PBN: Towards Practical Activity Recognition Using Smartphone-based Body Sensor Networks

CSCI 780 Sensors & Ubiquitous Computing

Gang Zhou

Computer Science

College of William and Mary

Based on slides from Matthew Keally

Introduction

- Body Sensor Networks
  - Athletic Performance
  - Health Care
  - **Activity Recognition**

On-Body Sensors
+Sensing Accuracy
+Flexibility

Phone
+User Interface
+Computational Power
+Additional Sensors

ator

Introduction

Activity Recognition Challenges | **Solutions**

*A **practical** approach*

- Portable and user friendly — **TinyOS-based motes + Android phone**
- Computationally lightweight — **Activity recognition approach appropriate for phones**; **Identify redundant sensors to reduce training costs**
- Accurate — **Classify difficult activities with nearly 90% accuracy**
- Not Invasive — **Retraining detection without ground truth**

Related Work

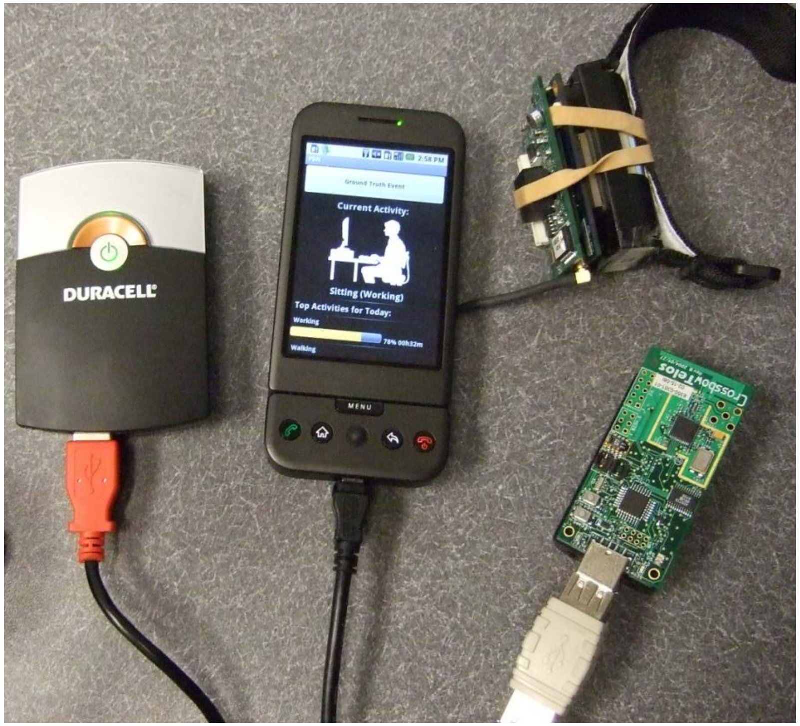
- No mobile or on-body aggregator
  - (Ganti, MobiSys'06), (Lorincz, SenSys'09), (Zappi, EWSN'08)
- Use of backend servers
  - (Miluzzo, MobiSys'10), (Miluzzo, SenSys'08)
- Single sensor modality or separate classifier per modality
  - (Azizyan, MobiCom'09), (Kim, SenSys'10), (Lu, SenSys'10)
- Do not provide online training
  - (Wang, MobiSys'09), (Wachuri, UbiComp'10)

Outline

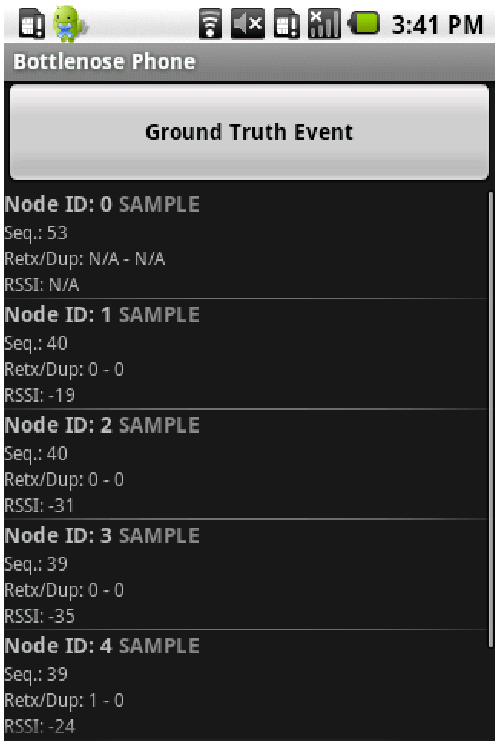
- Introduction
- Related Work
- Hardware and Software
- Experimental Setup
- PBN System Design
- Evaluation
- Conclusion & Discussion

Hardware: TinyOS + Android

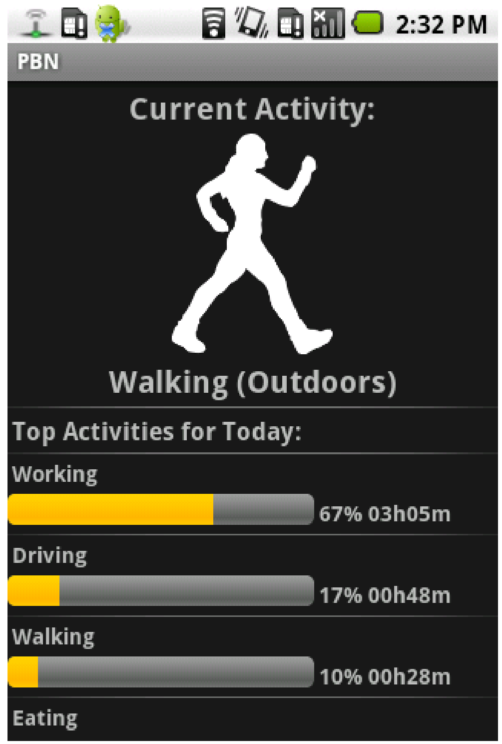
- IRIS on-body motes, TelosB base station, G1 phone
- Enable USB host mode support in Android kernel
- Android device manager modifications
- TinyOS JNI compiled for Android



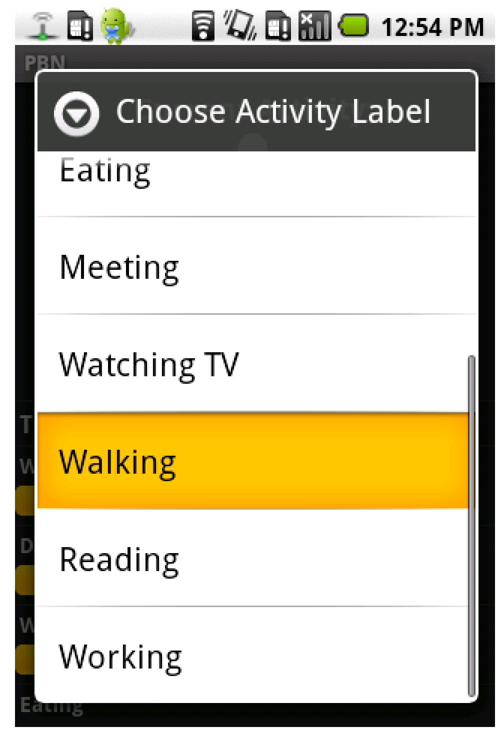
Software: Android Application



Sensor Configuration



Runtime Control and Feedback

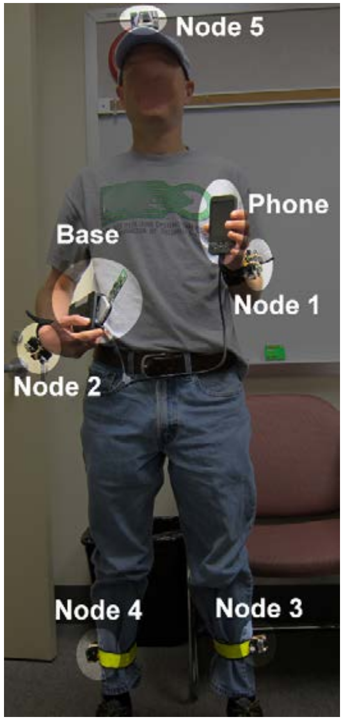


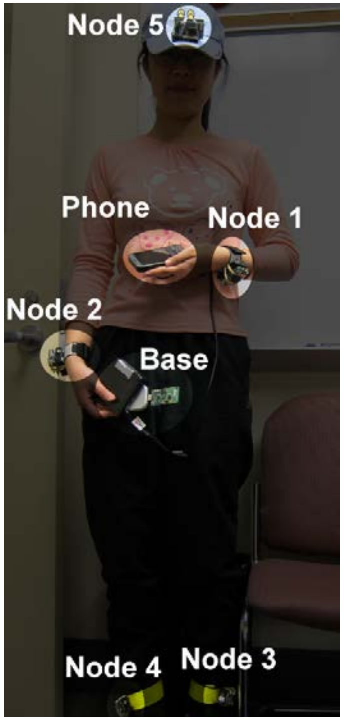
Ground Truth Logging

Data Collection Setup

- 2 subjects, 2 weeks
- Android Phone
  - 3-axis accelerometer, WiFi/GPS Localization
- 5 IRIS Sensor Motes
  - 2-axis accelerometer, light, temperature, acoustic, RSSI

