CSE 162 Mobile Computing

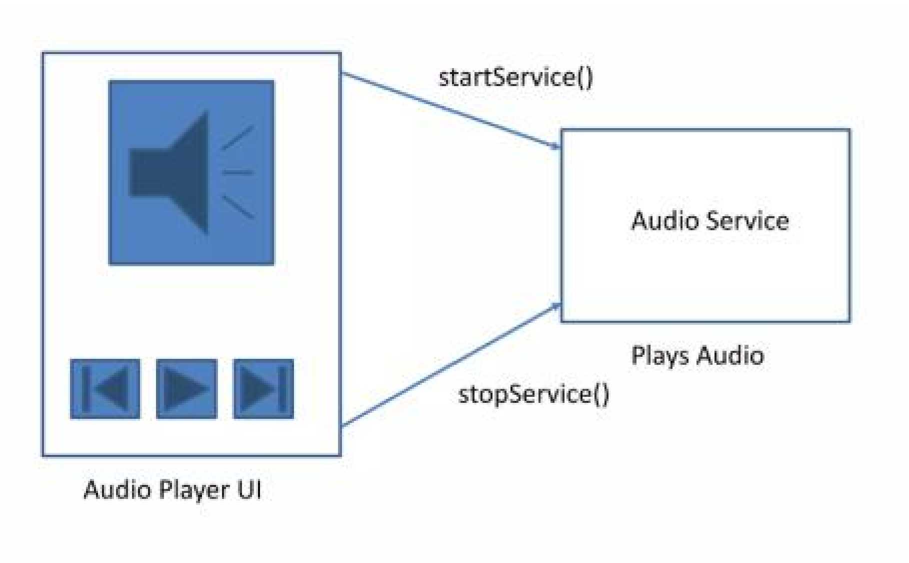
Mobile Activity Recognition

Hua Huang

Services

What is Android Service?

- Services are codes that run in the background
- They can be started and stopped
- Services doesn’t have UI



What a Service is NOT?

- There are some confusions:

- A Service is not a separate process.
  - The Service object itself does not imply it is running in its own process; unless otherwise specified, it runs in the same process as the application it is part of.

- A Service is not a thread.
  - It is not a means itself to do work off of the main thread (to avoid Application Not Responding errors).

Main Features of Service

- To tell the system about something it wants to be doing in the background (even when the user is not directly interacting with the application).

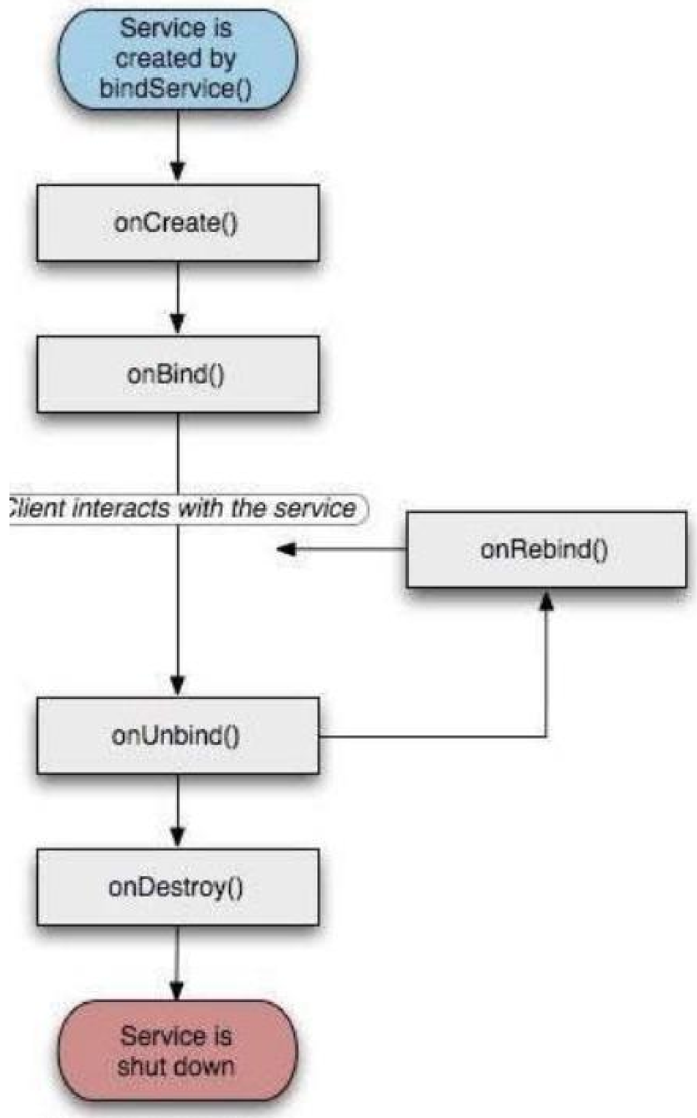
- To call Context.startService(), which asks the system to schedule work for the service, to be run until the service or someone else explicitly stops it.

When to use Services

- Activities are short-lived, can be shut down anytime (e.g when user presses back button)
- Services keep running in background
- Similar to Linux/Unix CRON job
- Example uses of services:
  - Periodically check/update device's GPS location
  - Check for updates to RSS feed
  - Independent of any activity, minimal interaction
- Typically an activity will control a service --start it, pause it, get data from it
- Services in an App are sub-class of Android's **Services** class

Bound Service

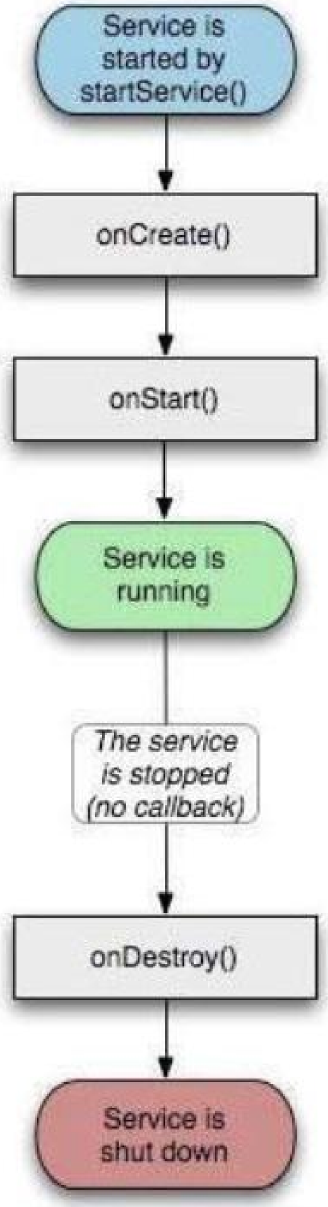
- An Android component may bind itself to a Service using bindservice ().
- A bound service would run as long as the other application components are bound to it.
- As soon as they unbind, the service destroys itself.



Bounded services

Unbound Service

- In this case, an application component starts the service, and it would continue to run in the background, even if the original component that initiated it is destroyed.
- For instance, when started, a service would continue to play music in the background indefinitely.


Service is started by startService()
onCreate()
onStart()
Service is running
The service is stopped (no callback)
onDestroy()
Service is shut down

UN Bounded Service

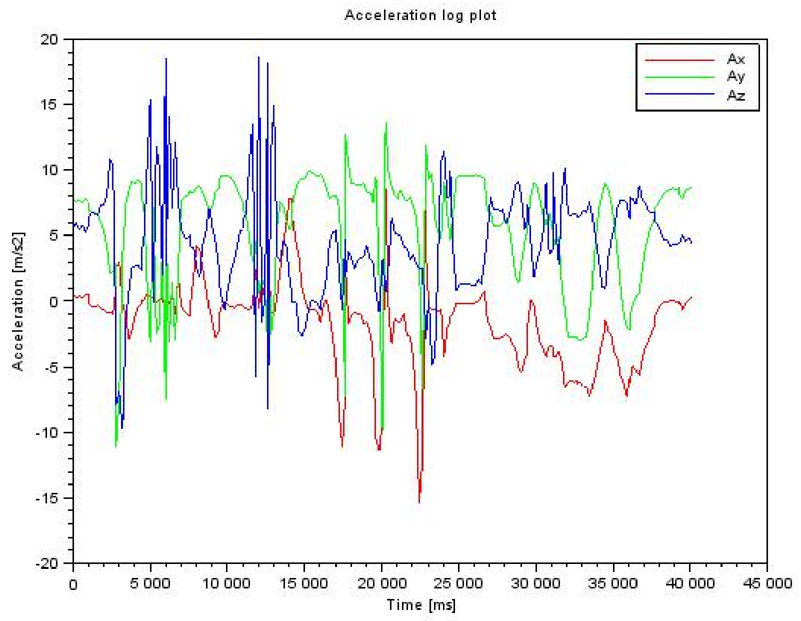
Android Platform Services

- Android Services can either be on:
  - On smartphone or Android device (local)
  - Remote, on Google server/cloud
- Android platform local services examples (on smartphone):
  - **LocationManager:** location-based services.
  - **ClipboardManager:**access to device’s clipboard, cut-and-paste content
  - **DownloadManager:** manages HTTP downloads in background
  - **FragmentManager:**manages the fragments of an activity.
  - **AudioManager:**provides access to audio and ringer controls.

