healthy
- Greeny
Results

(a) Conversation classifier in different locations.

Table 1: Activity classifier confusion matrix

<table>
<thead>
<tr>
<th></th>
<th>Sitting</th>
<th>Standing</th>
<th>Walking</th>
<th>Running</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sitting</td>
<td>0.6818</td>
<td>0.2818</td>
<td>0.0364</td>
<td>0.0000</td>
</tr>
<tr>
<td>Standing</td>
<td>0.2096</td>
<td>0.7844</td>
<td>0.0060</td>
<td>0.0000</td>
</tr>
<tr>
<td>Walking</td>
<td>0.0025</td>
<td>0.0455</td>
<td>0.9444</td>
<td>0.0076</td>
</tr>
<tr>
<td>Running</td>
<td>0.0084</td>
<td>0.0700</td>
<td>0.1765</td>
<td>0.7451</td>
</tr>
</tbody>
</table>

Table 2: Conversation classifier confusion matrix

<table>
<thead>
<tr>
<th></th>
<th>Conversation</th>
<th>Non-Conversation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conversation</td>
<td>0.8382</td>
<td>0.1618</td>
</tr>
<tr>
<td>Non-Conversation</td>
<td>0.3678</td>
<td>0.6322</td>
</tr>
</tbody>
</table>

Table 3: Mobility mode classifier confusion matrix

<table>
<thead>
<tr>
<th></th>
<th>Vehicle</th>
<th>No Vehicle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vehicle</td>
<td>0.6824</td>
<td>0.3176</td>
</tr>
<tr>
<td>No Vehicle</td>
<td>0.0327</td>
<td>0.9673</td>
</tr>
</tbody>
</table>

Figure 7: Activity classification vs. body position.
Energy Consumption

- Battery Drainers
  - GPS
  - Bluetooth scan
  - DFT and classification

- Other factors
  - distance to cell tower
  - data to upload

Mean battery life is 6.2 hours
User Study

- 12 undergrads + 7 grad students
+ 1 Research assistant + 1 Professor + 1 staff

- Location feature most frequently used
- Some people turned off the “random photos” features

Improvements:
1) Better battery life
2) Better privacy settings
3) Shorter classification time

CenceMe makes the following claims
1) Almost everyone liked it - alright
2) Facebook users are more willing to share their personal information than others - okay
3) Privacy could be a concern, but there are settings
4) People want to know what others are doing - debatable
5) People can discover their own activity patterns and social status - cool
Questions ?
The Jigsaw Continuous Sensing Engine for Mobile Phone Applications
What is a Continuous Sensing Application?
Continuous Sensing Applications

- Continuous Sensing Applications
  - typically always on, read from sensors, make accurate inferences
- But, must maintain battery life, and not overload CPU.
  - shouldn’t compromise the normal functioning of the phone (calls, messages, internet, email)

- How would you maintain battery life, but also make accurate inferences?
Contributions

- Make accelerometer more resilient
  - body positions
- Reduce microphone sensing and computing costs
  - Don’t send silences to GMM classifier, etc
- Sample GPS judiciously
  - Sitting in an office vs Travelling on the highway
Accelerometer

- Accelerometer is cheap but good inference is difficult because of:
  1) Body positions (Eg: hip pocket, bag)
  2) Extraneous activities (Eg: messaging, calling)
  3) Transition states (Eg: taking phone out, etc)

Figure 1: Activity inferences are inaccurate when the phone is in a backpack and the model was trained for a front pocket.
Accelerometer Pipeline

1) Calibration :
   - Keep the phone absolutely steady (why?)
   - $g_{\text{axis}} = K_{\text{axis}} a_{\text{axis}} + b_{\text{axis}}$
   - Least squares
   - can be made automatic

2) Normalize : Convert to units of g

3) Admission control
   - filter out extraneous transitions
Accelerometer Pipeline

4) Projections:
- Convert from local to global coordinates
- Makes data insensitive to orientation

5) Feature Extraction:

