Making Apps Intelligent (Sensors Inference & Machine Learning)
My Goals in this Section

- If you know machine learning
  - Set off light bulb
  - Projects involving ML?
- If you don’t know machine learning
  - Get general idea, how it’s used
- Knowledge will also make papers easier to read/understand
Intuitive Introduction to Classification/Supervised Machine Learning
Classification

- Classification is type of machine learning used a lot in Ubicomp
- Classification? determine which class a sample belongs to
- Examples:

  Spam filter
  - Email
    - Spam
    - Not Spam

  Google Fit
  - User Activity
    - Walking
    - Running
    - Still
    - In vehicle
**Classifier**

- Spam filter, Google Fit run a classifier
- **Classifier:**
  - Inspects new sample, decides which class
  - Created using example-based approach
- **Classifier created using supervised machine learning**
  - Supervised: labelled data as input
  - Examples of each class => generate rules to categorize new samples
  - **E.g:** Examples of spam email, non-spam email => generate rules to categorize new email
Explaining Classification/Supervised Learning using Activity Recognition
Activity Recognition

- Want app to detect when user is performing any of the following 6 activities
  - Walking,
  - Jogging,
  - Ascending stairs,
  - Descending stairs,
  - Sitting,
  - Standing

- **Approach:** Classifier to decide user activity based on accelerometer readings
Example Accelerometer Data for Activities

Step 1: Gather lots of example accelerometer data for each activity type.
Example Accelerometer Data for Activities

(c) Ascending Stairs

(d) Descending Stairs
Gathering Accelerometer Data

- Can write simple app that retrieves accelerometer data while user is doing each of 6 activities (1 at a time)

- Label each data with activity performed.
  - E.g. label the following data as sitting
Funf (funf.org)

- Can also download, FUNF app from MIT to gather data
- Continuously collects user data in background:
  - Accelerometer readings
  - Phone calls
  - SMS messages, etc
- Simple to use:
  - Download app,
  - Check off sensors to log (e.g. accelerometer)
Step 2: Run Study to Gather Example Data

- Data collected from many (e.g. 30) subjects
- Users run Funf in their phones while performing each activity
  - Perform each of 6 activities (walking, sitting,.. Etc)
-Accelerometer data collected every 50ms
- Funf pushes data to dropbox, download data
- Now have 30 examples of each activity

![Figure 1: Axes of Motion Relative to User]
Segment Data (Windows)

- Divide raw time-series data divided into segments (e.g. 10 seconds)
Compute Features

- Within segments, compute features
- **Features:** Functions computed on accelerometer data, captures important accelerometer characteristics
- **Examples:** min-max values within segment, magnitude within segment, standard deviation, moving average
Compute Features

- **Important:** For given feature formula, each of activities should yield a different range of values
- **E.g:** Min-max Y axis range feature

![Graph showing acceleration over time for jogging and sitting activities.](image)

- Large min-max for jogging
- Small min-max for jogging
Feature Computation

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.
Machine Learning

- Pull calculated features + activity labels into Weka (or other Machine learning Framework)
What does Weka do?

- Features are just numbers
- Different values for different activities
- Weka figures out ranges corresponding to each activity
- Tries different classifier algorithms (SVM, Naïve Bayes, Random Forest, J48, etc)
- SVM example

![Activity 1 (e.g. walking) and Activity 2 (e.g. sitting) with classifier boundary](image)
Accuracy of Classifiers

- Weka also reports accuracy of each classifier type

Table 2: Accuracies of Activity Recognition

<table>
<thead>
<tr>
<th>Activity</th>
<th>J48</th>
<th>Logistic Regression</th>
<th>Multilayer Perceptron</th>
<th>Straw Man</th>
</tr>
</thead>
<tbody>
<tr>
<td>Walking</td>
<td>89.9</td>
<td>93.6</td>
<td>91.7</td>
<td>37.2</td>
</tr>
<tr>
<td>Jogging</td>
<td>96.5</td>
<td>98.0</td>
<td>98.3</td>
<td>29.2</td>
</tr>
<tr>
<td>Upstairs</td>
<td>59.3</td>
<td>27.5</td>
<td>61.5</td>
<td>12.2</td>
</tr>
<tr>
<td>Downstairs</td>
<td>55.5</td>
<td>12.3</td>
<td>44.3</td>
<td>10.0</td>
</tr>
<tr>
<td>Sitting</td>
<td>95.7</td>
<td>92.2</td>
<td>95.0</td>
<td>6.4</td>
</tr>
<tr>
<td>Standing</td>
<td>93.3</td>
<td>87.0</td>
<td>91.9</td>
<td>5.0</td>
</tr>
<tr>
<td>Overall</td>
<td>85.1</td>
<td>78.1</td>
<td>91.7</td>
<td>37.2</td>
</tr>
</tbody>
</table>
Export Classifier from Weka

- Export classifiers as Java JAR file
- Run classifier in Android app
- Classifies new accelerometer patterns while user is performing activity => Guess (infer) what activity

New accelerometer Sample in real time

Classifier in Android app

Activity (e.g. Jogging)
What if you don’t know Machine Learning

- Visually inspect accelerometer waveform, come up with rules by trial and error
- E.g. If (min-max range < threshold), activity = sitting
Concrete Examples of Classification
Voice Classification

- Voice input from Phone microphone

![Graph of observed noisy voice with features and classifier diagram showing Stressed, Nervous, Depressed, and Drunk categories.]
Facial Expression Classification

- Most of computer vision uses machine learning
- Classify camera images, to infer mood