An Introduction to Human Activity Recognition

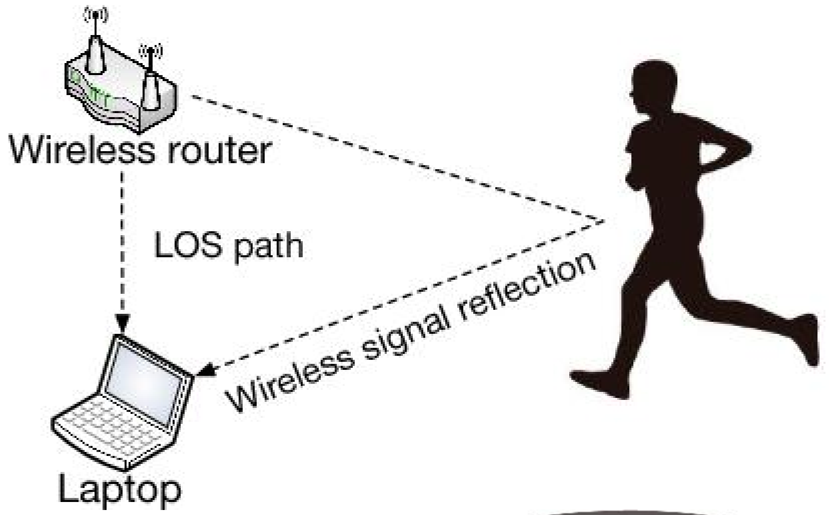
Activity Recognition

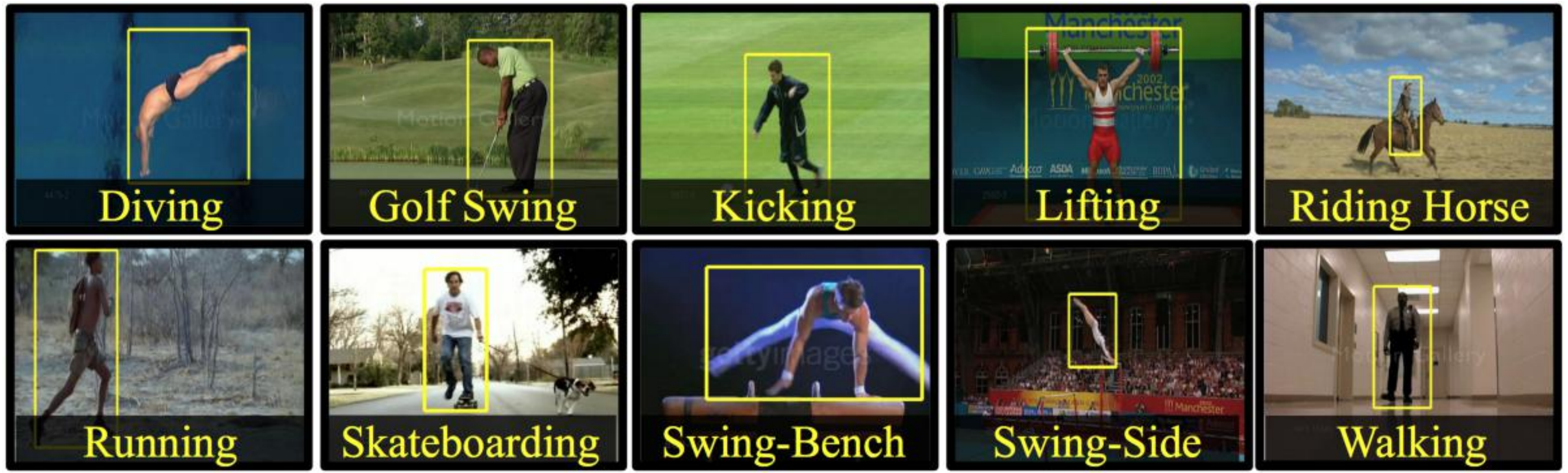
- Goal: Want our app to detect what activity the user is doing?
- Classification task: which of these 6 activities is user doing?
  - Walking,
  - Jogging,
  - Ascending stairs,
  - Descending stairs,
  - Sitting,
  - Standing
- Typically, use machine learning classifers to classify user's accelerometer signals



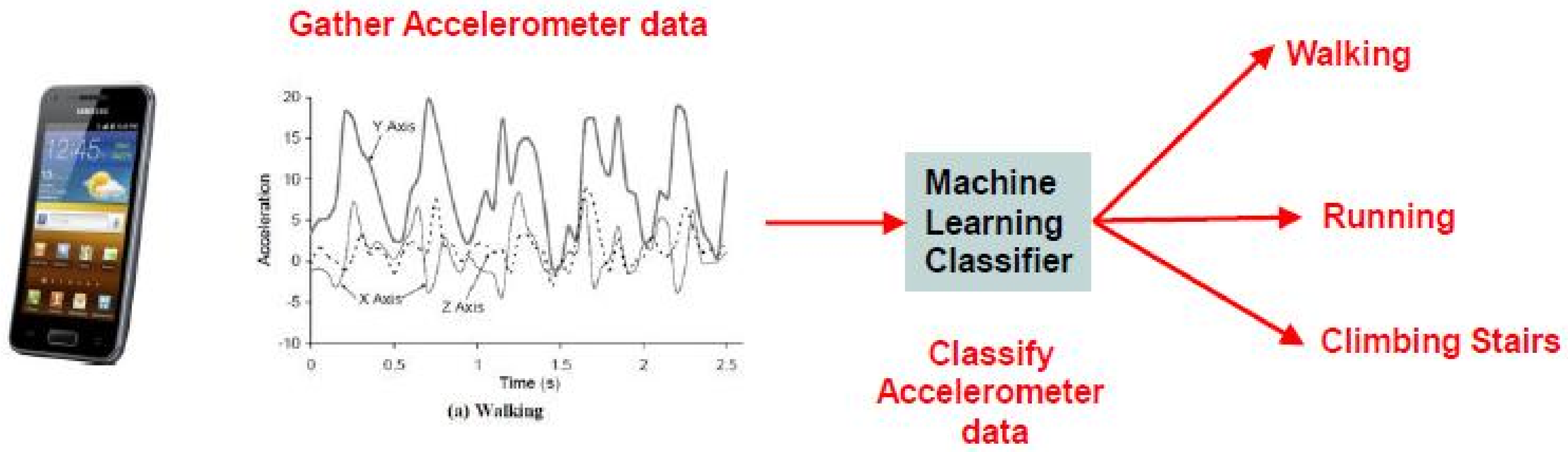
Activity Recognition Systems

- Computer vision based
- Environmental sensor based
- Mobile sensor based

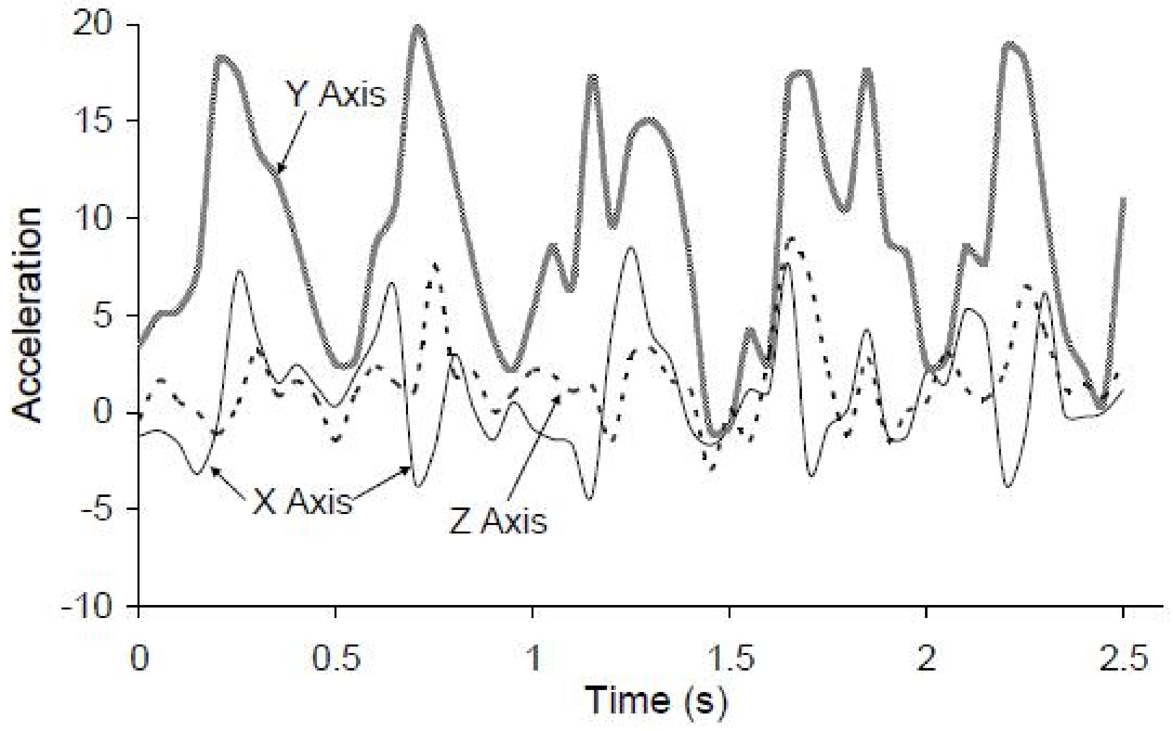




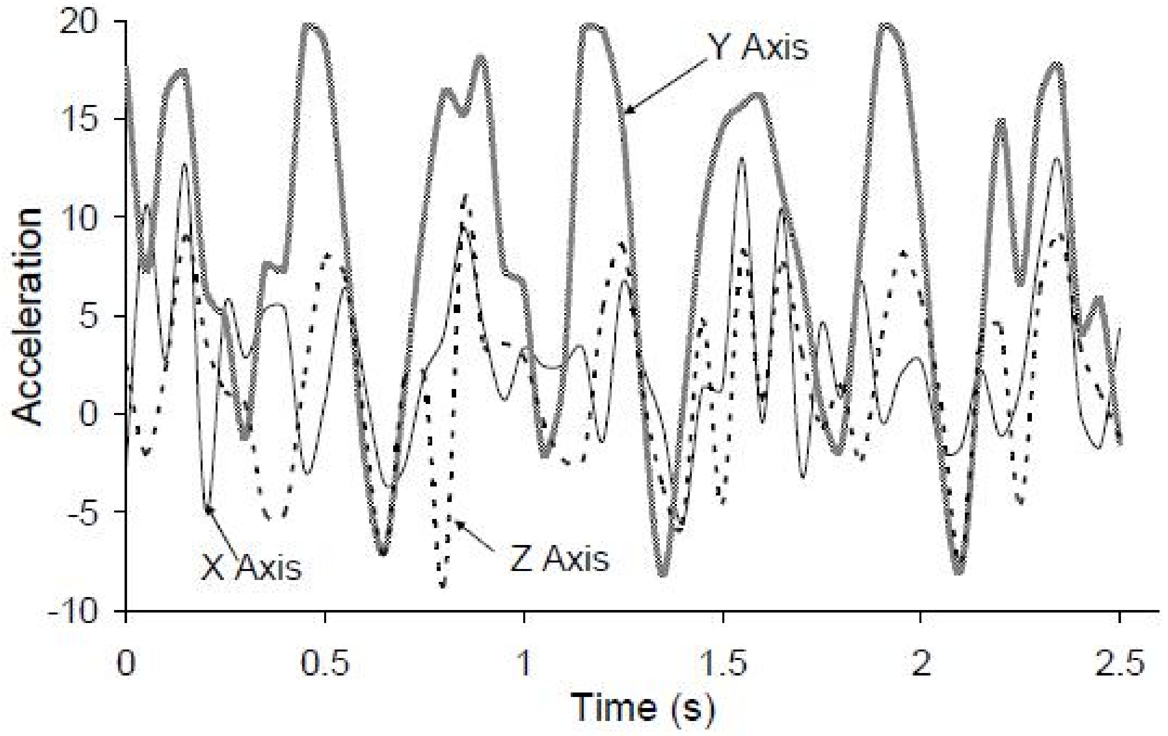
Mobile Activity Recognition Architecture


Gather Accelerometer data
Y Axis
X Axis
Z Axis
Acceleration
Time (s)
(a) Walking
Machine Learning Classifier
Classify Accelerometer data
Walking
Running
Climbing Stairs