Node ID	Location
0	BS/Phone
1	L. Wrist
2	R. Wrist
3	L. Ankle
4	R. Ankle
5	Head



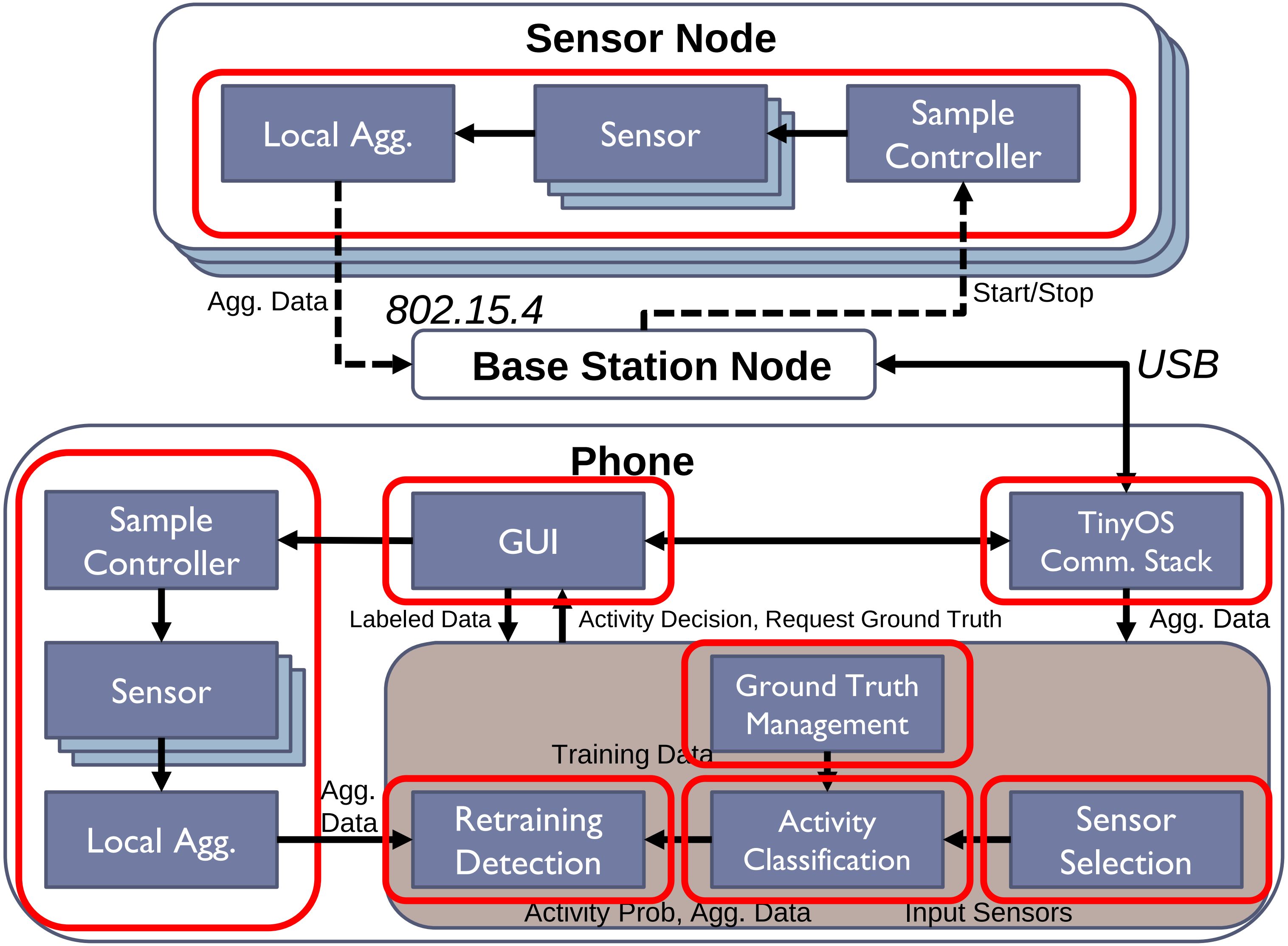


Data Collection Setup

- Classify typical daily activities, postures, and environment
- Previous work (Lester, et. al.) identifies some activities as hard to classify
- Classification Categories:

Environment	Indoors, Outdoors
Posture	Cycling, Lying Down, Sitting, Standing, Walking
Activity	Cleaning, Cycling, Driving, Eating, Meeting, Reading, Walking, Watching TV, Working

PBN Architecture

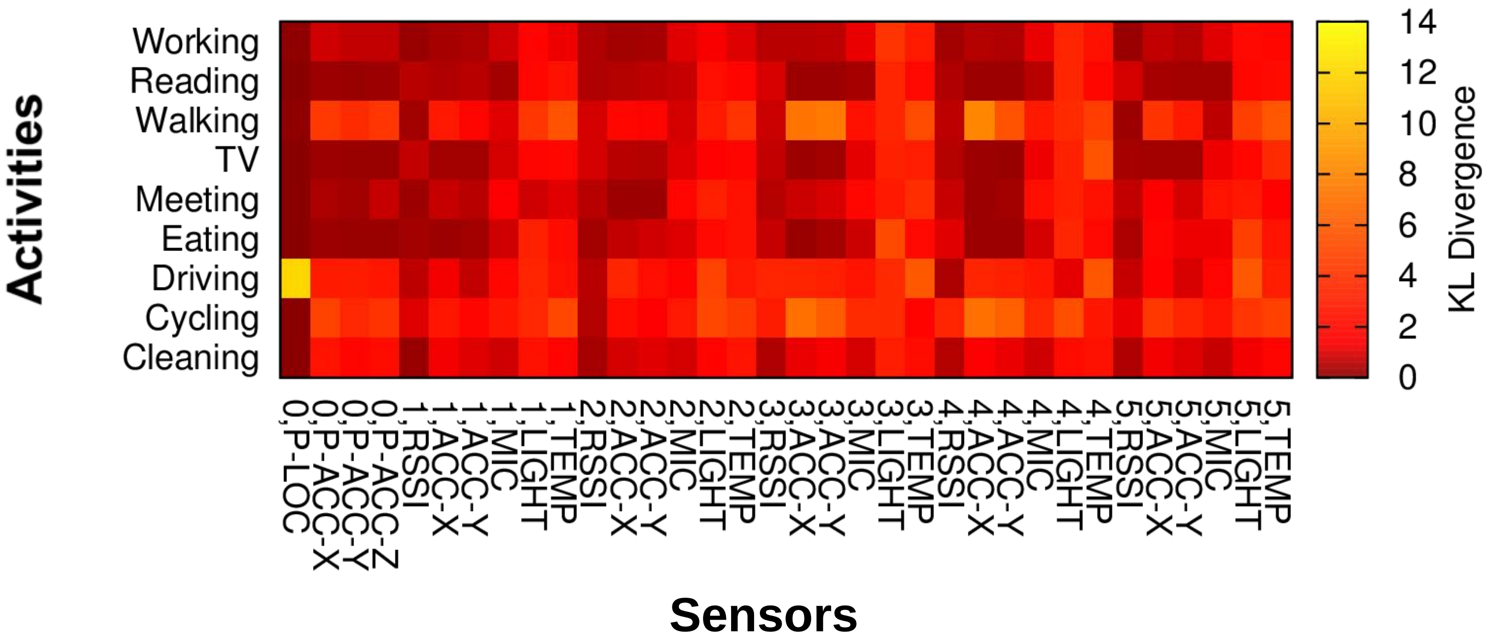


Retraining Detection

- **Body Sensor Network Dynamics** affects accuracy during runtime:
  - Changing physical location
  - User biomechanics
  - Variable sensor orientation
  - Background noise
- How to detect that retraining is needed without asking for ground truth?
  - Constantly nagging the user for ground truth is annoying
  - Perform with limited initial training data
  - Maintain high accuracy

Retraining Detection

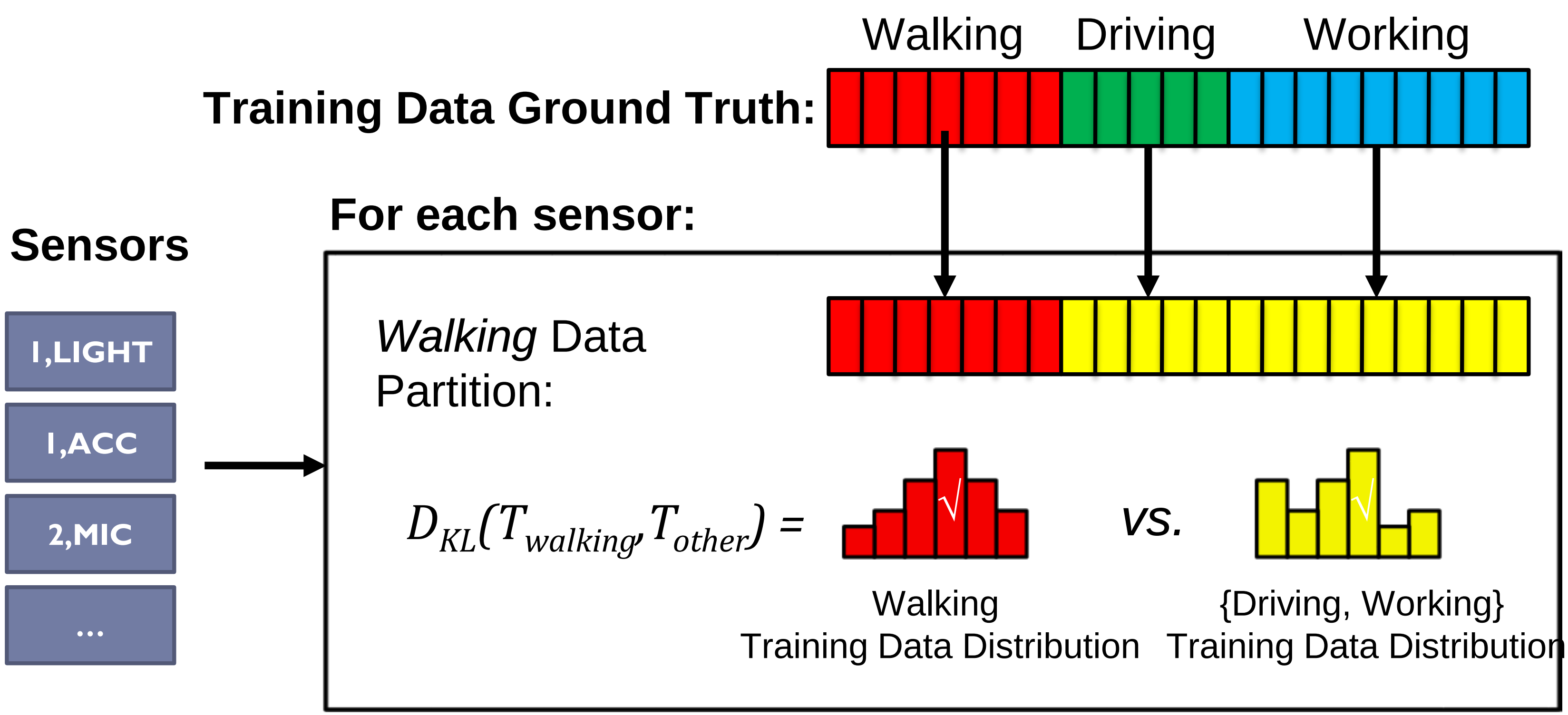
- Measure the discriminative power of each sensor: K-L divergence
  - ➢ Quantify the difference between sensor reading distributions



- Retraining detection with K-L divergence:
  - ➢ Compare training data to runtime data for each sensor

Retraining Detection

- Training
  - Compute “one vs. rest” K-L divergence for each sensor and activity



Retraining Detection

- Runtime
  - At each interval, sensors compare runtime data to training data for current classified activity

Sensors

Current AdaBoost Classified Activity: **Walking**

1,LIGHT

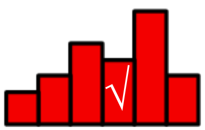
1,ACC

2,MIC

...