<table>
<thead>
<tr>
<th>Time domain</th>
<th>mean, variance, mean crossing rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency domain</td>
<td>spectrum peak, sub-band energy, sub-band energy ratio, spectral entropy</td>
</tr>
</tbody>
</table>

Table 1: Motion Feature Set

Figure 4: Jigsaw Accelerometer Pipeline
Accelerometer Pipeline

6) Activity Classification:

- Decision tree
- Sub-classes

<table>
<thead>
<tr>
<th>Category</th>
<th>Sub-class</th>
</tr>
</thead>
<tbody>
<tr>
<td>stationary</td>
<td>sitting, standing, on the table</td>
</tr>
<tr>
<td>lower body</td>
<td>front and back pants pockets</td>
</tr>
<tr>
<td>in hand</td>
<td>in hand when reading the screen, armband, in hand when swinging naturally</td>
</tr>
<tr>
<td>all others</td>
<td>jacket pocket, backpack, belt</td>
</tr>
<tr>
<td>cycling</td>
<td></td>
</tr>
<tr>
<td>lower body</td>
<td>front and back pants pockets</td>
</tr>
<tr>
<td>upper body</td>
<td>jacket pocket, armband, backpack, belt</td>
</tr>
<tr>
<td>running</td>
<td></td>
</tr>
<tr>
<td>running</td>
<td>all body positions</td>
</tr>
<tr>
<td>vehicle</td>
<td></td>
</tr>
<tr>
<td>vehicle</td>
<td>car, bus, light rail</td>
</tr>
</tbody>
</table>

Table 2: Activity Subclasses

Figure 4: Jigsaw Accelerometer Pipeline
Microphone

- Sampling rate and sound classification increases load on CPU.

- Tasks are: “human voice or not”, and “reading”, “chatting”, “walking”, “toilet flush”. (20 classes)
Microphone Pipeline

1) Admission Control:
   - If RMS intensity of sound below thresh. don’t process.

2) Duty cycle:
   - If user is sleeping, quiet, don’t process

3) Features:

<table>
<thead>
<tr>
<th>Category</th>
<th>Feature set</th>
</tr>
</thead>
<tbody>
<tr>
<td>voice</td>
<td>Spectral Rolloff [12], Spectral Flux [25], Bandwidth [12], Spectral Centroid [12], Relative Spectral Entropy [6], Low Energy Frame Rate [25]</td>
</tr>
<tr>
<td>other activities</td>
<td>13 MFCC coefficient feature set [32], Spectral Centroid [12], Bandwidth [12], Relative Spectral Entropy [6], Spectral Rolloff [12]</td>
</tr>
</tbody>
</table>

Figure 5: Jigsaw Microphone Pipeline
Microphone Pipeline

4) Voice Classification:
   - Voice vs No Voice
   - Decision tree

5) Activity Classification:
   - GMM + Classifier

6) Similarity Detector
   - If nearby samples are very similar (cosine distance), give them same labels.
GPS Pipeline

- What is the factors to take into consideration when sampling GPS?
GPS Pipeline

- What is the factors to take into consideration when sampling GPS?

Ans:
1) User’s mobility
2) Battery remaining
GPS Pipeline

- Markov Decision Process
- Sampling freq : Every 20min, 10min, 5 min, 2 min, 1 min, 5 sec
- The bottomline is :
  - for every (accelerometer, battery, time) tuple, a sampling rate is available
  - stored as table

If the energy budget is breached too quickly, there is a penalty. Reward = -penalty.
GPS Pipeline example

Figure 14: Jigsaw Emulated on a Weekend Trace
Results - Read the paper!

- Accelerometer - Close to 95% accuracy
- Microphone - ~85% on “voice vs no voice”
- GPS - low error, low power usage. (2nd best model after accelerometer augmented GPS)
Case Study

- JigMe - an automatic log of daily activities
- GreenSaw - daily calorie expenditure and carbon footprint (accelerometer)

Figure 16: (a) The GreenSaw application provides carbon footprint and caloric expenditure information, and (b) The JigMe application provides a log of daily activities, significant places, and transportation methods.
Questions ?