Sample Accelerometer during activities

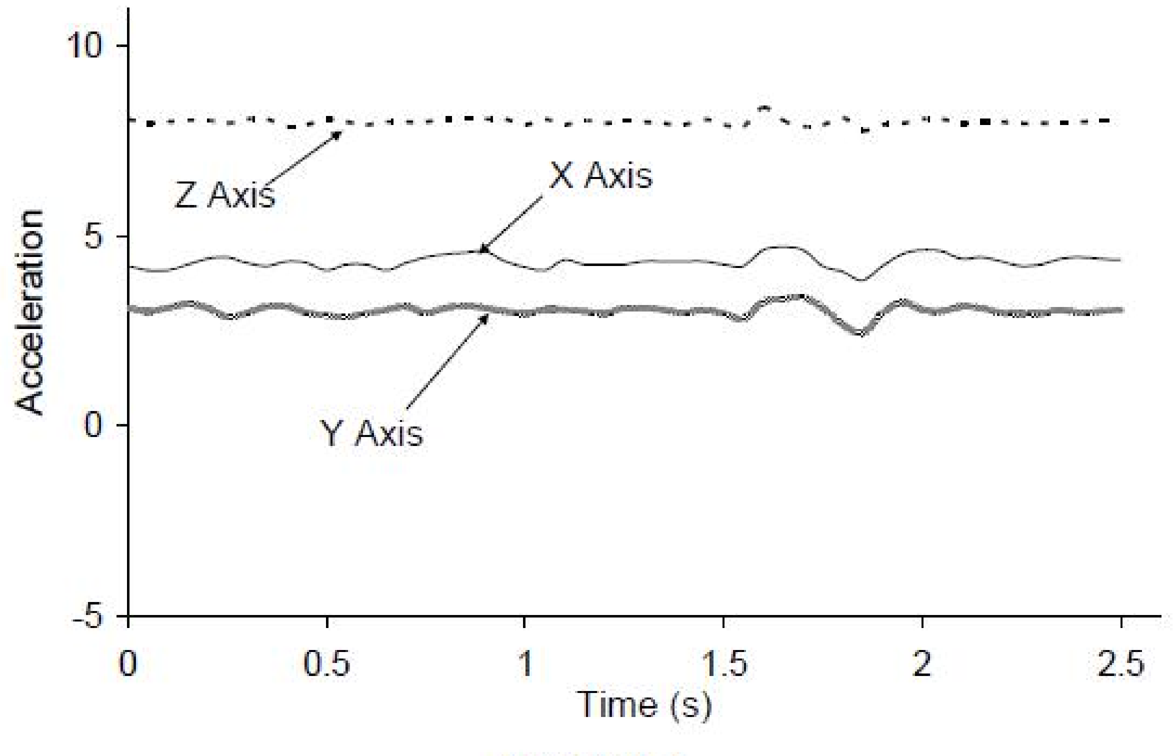


(a) Walking

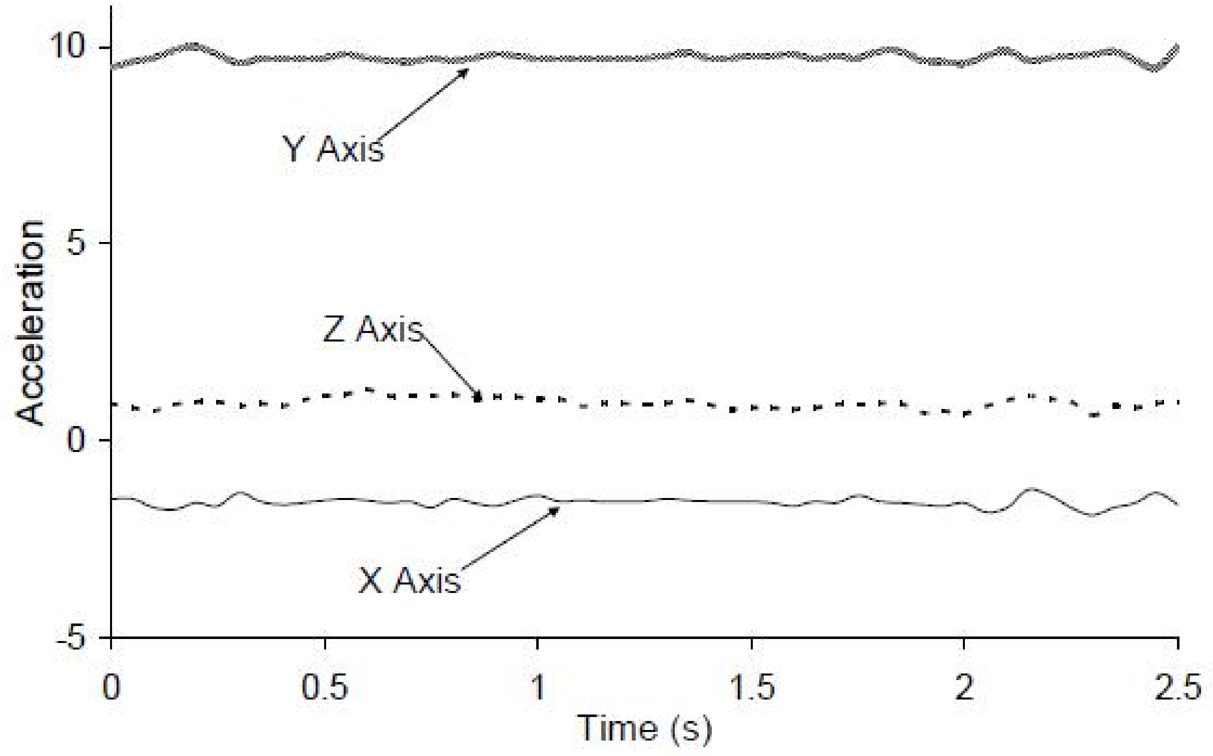


(b) Jogging

Sample Accelerometer during activities



(e) Sitting

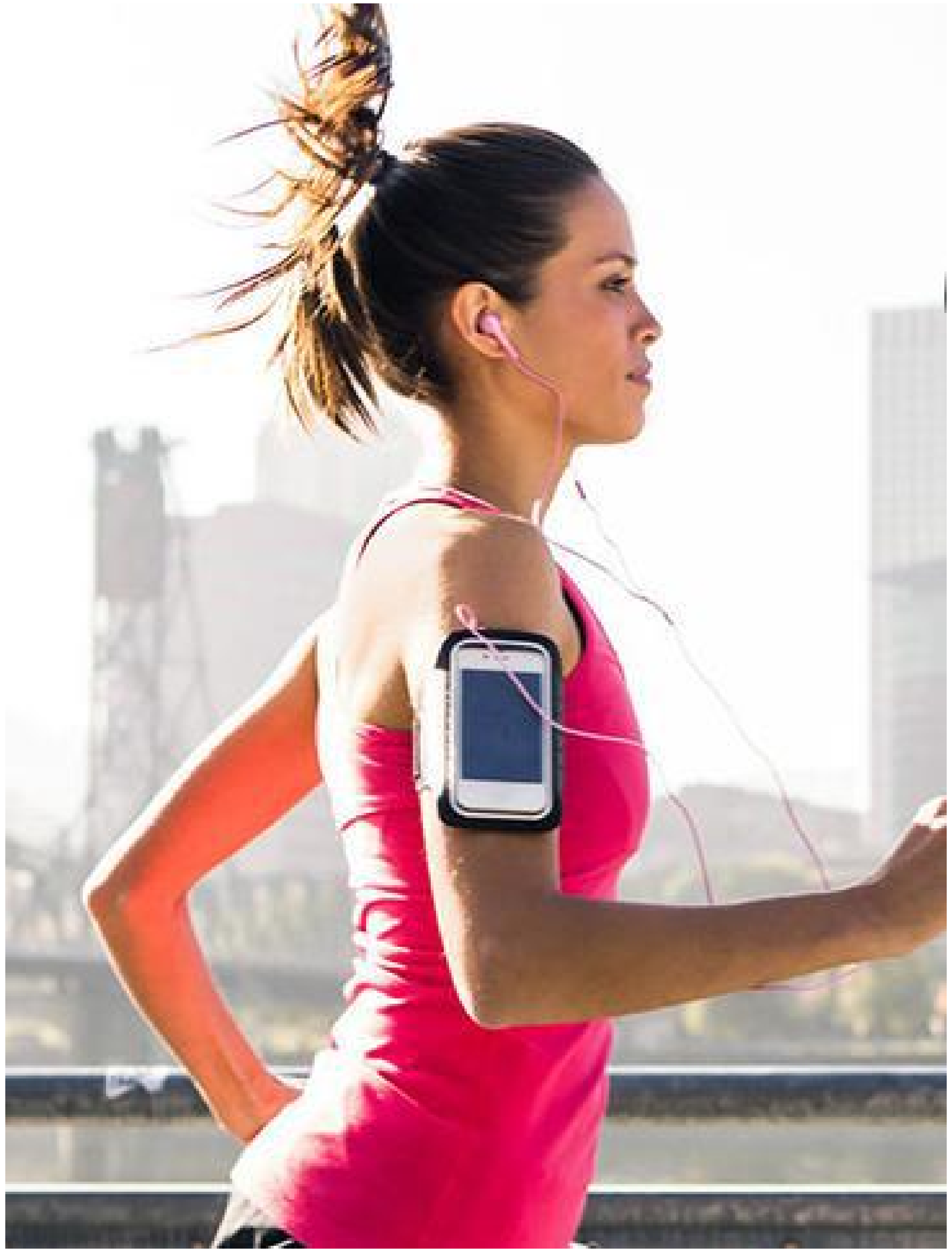


(f) Standing

Activity Recognition Applications

Fitness Tracking

- **Initially:**
  - Physical activity type,
  - Distance travelled,
  - Calories burned
- **Newer features:**
  - Stairs climbed,
  - Physical activity (duration + intensity)
  - Activity type logging + context
  - Sleep tracking
  - Activity history



Health Monitoring

- Make clinical monitoring pervasive, continuous, real world!!
  - Gather context information (e.g. what makes condition worse/better?)
  - E.g. timed up and go test
- Show patient contexts that worsen condition => Change behavior
  - E.g. walking in narrow hallways worsens gait freeze

Gait Freezing

Question: What data would you need to build PD gait classifier? From what types of subjects?