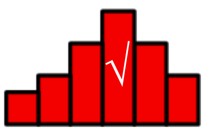
For each sensor:

$D_{KL}(R_{walking}, T_{walking})$ =



Walking
Runtime Data Distribution

vs.

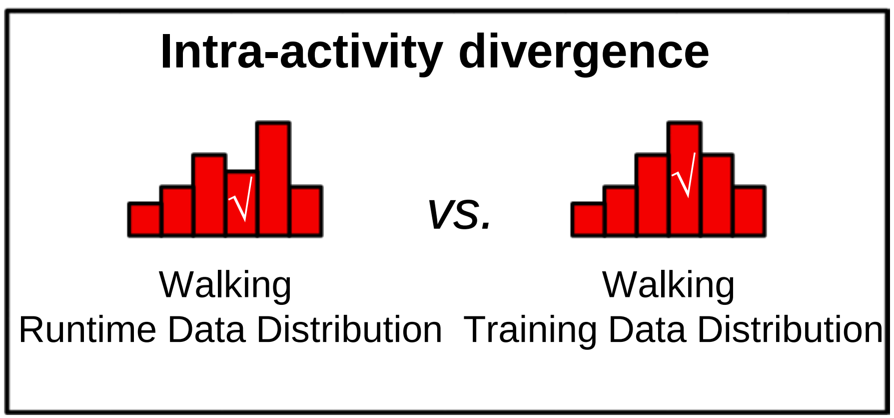


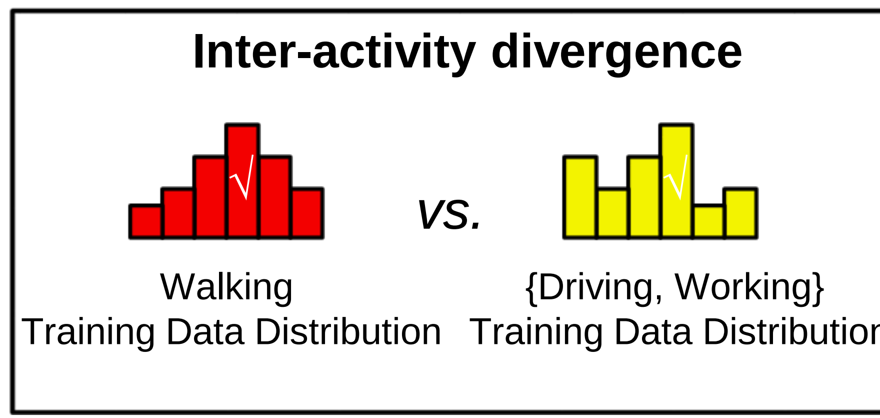
Walking
Training Data Distribution

Retraining Detection

- Runtime
  - At each interval, sensors compare runtime data to training data for current classified activity
  - Each individual sensor determines retraining is needed when:

$$D_{KL}(R_{walking}, T_{walking}) \quad > \quad D_{KL}(T_{walking}, T_{other})$$





Retraining Detection

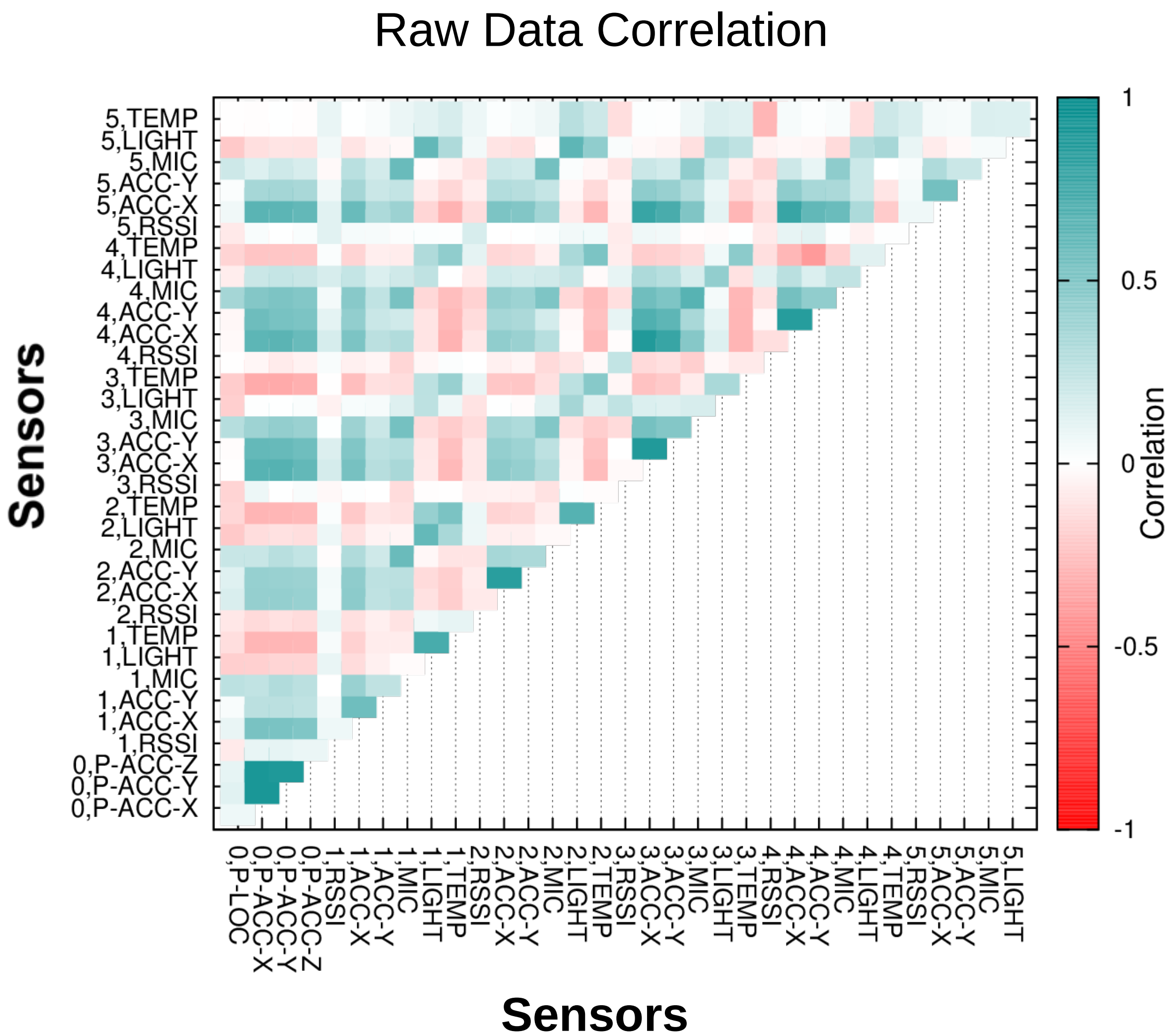
- Runtime
  - At each interval, sensors compare runtime data to training data for current classified activity
  - Each individual sensor determines retraining is needed
  - The ensemble retrains when a weighted majority of sensors demand retraining

Sensor Selection

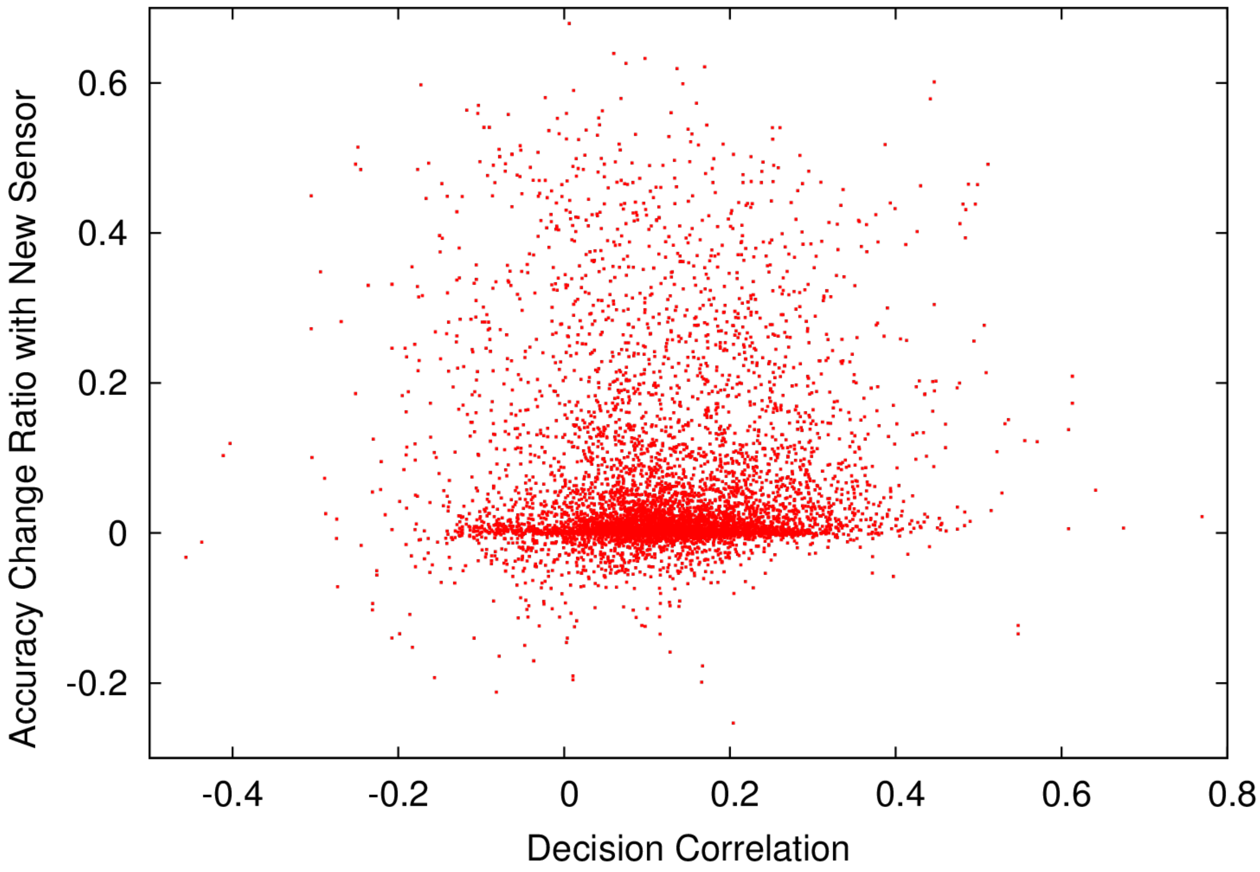
- AdaBoost training can be computationally demanding
  - Train a weak classifier for each sensor at each iteration
  - > 100 iterations to achieve maximum accuracy