Fall Detection

- A leading cause of death for seniors
- Smartphone/watch, wearable detects senior who has fallen, alert family
  - Text message, email, call relative

Context-aware Behavior

- Study found that messages delivered when transitioning between activities better received
  - In-meeting? => Phone switches to silent mode
  - Exercising? => Play song from playlist, use larger font sizes for text
  - Arrived at work? => download email
- Adaptive Systems to Improve User Experience:
  - Walking, running, riding bike? => Turn off Bluetooth, WiFi (save energy)
  - an increase battery life up to 5x

Smart Home

- Smart Thermostat
  - Determines if the users are at home
  - Recognize where the users are and adjust temperature accordingly

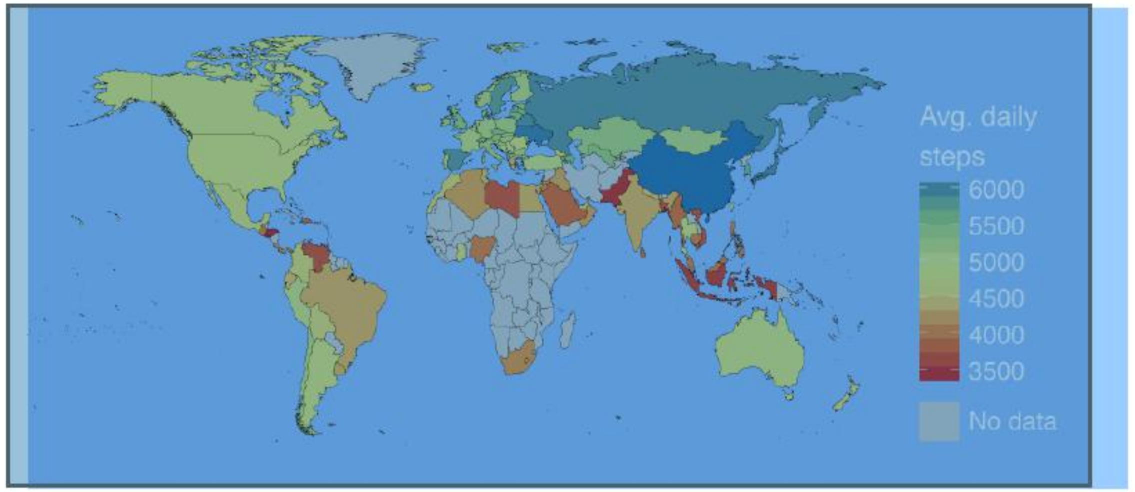
- Smart TV
  - Turns on TV when the user is at the couch

Targeted Advertisements

- User runs a lot => Get exercise clothing ads
- Goes to pizza places often + sits there => Get pizza ads

Research Platforms for Data Collection

- E.g. public health officials want to know how much time various people (e.g. students) spend sleeping, walking, exercising, etc
- Mobile AR: inexpensive, automated data collection
- E.g. Stanford Inequality project: Analyzed physical activity of 700k users in 111 countries using smartphone AR data
  - http://activityinequality.stanford.edu/

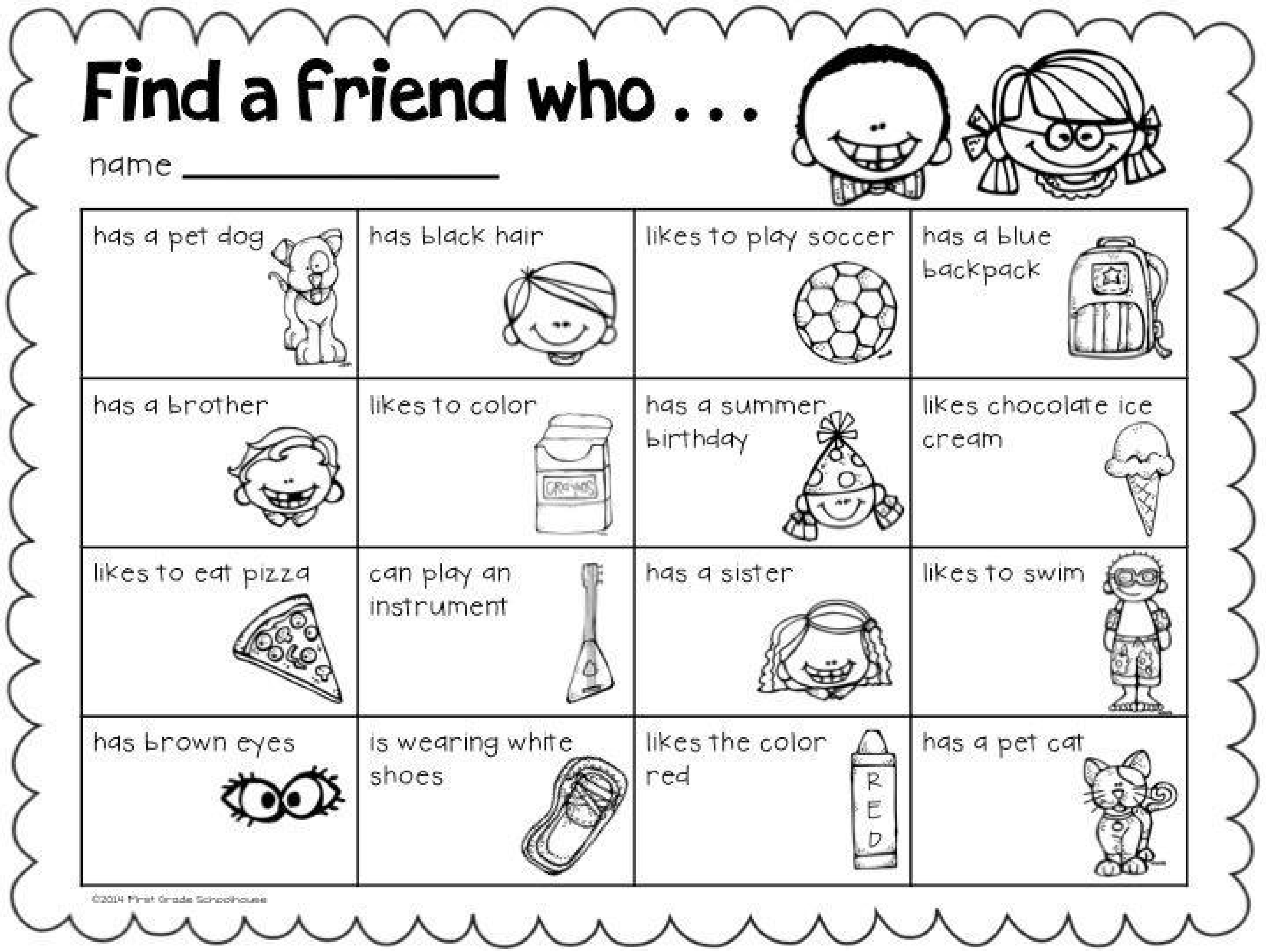


Track, manage staff on-demand

- E.g. at hospital, determine “availability of nurses”, assign them to new jobs/patients/surgeries/cases

Activity-Based Social Networking

- Automatically connect users who do same activities + live close together



Find a friend who . . .

name ___________________

has a pet dog	has black hair	likes to play soccer	has a blue backpack
has a brother	likes to color	has a summer birthday	likes chocolate ice cream
likes to eat pizza	can play an instrument	has a sister	likes to swim
has brown eyes	is wearing white shoes	likes the color red	has a pet cat

Activity Recognition using Google API

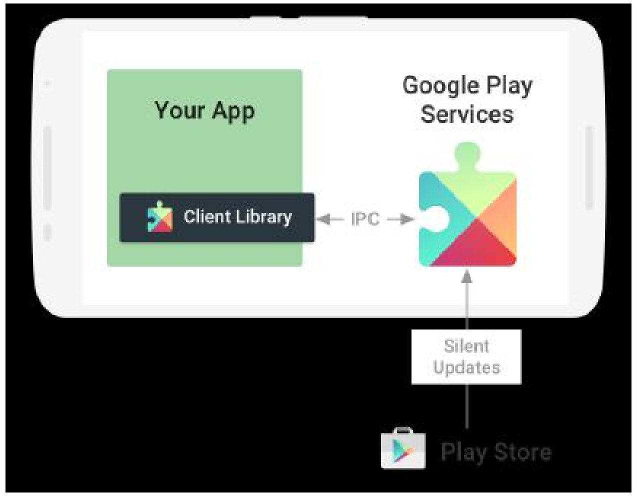
- We can adapt the mobile behaviors based on the user’s behaviors
  - E.g., If the user is driving, don’t send notifications

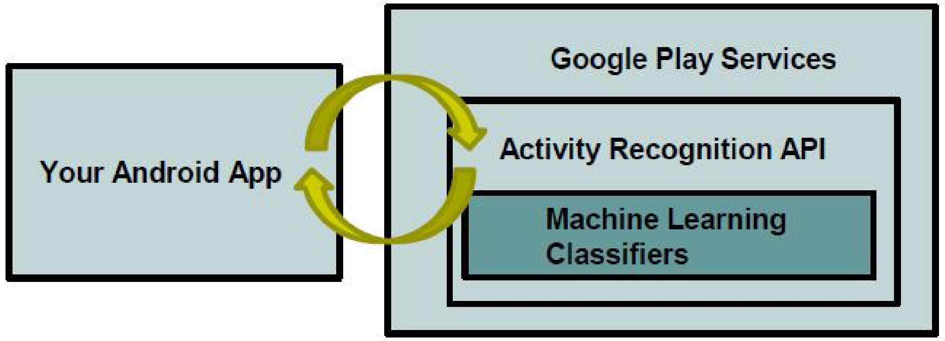
Google Activity Recognition API

- API to detect smartphone user’s current activity
- Programmable, can be used by your Android app
- Currently detects 8 states:
  - In vehicle
  - On Bicycle
  - On Foot
  - Running
  - Walking
  - Still
  - Tilting
  - Unknown

Google Activity Recognition API

- Deployed as part of Google Play Services





Activity Recognition API

- Understanding what users are doing in the physical world allows your app to be smarter about how to interact with them.
  - For example, an app can start tracking a user's heartbeat when she starts running,
  - another app can switch to *car mode* when it detects that the user has started driving.
- The Activity Recognition API is built on top of the sensors available in a device.
  - Device sensors provide insights into what users are currently doing.

- The Activity Recognition API automatically detects activities by periodically reading short bursts of sensor data and processing them using machine learning models.
- To optimize resources, the API may stop activity reporting if the device has been still for a while, and uses low power sensors to resume reporting when it detects movement.

Features of Google API

- Receive information about activities using minimal resources
  - Get notified when your user starts or ends a particular activity
  - Perform an action when your app receives activity information
  - Receive detected activities that include a confidence grade

Get notified when your user starts or ends a particular activity

- For example, a mileage tracking app could start tracking miles when a user starts driving, or a messaging app could mute all conversations until the user stops driving.
- The Activity Recognition Transition API detects changes in the user's activity.
- The app subscribes to a transition in activities
  - The API notifies your app only when needed.
- No need to implement complex heuristics to detect when an activity starts or ends.

Perform an action when your app receives activity information

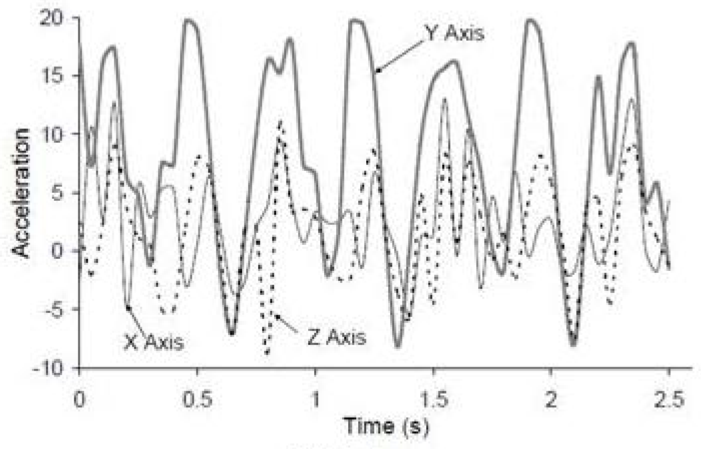
- The Activity Recognition API delivers its results to a callback, which is usually implemented as an IntentService
- The results are delivered at intervals that you specify,
  - or your app can use the results requested by other clients without consuming additional power itself.
- You can tell the API how to deliver results by using a PendingIntent
  - It removes the need to have a service constantly running in the background for activity detection purposes.
  - The app receives the corresponding Intents from the API, extracts the detected activities, and decides if it should take an action.

Receive detected activities that include a confidence grade

- The Activity Recognition API does the heavy lifting by processing the signals from the device to identify the current activities.
- Your app receives a list of detected activities, each of which includes confidence and type properties.
  - The confidence property indicates the likelihood that the user is performing the activity represented in the result.
  - The type property represents the detected activity of the device relative to entities in the physical world, for example, the device is on a bicycle or the device is on a user who is running.

Activity Recognition System Design

Activity recognition on Android



(b) Jogging

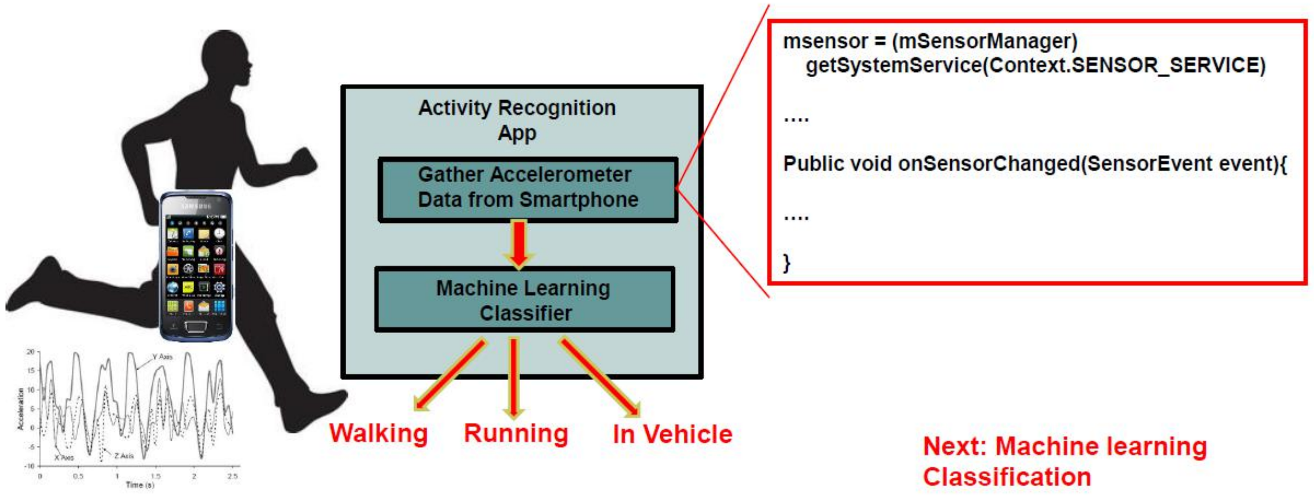
New accelerometer
Sample in real time

Classifier in
Android app

Activity
(e.g. Jogging)

Activity Recognition (AR) Android App

- As user performs an activity, AR app on user's smartphone
  - Gathers accelerometer data
  - Uses **machine learning classifier** to determine what activity (running, jumping, etc) accelerometer pattern corresponds to
- **Classifier:** Machine learning algorithm that guesses what activity **class**(or type) accelerometer sample corresponds to

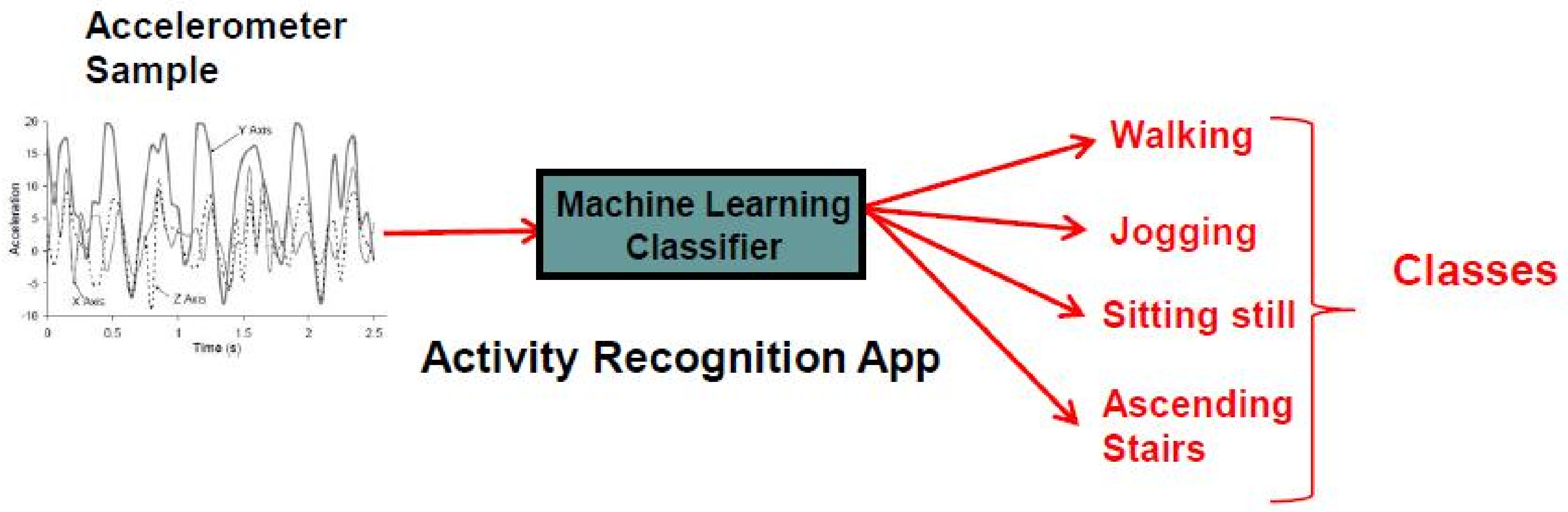

Activity Recognition App
Gather Accelerometer Data from Smartphone
Machine Learning Classifier
Walking
Running
In Vehicle

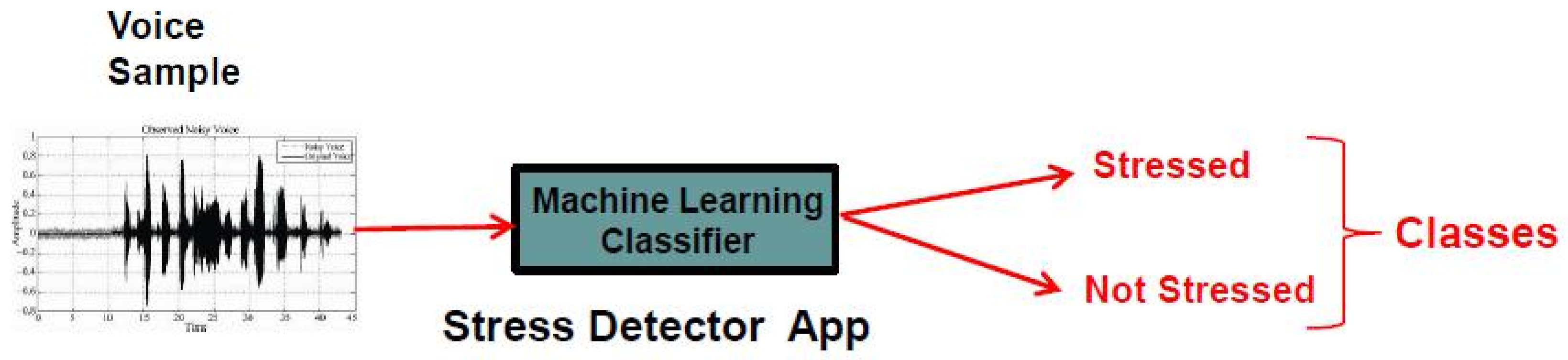
```
msensor = (mSensorManager)
  getSystemService(Context.SENSOR_SERVICE)

....

Public void onSensorChanged(SensorEvent event){

....

}
```

Next: Machine learning Classification

Classification

- Classification? determine which **class** a sample (e.g. snippet of accelerometer data) belongs to. Examples:





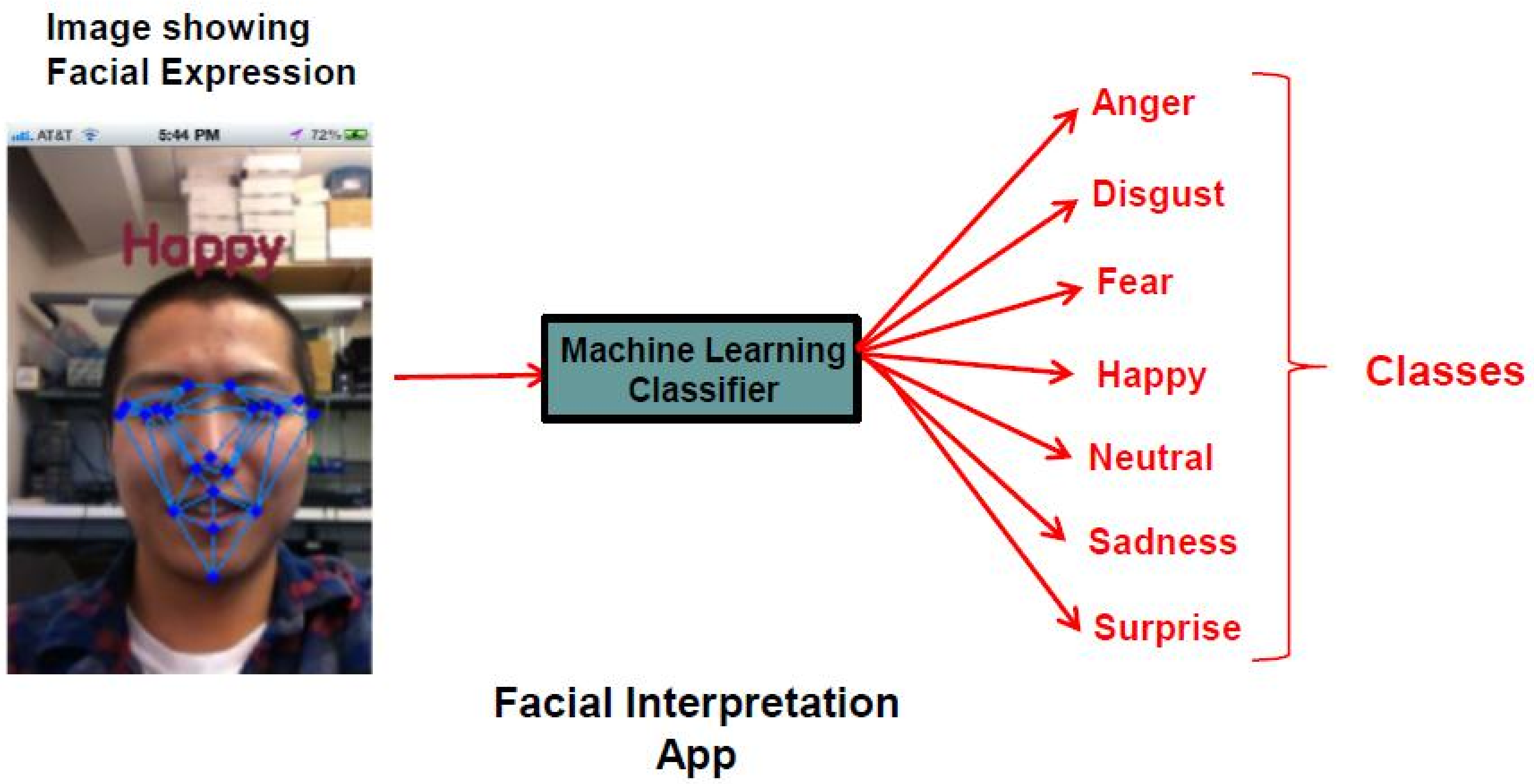
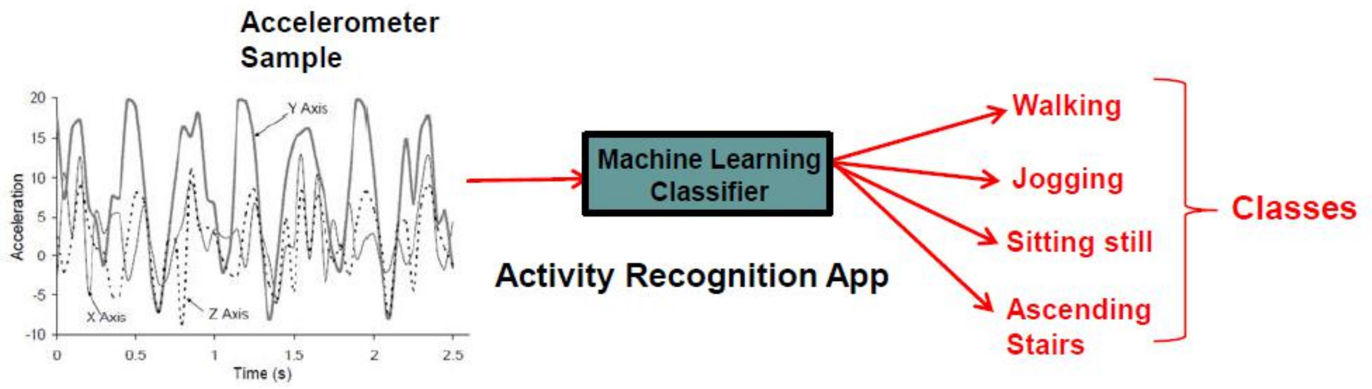


Image showing
Facial Expression
AT&T
5:44 PM
72%
Happy
Machine Learning Classifier
Anger
Disgust
Fear
Happy
Neutral
Sadness
Surprise
Classes
Facial Interpretation App

Classifier

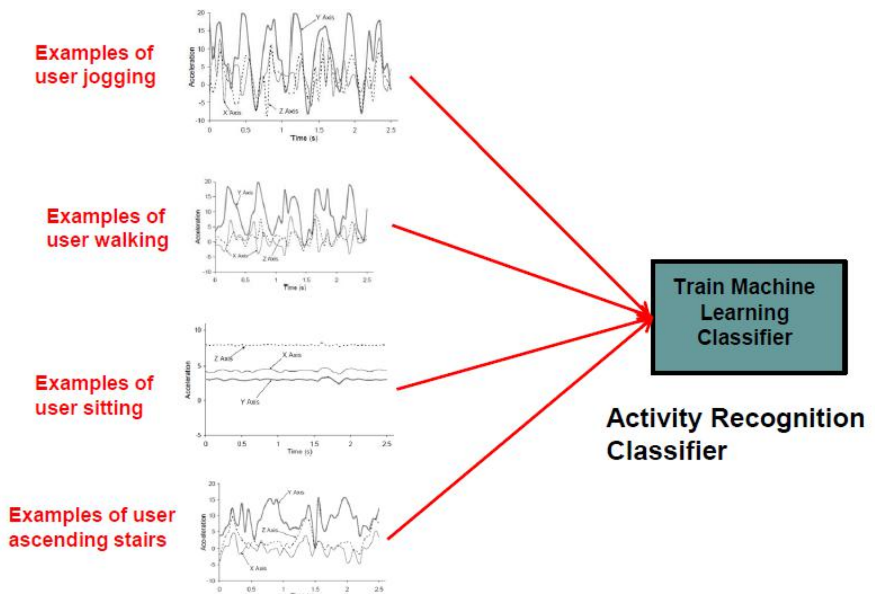
- Analyzes new sample, guesses corresponding class
- Intuitively, can think of classifier as set of rules for classification
- Example rules for classifying accelerometer signal in Activity Recognition

```
If ((Accelerometer peak value > 12 m/s)
      and (Accelerometer average value < 6 m/s)){
             Activity = "Jogging";
}
```



Training a Classifier

- Created using example-based approach (called training)
- **Training a classifier:** Given examples of each class => generate rules to categorize new samples
- **E.g:** Analyze 30+ Examples (from 30 subjects) of accelerometer signal for each activity type (walking, jogging, sitting, ascending stairs) => generate rules (classifier) to classify future activities


Examples of user jogging
Examples of user walking
Examples of user sitting
Examples of user ascending stairs
Train Machine Learning Classifier
Activity Recognition Classifier

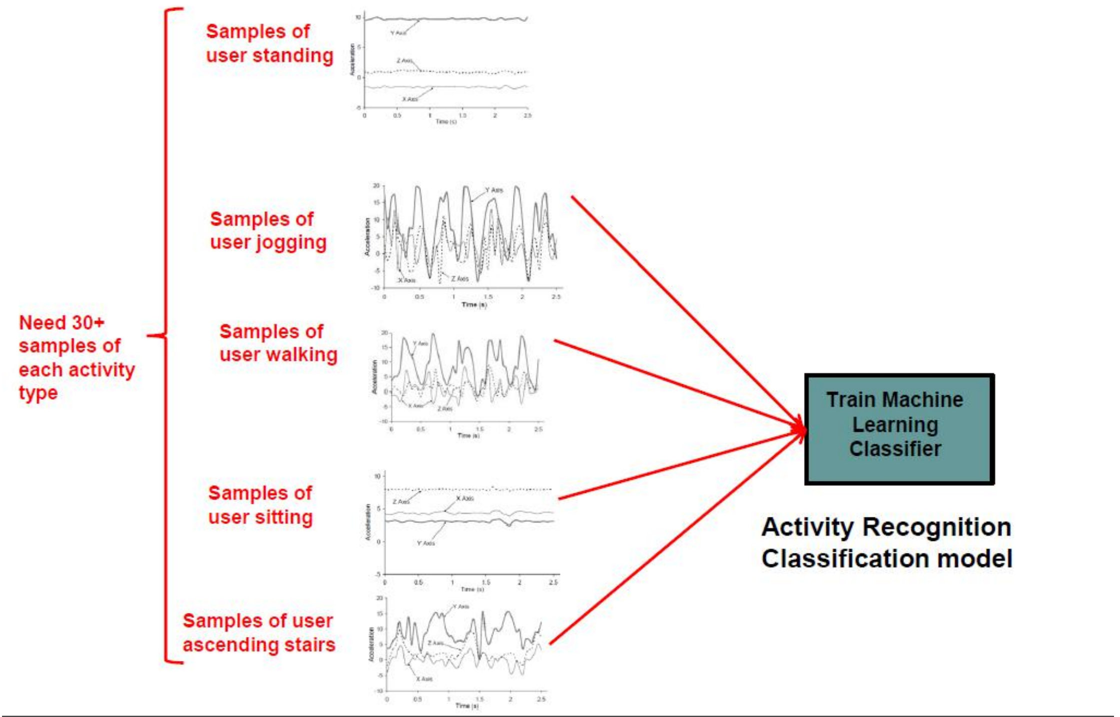
Training a Classifier: Steps

Steps for Training a Classifier

1. Gather data samples + label them
2. Import accelerometer samples into classification library (e.g. python, MATLAB)
3. Pre-processing (segmentation, smoothing, etc)
4. Extract features
5. Train classifier
6. Export classification model as JAR file
7. Import into Android app

Step 1: Gather Sample data + Label them

- Need many samples of accelerometer data corresponding to each activity type (jogging, walking, sitting, ascending stairs, etc)


Need 30+ samples of each activity type
Samples of user standing
Samples of user jogging
Samples of user walking
Samples of user sitting
Samples of user ascending stairs
Train Machine Learning Classifier
Activity Recognition Classification model

Step 1: Gather Sample data + Label them

- Conduct a study to gather sample accelerometer data for each activity class
  - Recruit 30+ subjects
  - Run program that gathers accelerometer sensor data on subject's phone
  - Each subject:
    - Perform each activity (walking, jogging, sitting, etc)
    - Collect accelerometer data while they perform each activity (walking, jogging, sitting, etc)
  - Label data. i.e. tag each accelerometer sample with the corresponding activity
- Now have 30+ examples of each activity