- Can we give only the most helpful sensors to AdaBoost?
  - Identify both helpful and redundant sensors
  - Train fewer weak classifiers per AdaBoost iteration
  - Bonus: use even fewer sensors

Sensor Selection



Sensor Selection

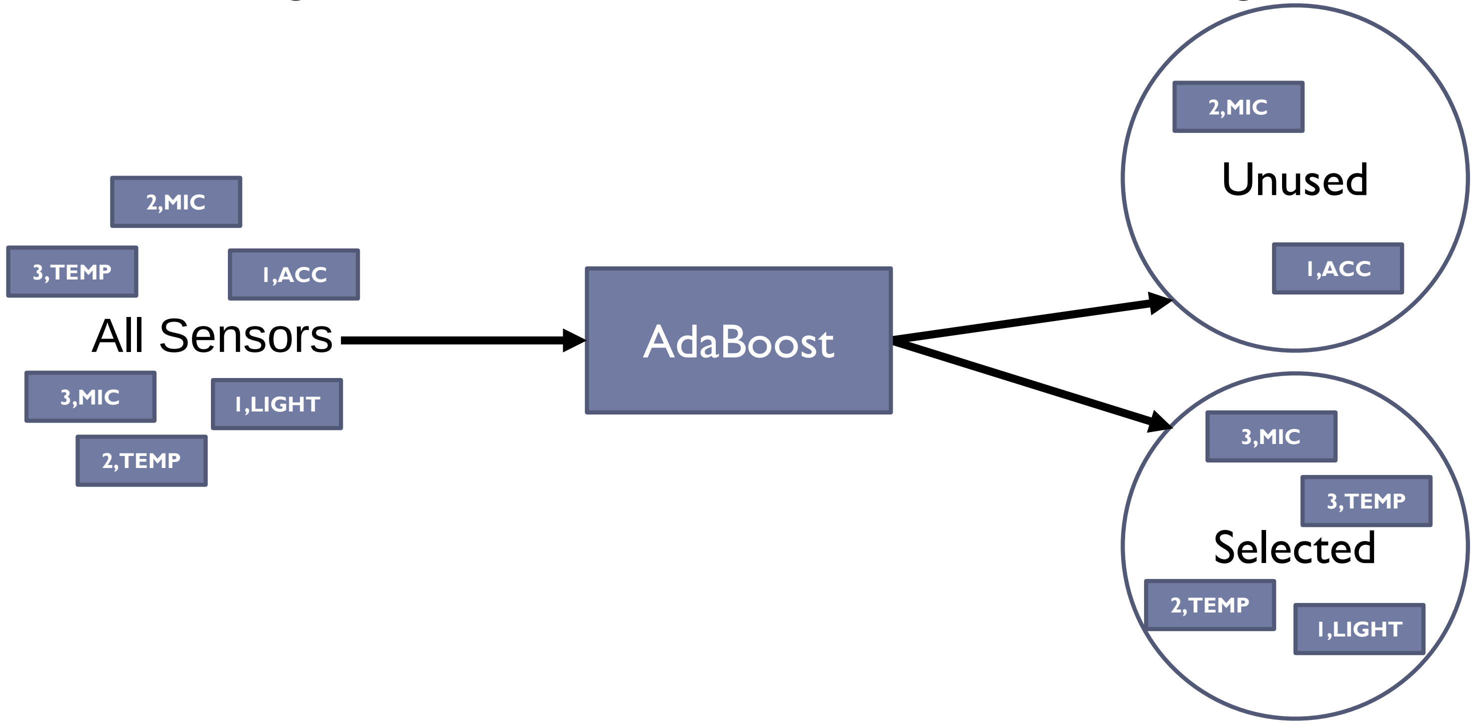


Choosing sensors with **slight correlation** yields the highest accuracy

Sensor Selection

Goal: determine the sensors that AdaBoost chooses using correlation

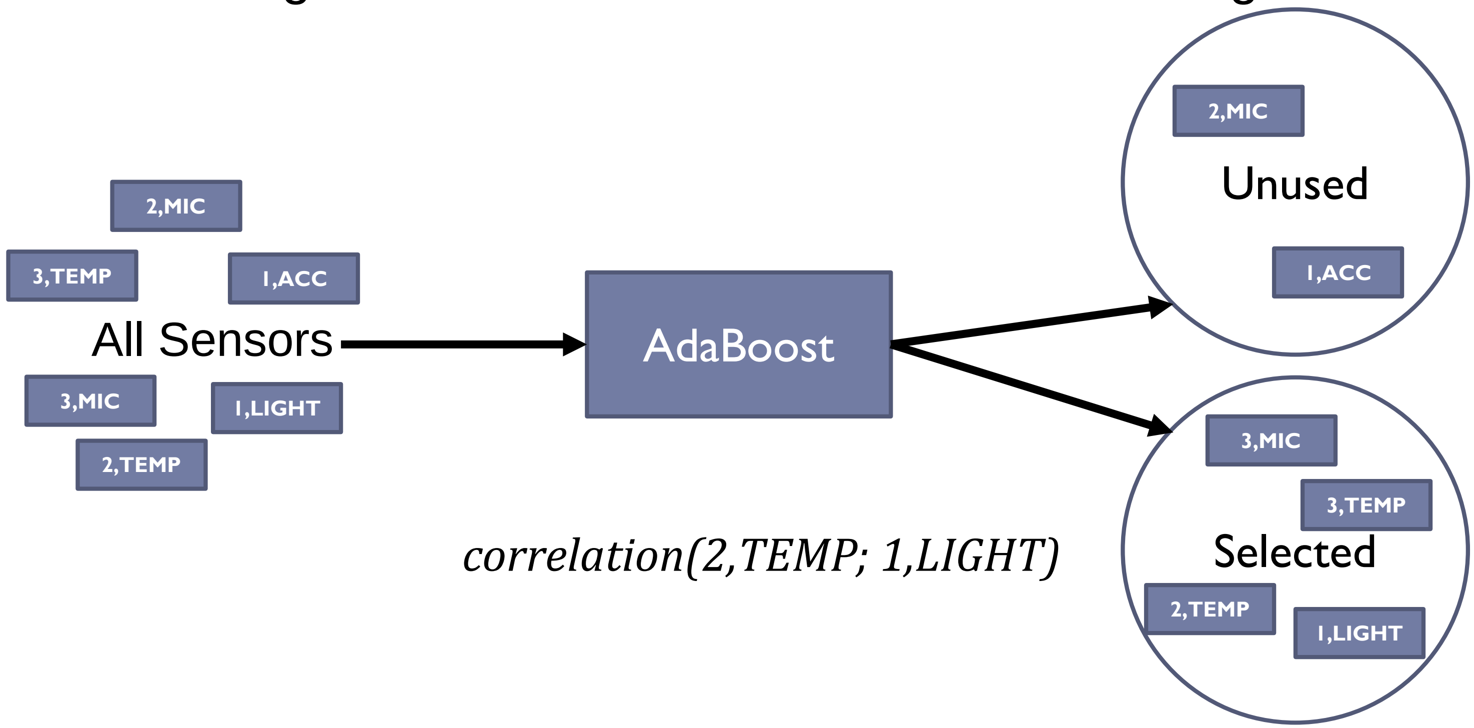
- Find the correlation of each pair of sensors selected by AdaBoost
- Use average correlation as a threshold for choosing sensors