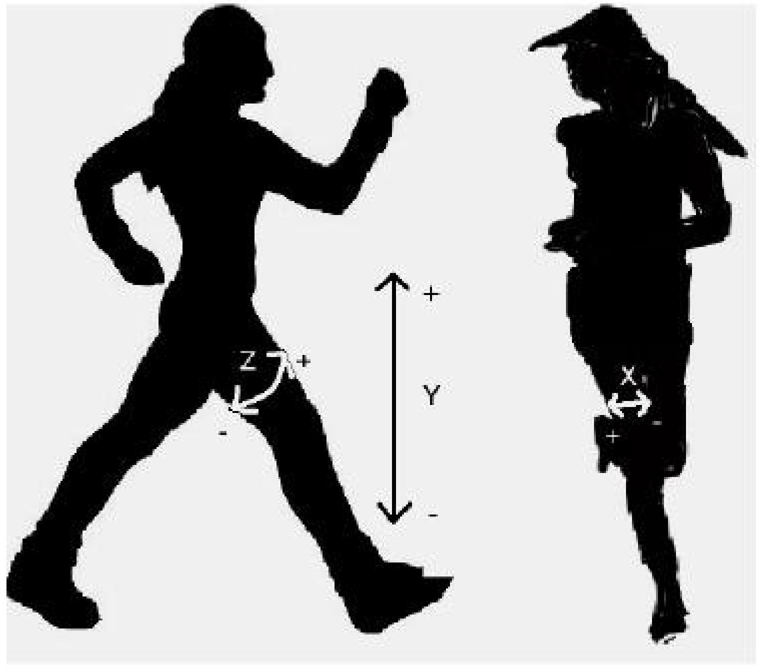
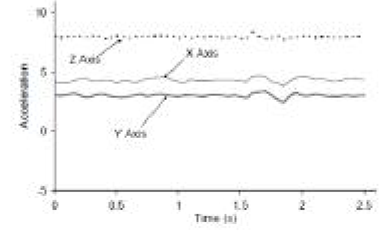
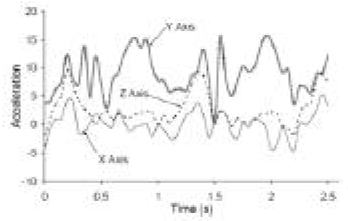


Figure 1: Axes of Motion Relative to User

30+ Samples of user sitting

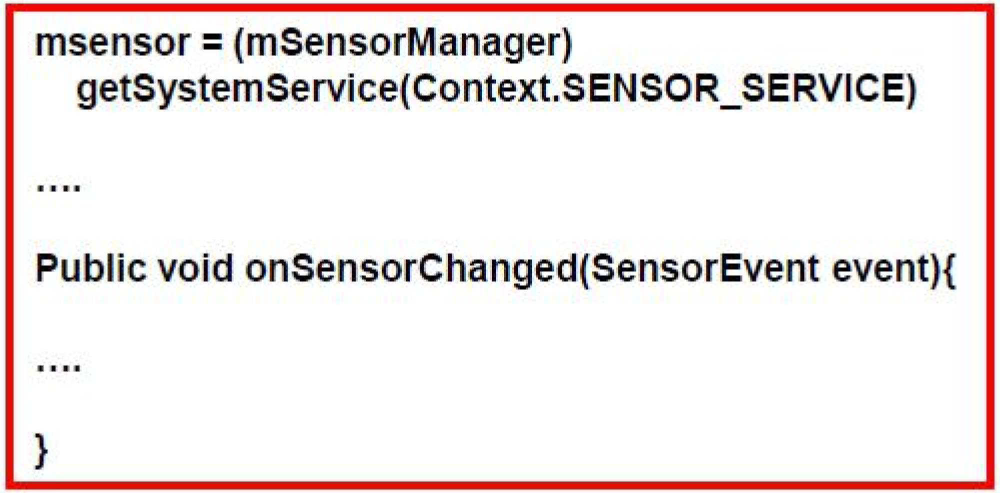


30+ Samples of user ascending stairs



Step 1: Gather Sample data + Label themProgram to Gather Accelerometer Data

- **Option 1:** Can write sensor program app that gathers accelerometer data while user is doing each of 6 activities (1 at a time)



```
msensor = (mSensorManager)
   getSystemService(Context.SENSOR_SERVICE)

....

Public void onSensorChanged(SensorEvent event){

....

}
```

Step 1: Gather Sample data + Label themProgram to Gather Accelerometer Data

- **Option 2:** Use 3rdparty app to gather accelerometer
  - 2 popular ones: **Funf** and **AndroSensor**
  - Just download app,
  - Select sensors to log (e.g. accelerometer)
  - Continuously gathers sensor data in background

Step 2: Import accelerometer samples into classification library (e.g. Python, MATLAB)

LABELS

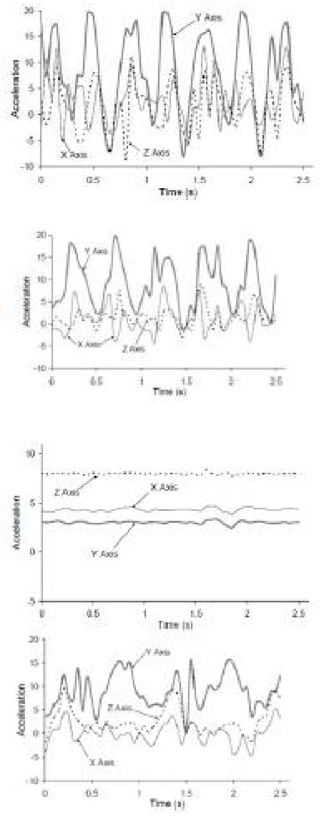
ACCELEROMETER DATA

Jogging

Walking

Sitting

Ascending stairs



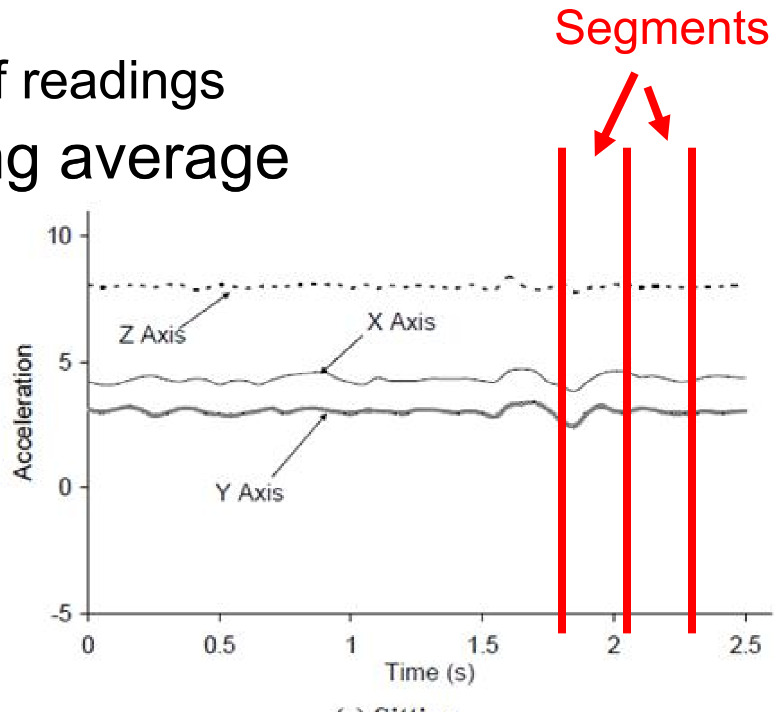
Weka, Matlab

Classifiers

Classifier is trained offline

Step 3: Pre-processing (segmentation, smoothing, etc)Segment Data (Windows)

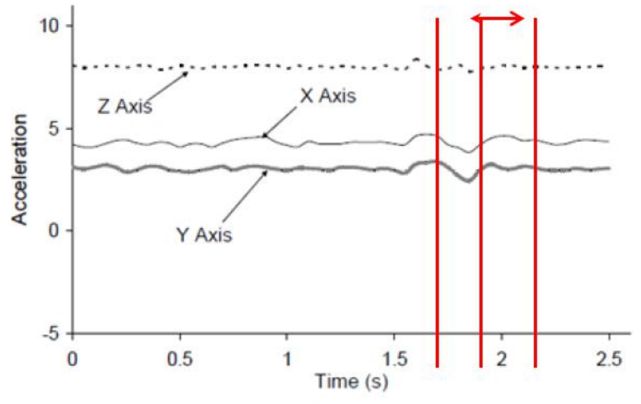
- Pre-processing data may include segmentation, smoothing, etc
  - **Segment:** Divide data into smaller chunks. E.g. divide 60 seconds of raw time-series data into 5 second chunks
    - Note: 5 seconds of accelerometer data could be 100s of readings
  - **Smoothing:** Replace groups of values with moving average



(e) Sitting

Step 4: Compute (Extract) Features

- For each 5-second segment (batch of accelerometer values) compute features
- **Features:** Formulas computed to quantify attributes of accelerometer data, captures accelerometer characteristics
- **Examples:** min-max of values within each segment, largest magnitude, standard deviation