Sensor Selection

Goal: determine the sensors that AdaBoost chooses using correlation

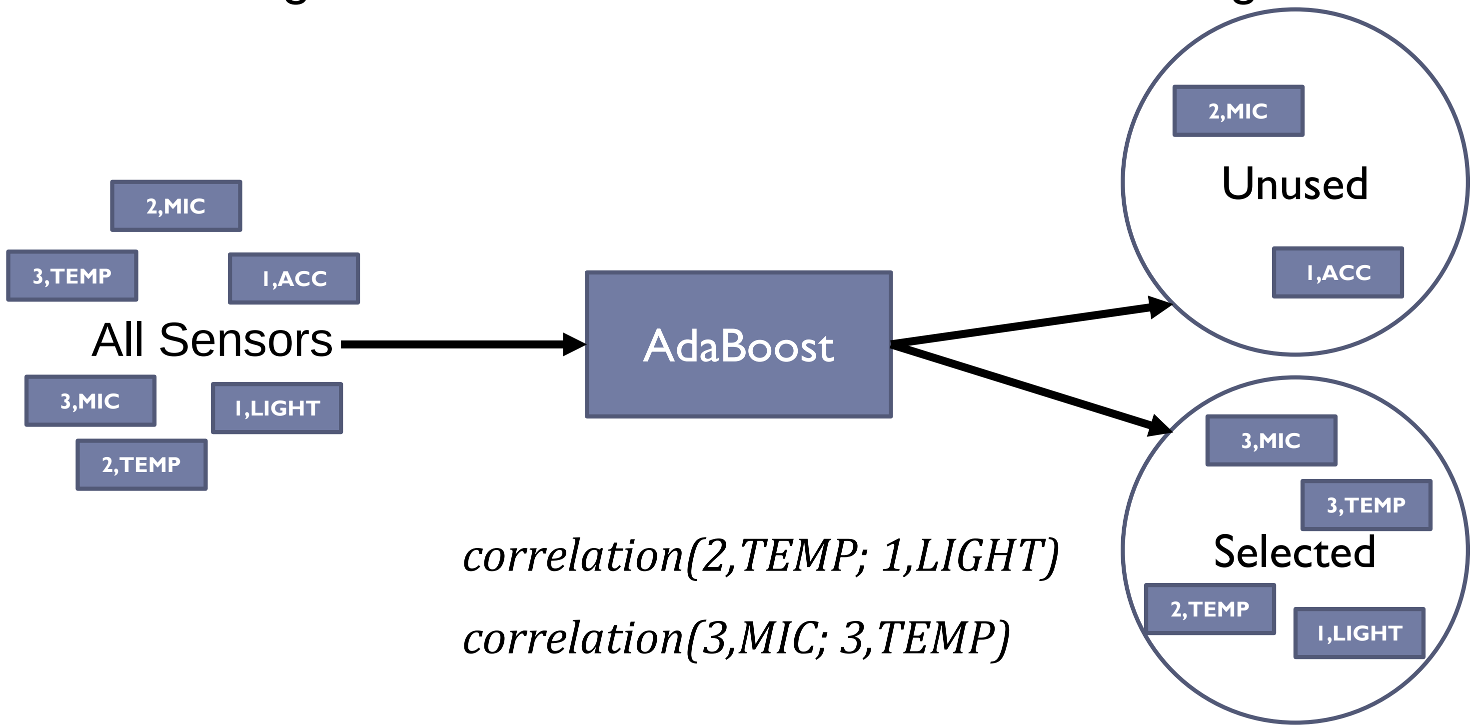
- Find the correlation of each pair of sensors selected by AdaBoost
- Use average correlation as a threshold for choosing sensors



Sensor Selection

Goal: determine the sensors that AdaBoost chooses using correlation

- Find the correlation of each pair of sensors selected by AdaBoost
- Use average correlation as a threshold for choosing sensors



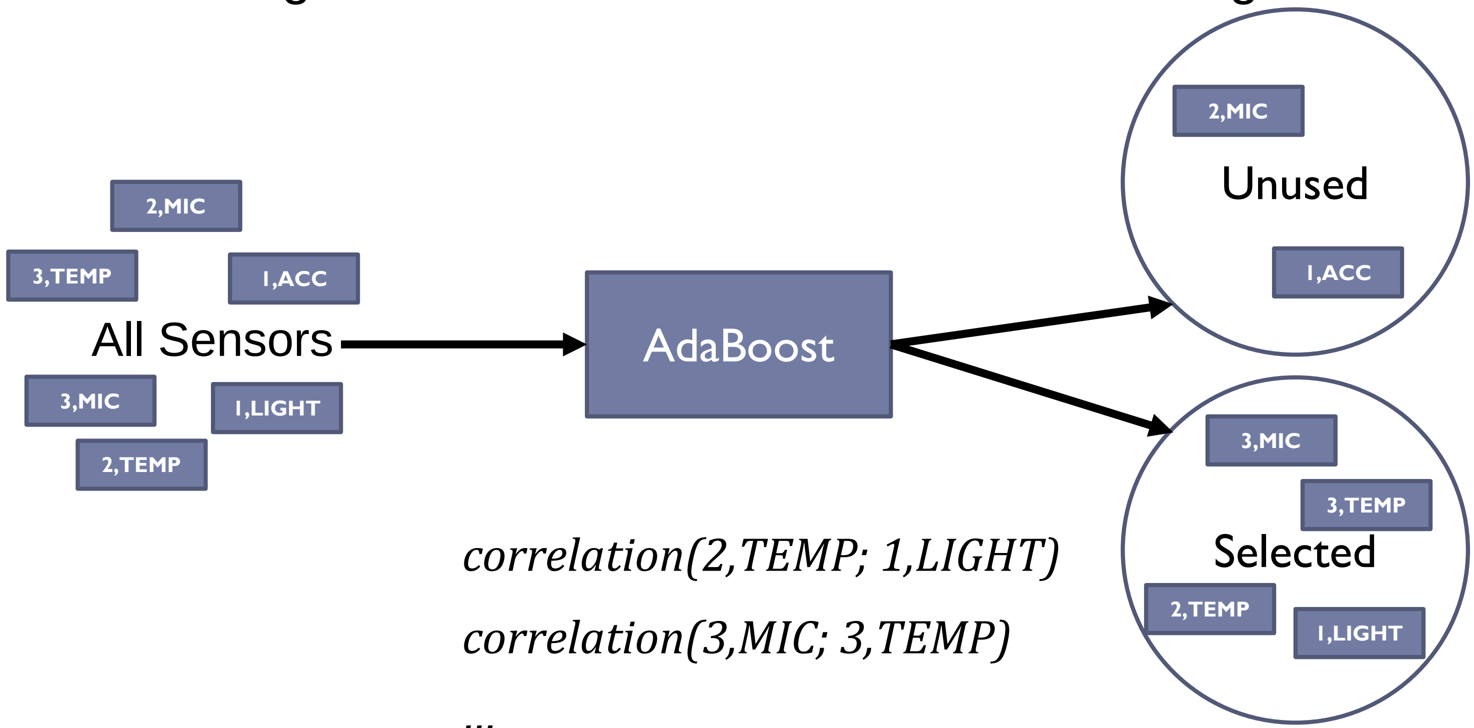
correlation(2,TEMP; 1,LIGHT)

correlation(3,MIC; 3,TEMP)

Sensor Selection

Goal: determine the sensors that AdaBoost chooses using correlation

- Find the correlation of each pair of sensors selected by AdaBoost
- Use average correlation as a threshold for choosing sensors



correlation(2,TEMP; 1,LIGHT)

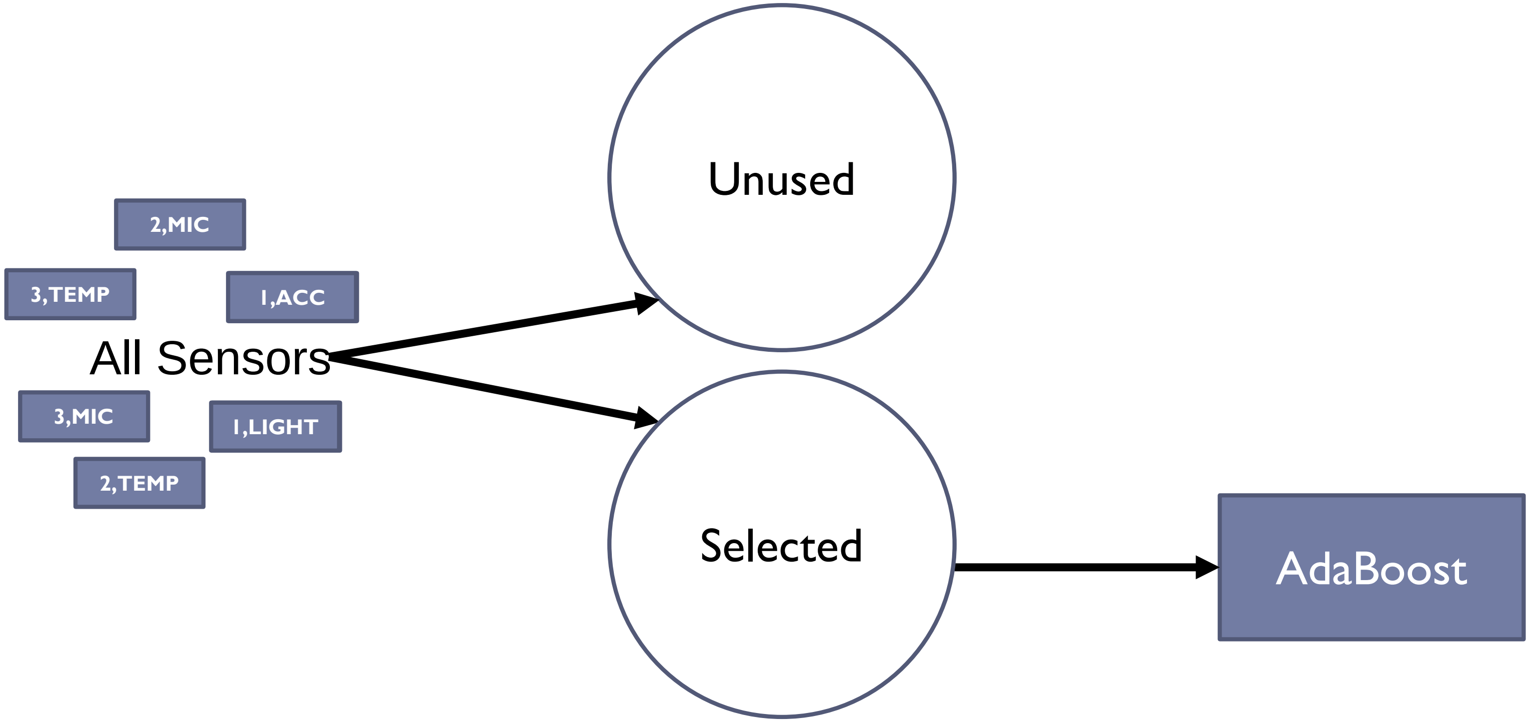
correlation(3,MIC; 3,TEMP)

...

Set threshold **α** based on average correlation: $\alpha = \mu_{corr} + \sigma_{corr}$

Sensor Selection

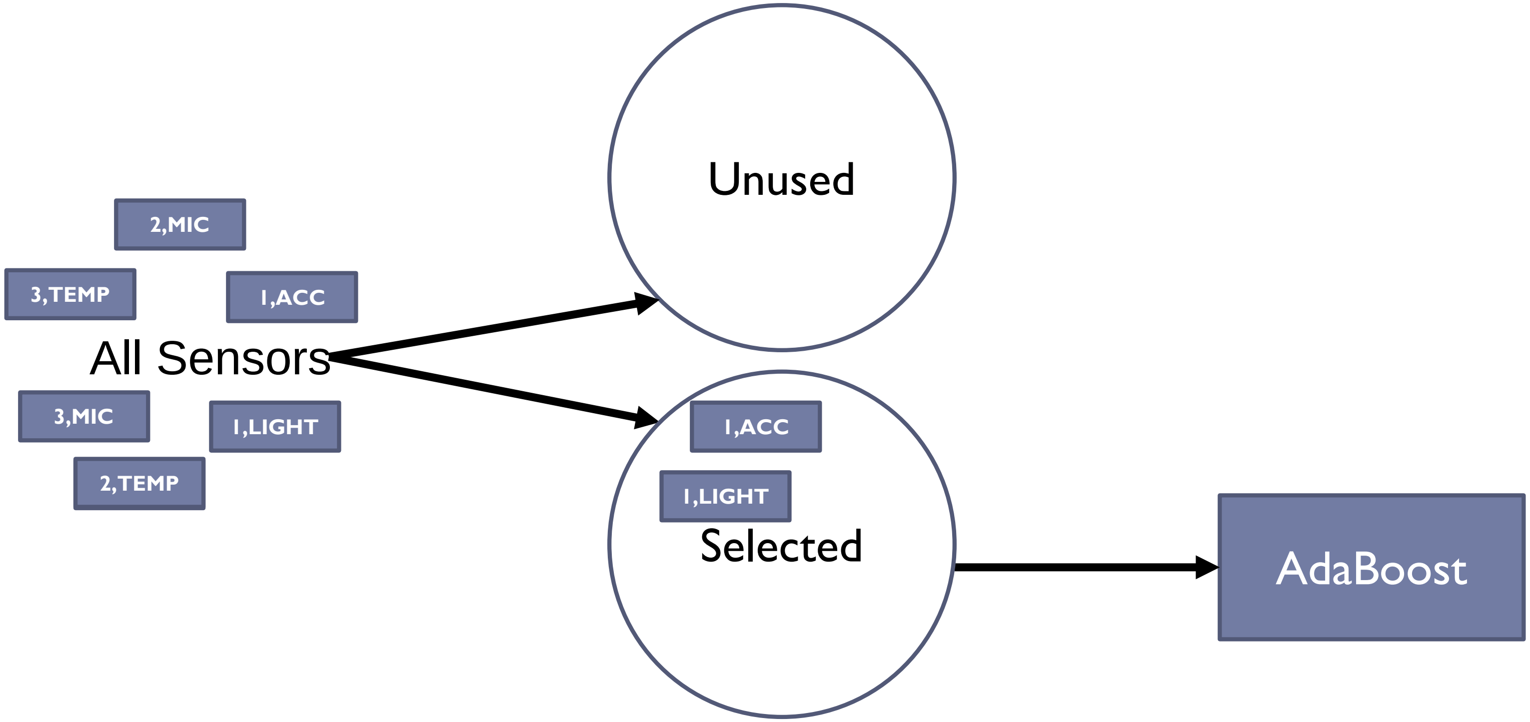
- Choose sensors for input to AdaBoost based on the correlation threshold



$$correlation(1,ACC;\ 1,LIGHT) \leq \alpha$$

Sensor Selection

- Choose sensors for input to AdaBoost based on the correlation threshold

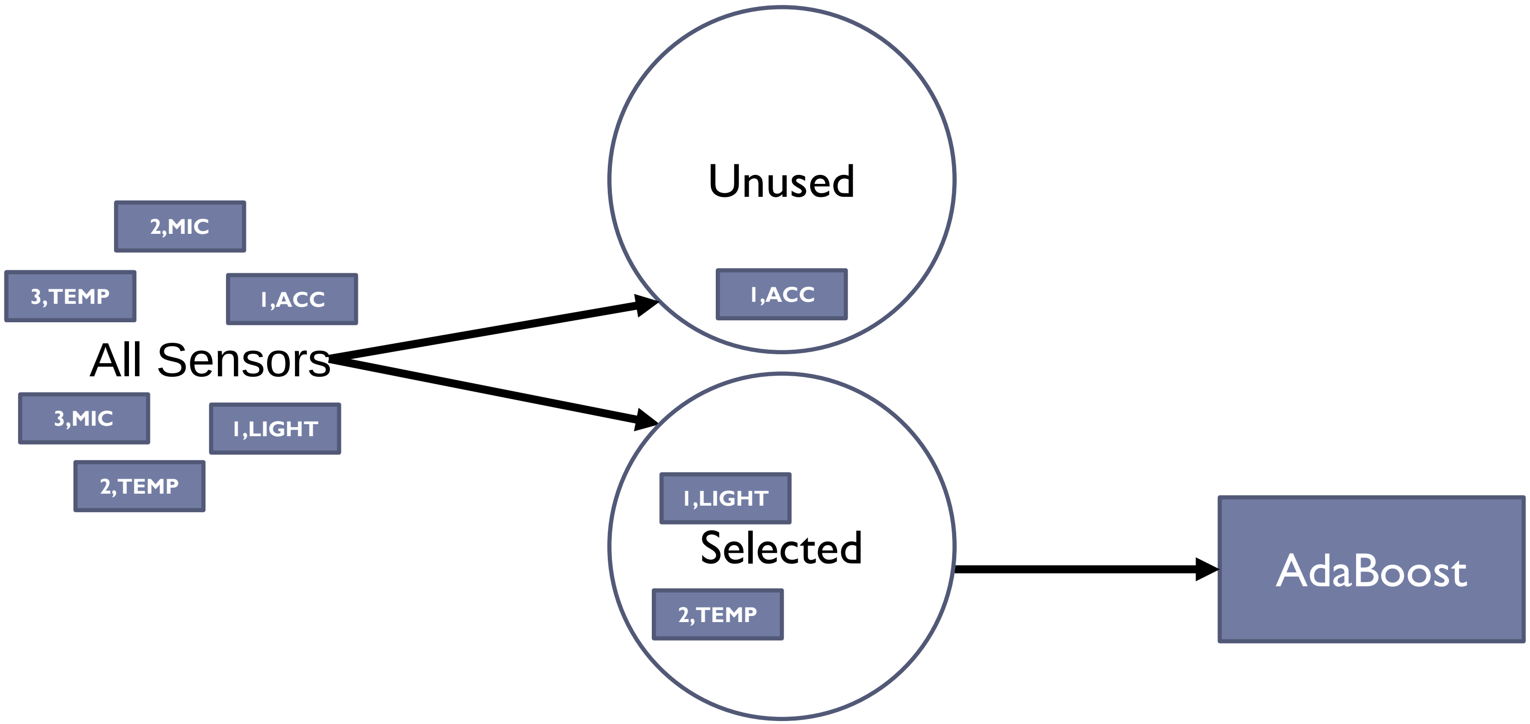


$$correlation(2,TEMP;\ 1,ACC) > \alpha$$

$$acc(2,TEMP) > acc(1,ACC)$$

Sensor Selection

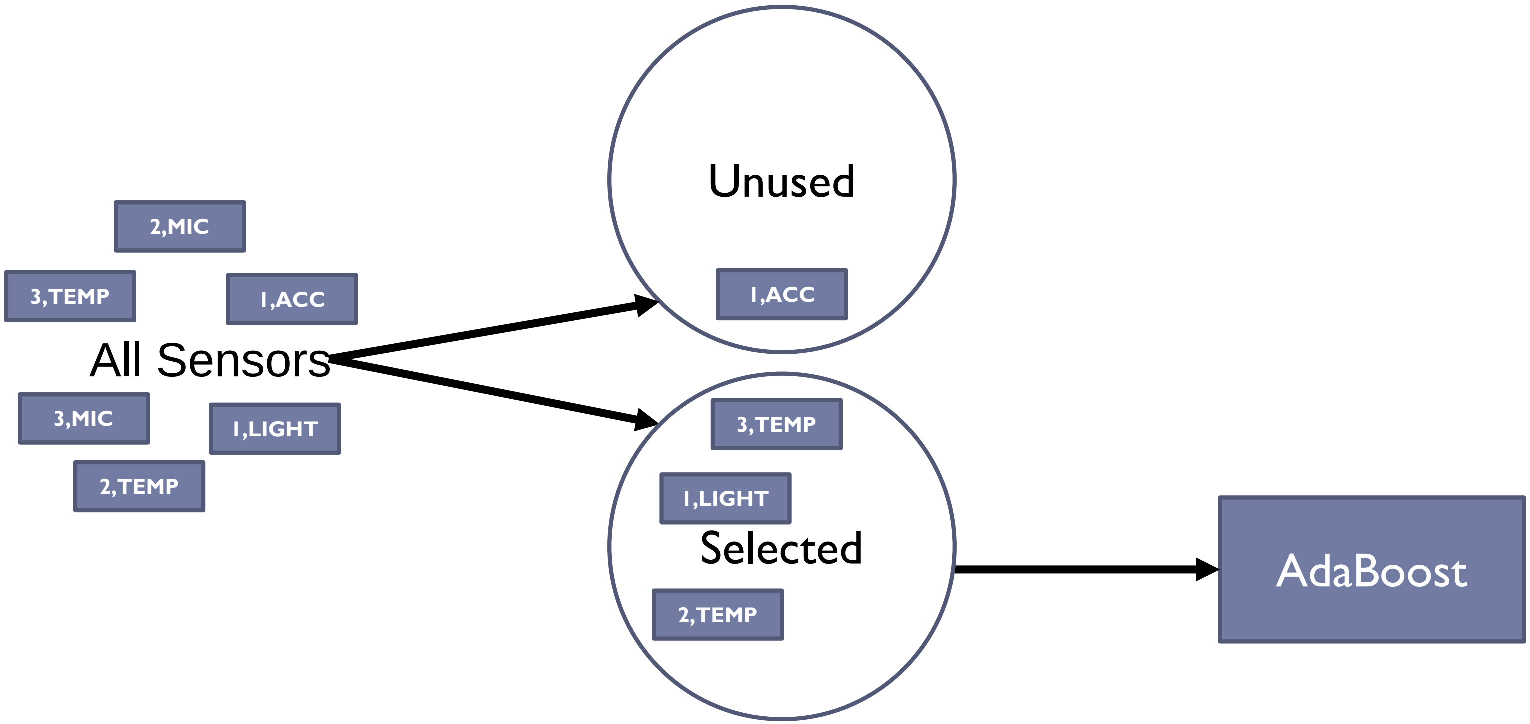
- Choose sensors for input to AdaBoost based on the correlation threshold