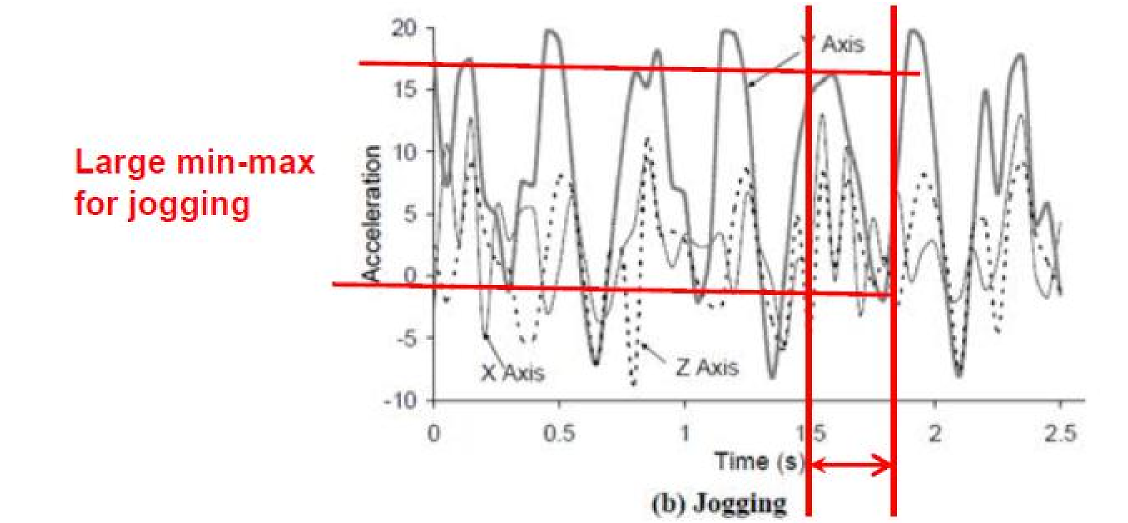


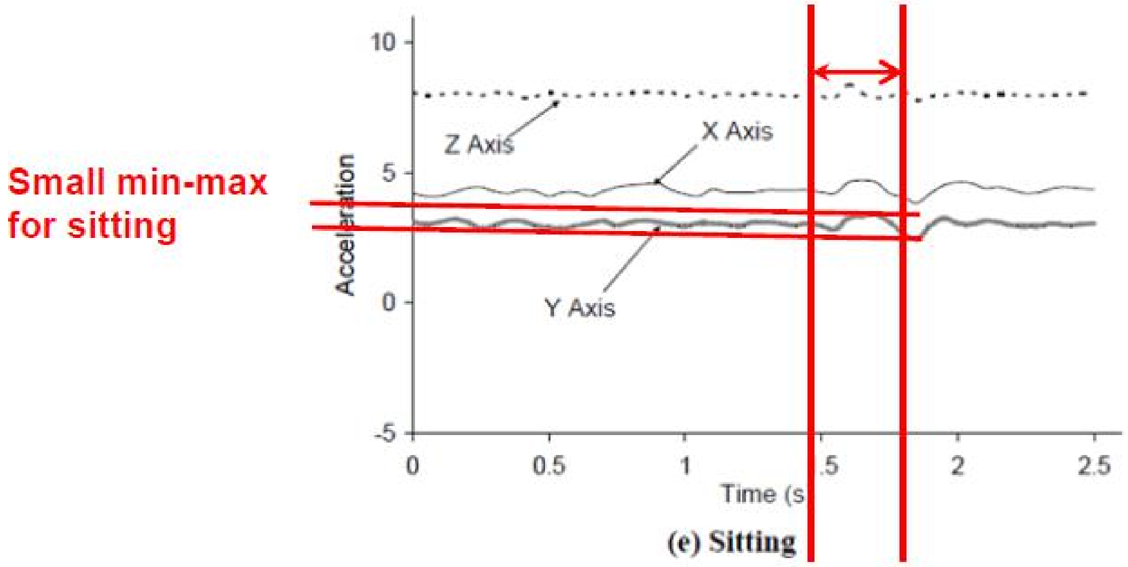
(e) Sitting

Step 4: Compute (Extract) Features

- **Important:** Ideally, values of features different for, distinguish each activity type (class)

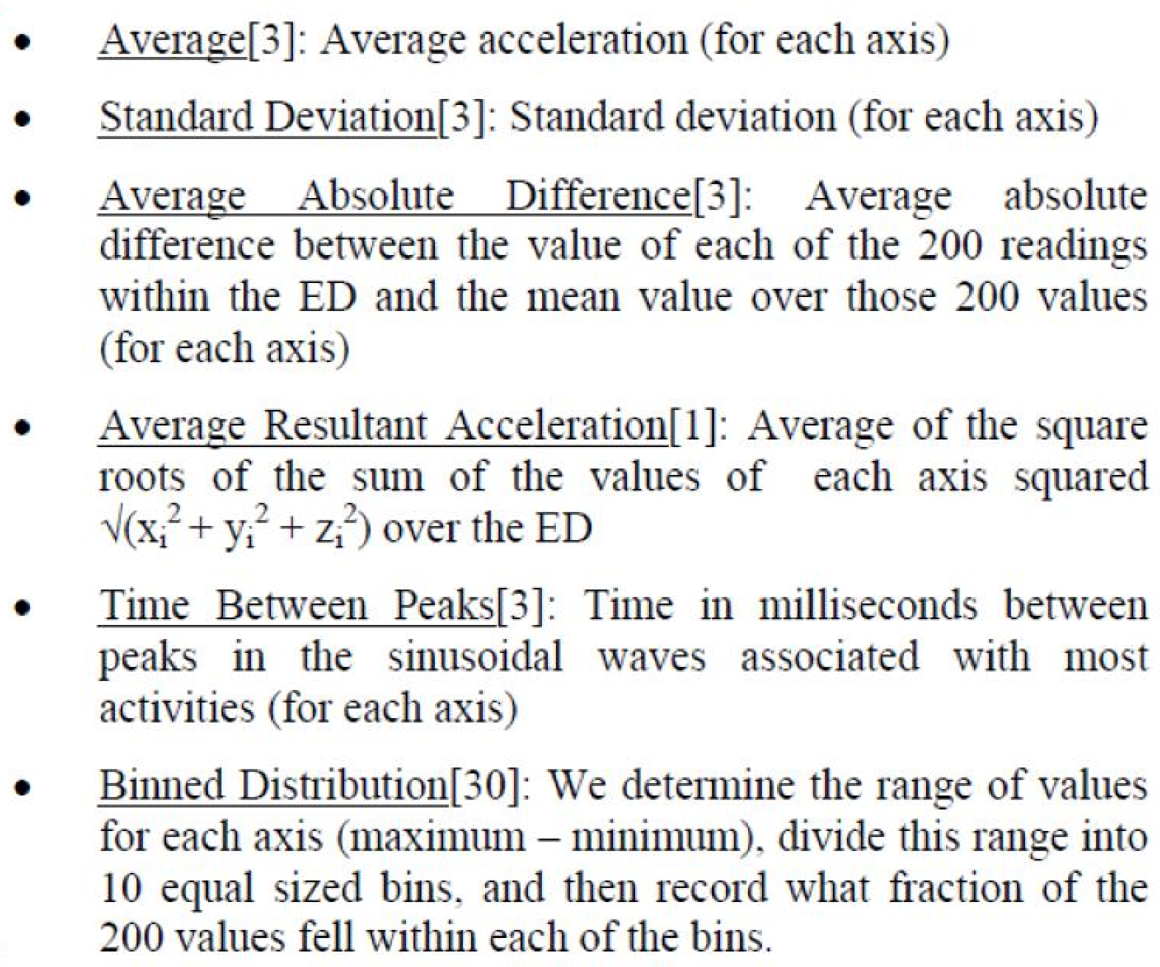
- **E.g:**Min-max range feature





Step 4: Compute (Extract) Features

Calculate many different features



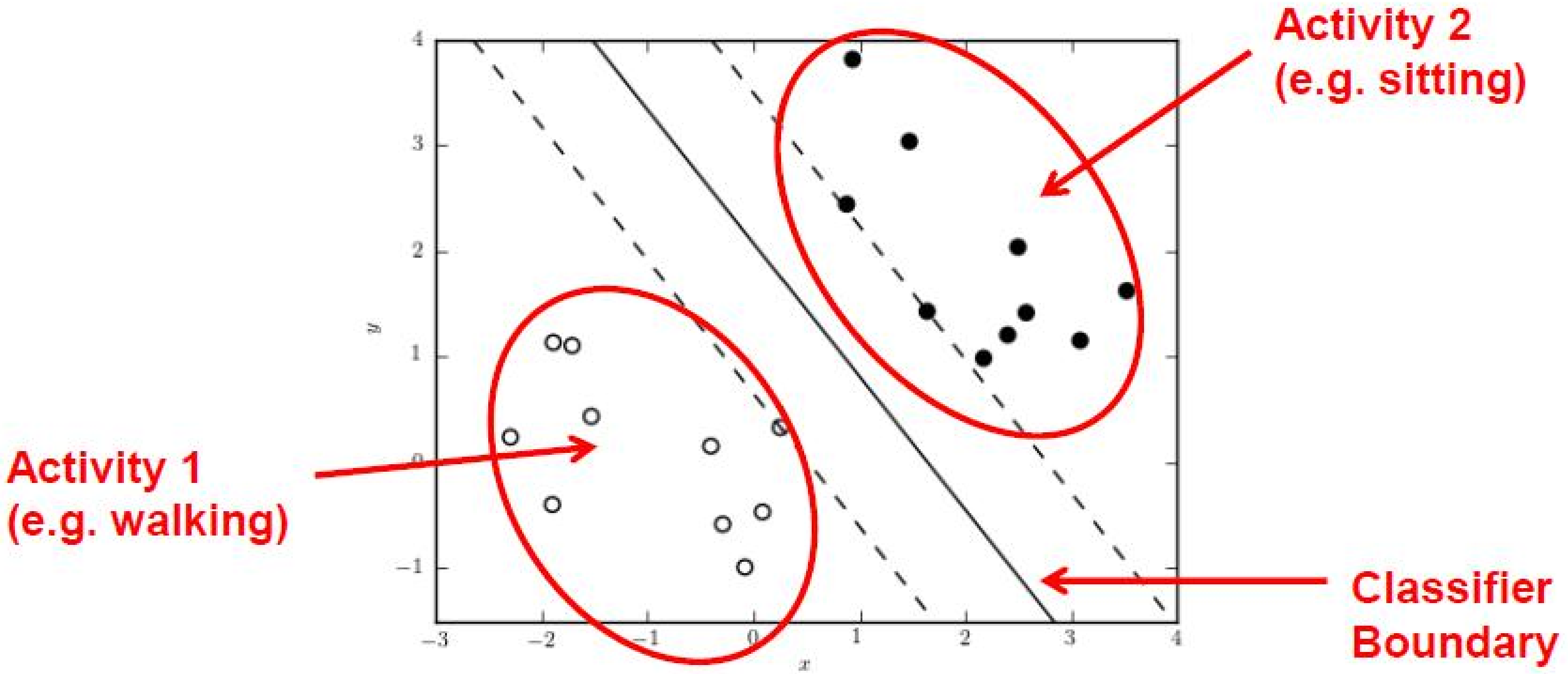
- Average[3]: Average acceleration (for each axis)
- Standard Deviation[3]: Standard deviation (for each axis)
- Average Absolute Difference[3]: Average absolute difference between the value of each of the 200 readings within the ED and the mean value over those 200 values (for each axis)
- Average Resultant Acceleration[1]: Average of the square roots of the sum of the values of each axis squared $\sqrt{(x_i^2 + y_i^2 + z_i^2)}$ over the ED
- Time Between Peaks[3]: Time in milliseconds between peaks in the sinusoidal waves associated with most activities (for each axis)
- Binned Distribution[30]: We determine the range of values for each axis (maximum – minimum), divide this range into 10 equal sized bins, and then record what fraction of the 200 values fell within each of the bins.

Step 5: Train classifier

- Features are just numbers (e.g. values of features for different subjects, activities)
- Different values for different activities
- **Training classifier:**figures out feature values corresponding to each activity
- Python, MATLAB already programmed with different classification algorithms (SVM, Naïve Bayes, Random Forest, J48, logistic regression, SMO, etc)
- Try different ones, compare accuracy

Step 5: Train classifier

- SVM example



Step 6: Export Classification model as JAR file
Step 7: Import into Android app

- Export classification model (most accurate classifier type + data threshold values) as Java JAR file
- Import JAR file into Android app
- In app write Android code to
- Gather accelerometer data, segment, extract feature, classify using classifier in JAR file
- Classifies new accelerometer patterns while user is performing activity => Guess (infer) what activity