$$correlation(1,ACC;\ 3,TEMP) \leq \alpha$$

Sensor Selection

- Choose sensors for input to AdaBoost based on the correlation threshold

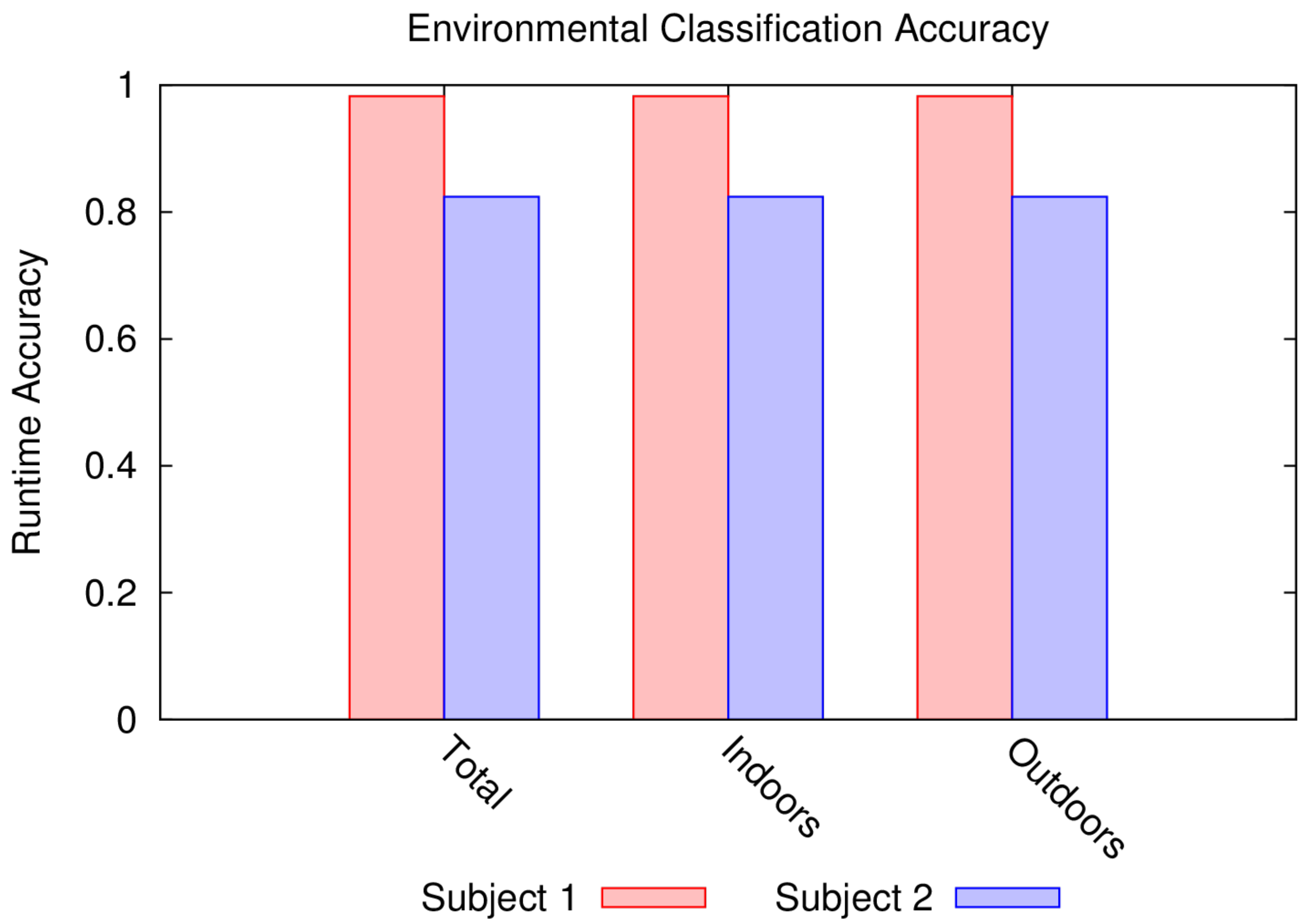


Evaluation Setup

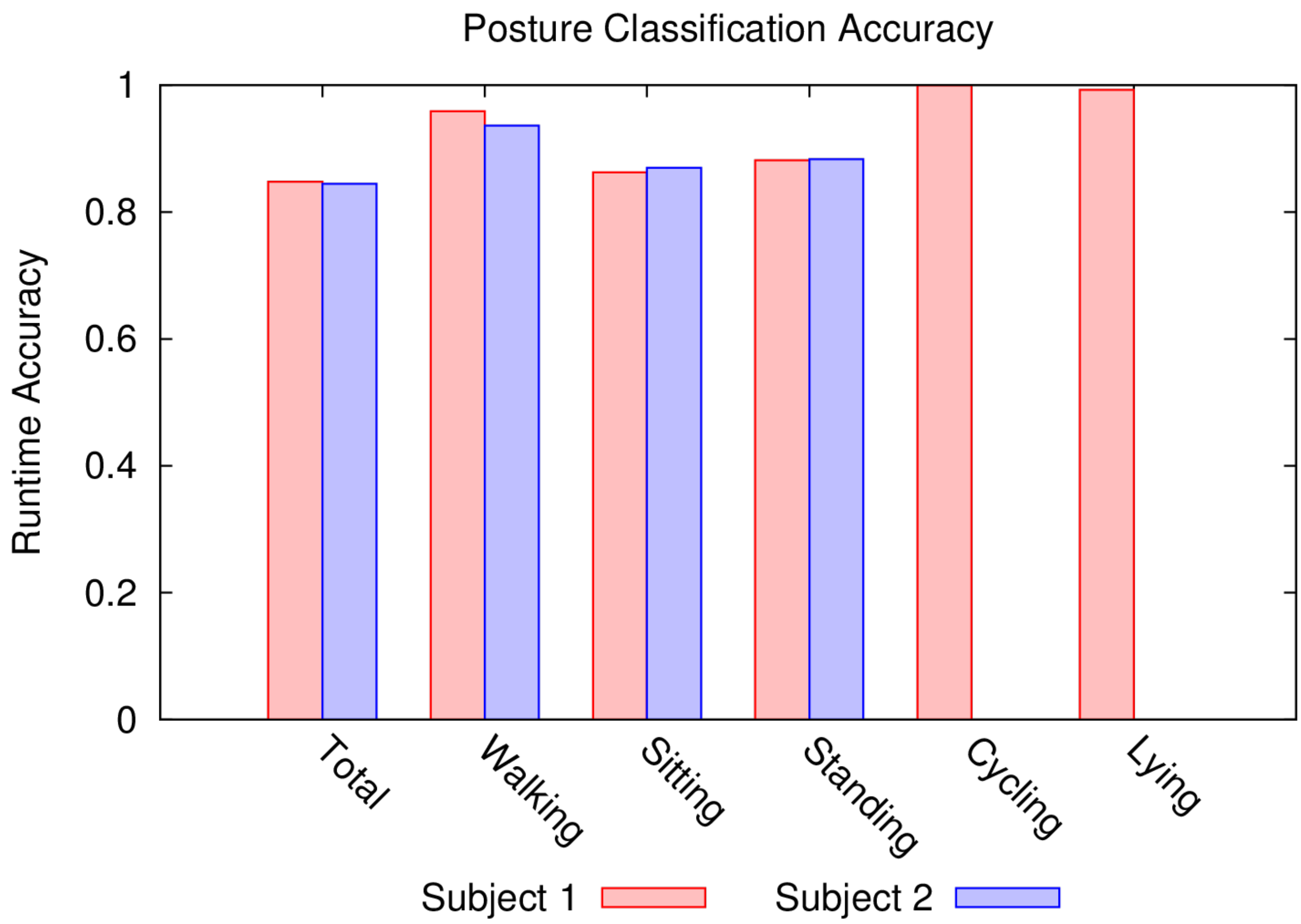
- 2 subjects over 2 weeks
- Classify typical daily activities, postures, and environment

Environment	Indoors, Outdoors
Posture	Cycling, Lying Down, Sitting, Standing, Walking
Activity	Cleaning, Cycling, Driving, Eating, Meeting, Reading, Walking, Watching TV, Working

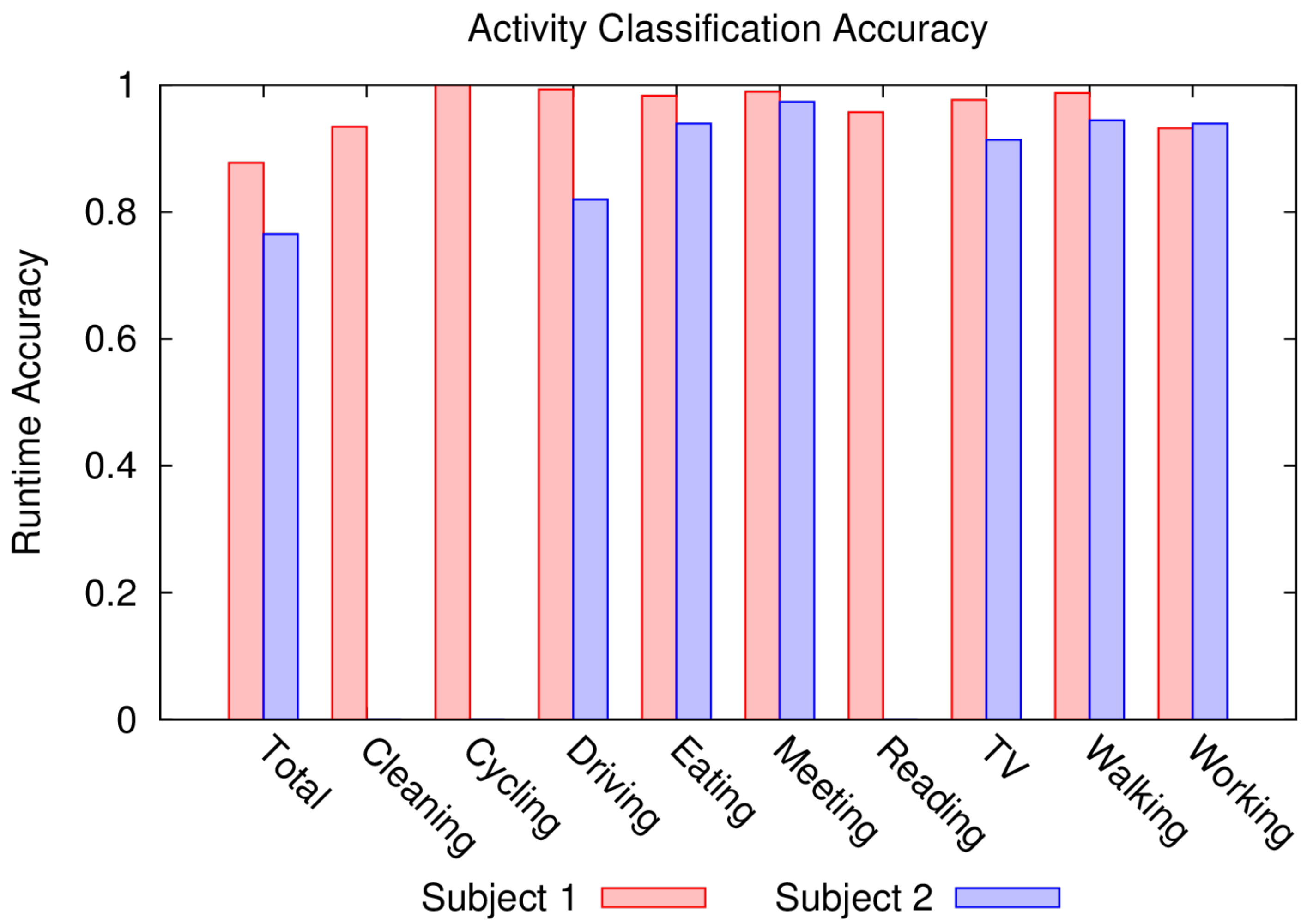
Classification Performance



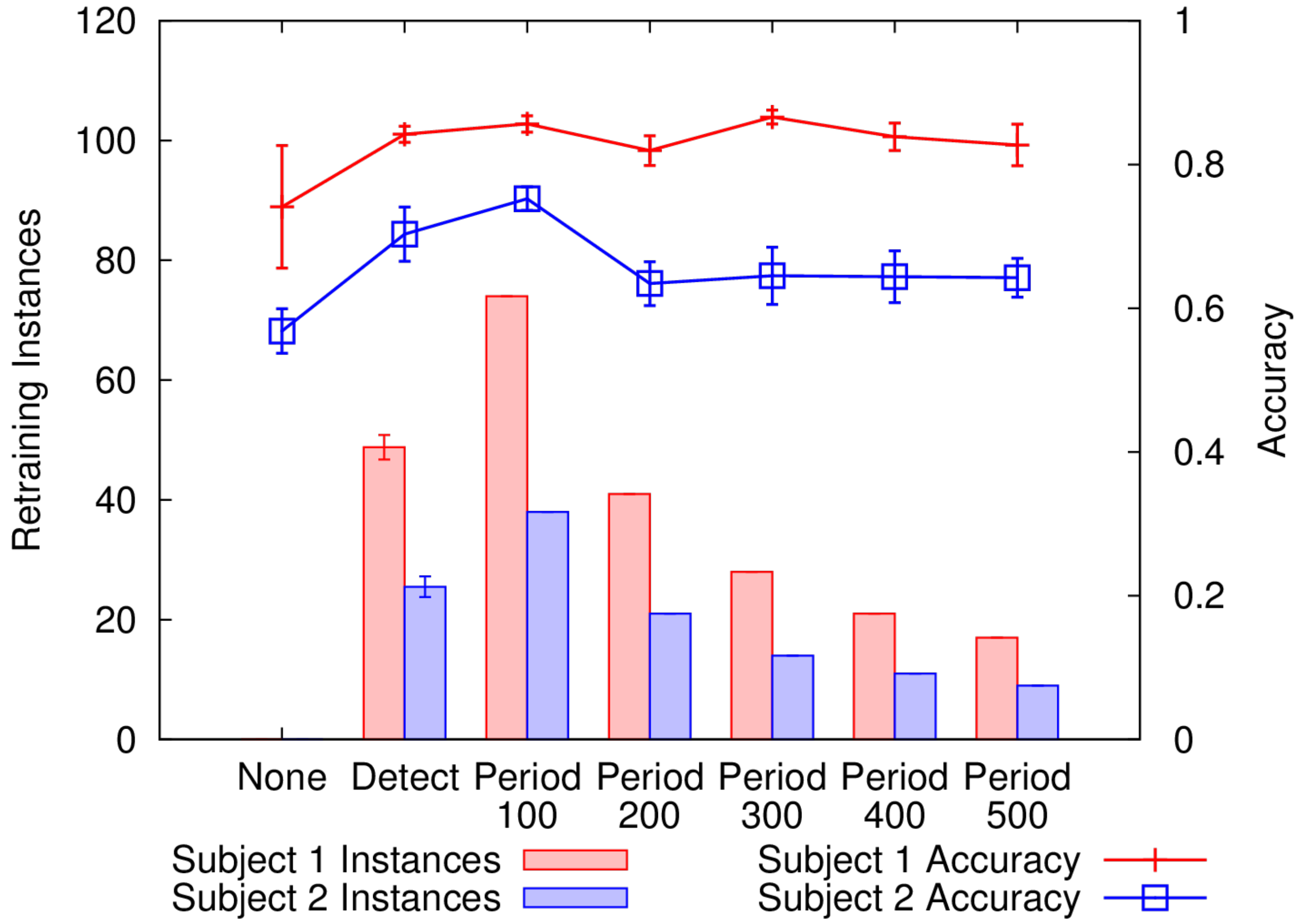
Classification Performance



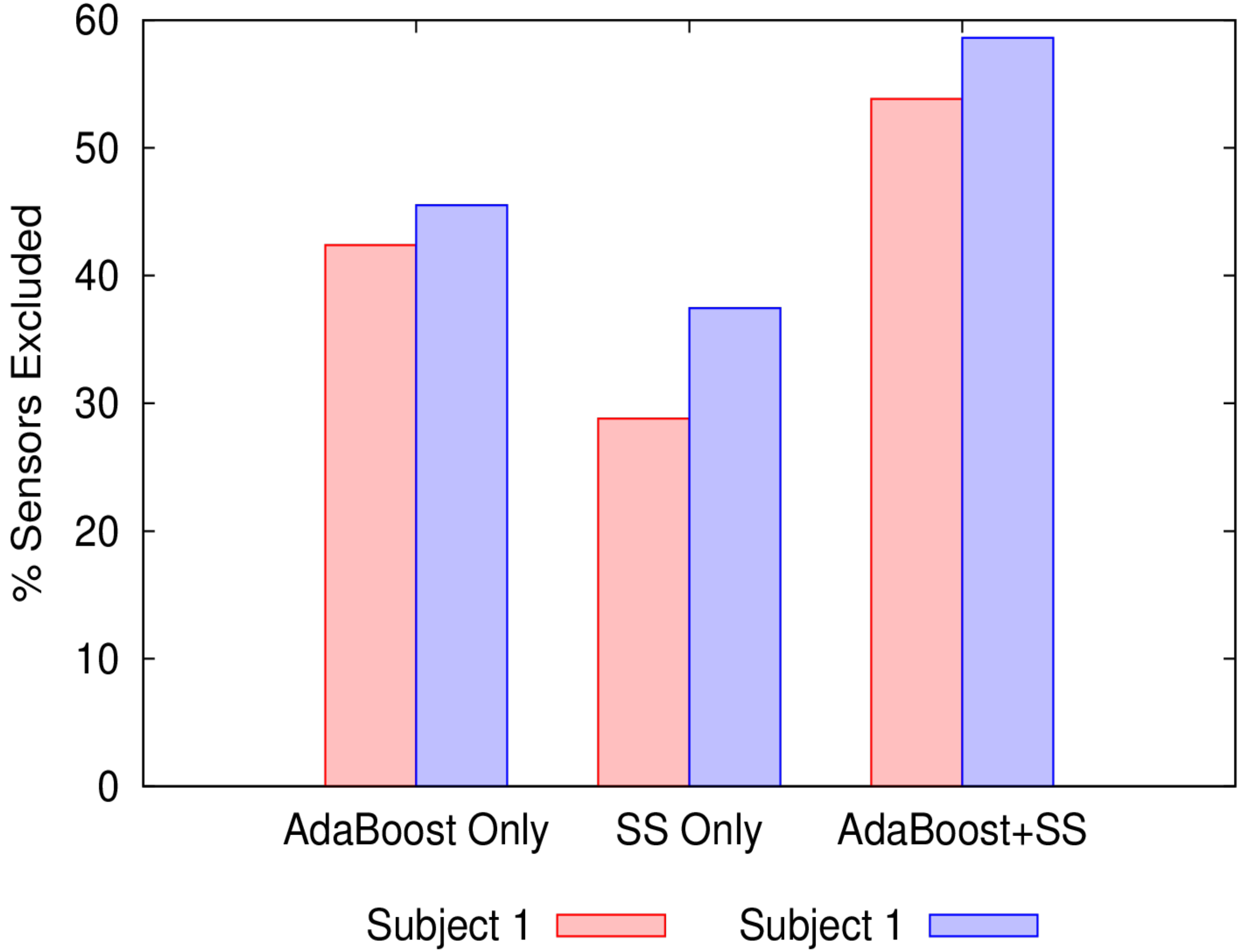
Classification Performance



Retraining Performance



Sensor Selection Performance



Application Performance

Mode	CPU	Memory	Power
Idle (No PBN)	<1%	4.30MB	360.59mW
Sampling (WiFi)	19%	8.16MB	517.74mW
Sampling (GPS)	21%	8.47MB	711.74mW
Sampling (Wifi) + Train	100%	9.48MB	601.02mW
Sampling (WiFi) + Classify	21%	9.45MB	513.57mW

Conclusion & Discussion

- PBN: Towards practical BSN daily activity recognition
- PBN provides:
  - User-friendly hardware and software
  - Strong classification performance
  - Retraining detection to reduce invasiveness
  - Identification of redundant resources
- Discussion
  - Extensive usability study?
  - Improving phone energy usage?
  - Reducing the number of on-body sensor nodes?
  - Turning on-body sensor nodes into wearable devices?
  - Sharing resources among multiple BSN?
  - Sharing resources with other sensors/devices in environment?