Mobile Sensing I - Fundamentals

Spring 2015

Petteri Nurmi
About the course

• Advanced course:
  • Main specialization: Networking and Services
  • Secondary specialization: Algorithms, Data Analytics and Machine Learning
  • 8 credit units for passing the course
    – Lectures and weekly exercises
    – Home exam at the end of the course
    – Compulsory project work (approx 3 credit units)
  • Passing the course requires at least 50% of points from home exam and exercises + successful completion of project work
About the course

- Course webpage:
  - Course material, exercises, and (selected) solutions available from the webpage

- Exercises (and home exam)
  - Have to be returned electronically to the instructors, minimum 15 minutes before the exercise session
  - Exercises will be available online on Thursday
  - Selection solutions made available online after exercise session
Prerequisites

- Mathematics, Machine Learning and Data Analysis
  - Knowledge of basics of machine learning, especially classification techniques
  - Understanding of basic mathematics and statistics, especially linear and matrix algebra

- Services
  - Basic knowledge of networking concepts and middleware

- Programming
  - Implementation language can be chosen freely, but preferred languages include Matlab, python, Java
  - Mobile (Android) programming knowledge beneficial but not a formal prerequisite (basics covered in exercises)
Learning Objectives – Lecture I

• What is mobile sensing? Which steps are involved in mobile sensing applications?
• Which dimensions can be used to categorize mobile sensing applications?
• How ground truth can be collected and how its validity can be assessed?
• How sensing performance can be assessed?
What is Mobile Sensing?

- **Sensing**: perceiving/becoming aware of external or internal stimuli
- **Mobile sensing**: the use of sensors embedded on mobile devices for sensing stimuli

- What can be sensed?
  - Environmental characteristics: where the user is? Who are around the user?
  - User characteristics: what mood is the user in? What personality type is the user? What does the user like?
  - Device characteristics: remaining battery, current network connectivity, etc.
Types of Sensors

- Inertial sensors: accelerometer and gyroscope
  - Landscape/portrait mode detection
  - Health/fitness tracking
- Environment sensors
  - Temperature, humidity, barometer (air pressure)
  - Proximity and ambient light sensors (used, e.g., for adapting screen and activate energy saving policies)
- Physiological sensors
  - Heart beat, blood pressure
Types of Sensors

- “Non-sensor sensors”
  - Microphone: prosodic analysis, ambiance sensing
  - Cameras (front and/or back)
- Location-related sensors:
  - GPS (and Glonass)
  - GSM/Cellular sensor (base station, signal strength)
  - WiFi (access points and signal strengths)
- Behaviour sensors
  - Application usage behaviour, extent of calls made
- ...and many others
Examples: Transportation Behaviour Modelling

- Transportation behaviour monitoring
  - Persuasive mobility applications
  - Mobility tracking applications
  - Driving-related application
  - Monitoring gas consumption, driving skill, driver drowsiness etc.
- Traffic flow and congestion estimation
- Sensors:
  - Accelerometer and gyroscope most common, followed by GPS and radio environment (GSM, WiFi)
  - Other sensors considered include magnetometer, cameras, and microphones
Examples: UbiGreen

https://www.youtube.com/v/3iqI786nOEM
Examples: CenceMe

https://www.youtube.com/v/8rDFbTF47PA
Examples: Participatory Sensing

• Mapping phenomena in the urban environment
  • User reporting based applications, e.g., measuring CO2 at different locations or taking pictures of garbage containers

• Road condition estimation
  • E.g., Pothole Patrol and Nericell, automatically detect and report problems in roads

• Sensors:
  • Location information (GPS)
  • Inertial and magnetic sensors (accelerometer, gyroscope, magnetometer)
  • Audio (e.g., Nericell detects honking intensity)
Other examples: SoundSense

https://www.youtube.com/v/VK9AE_7dhc4
Anatomy of Mobile Sensing Application: Stages

• Three common components can be identified from the examples:
  1. Sensing: each application relied on one or more sensor data
  2. Intelligence: some information extracted from sensor data
  3. Interface level
     – Adaptation: Interface modified based on extracted information (e.g., rotating screen)
     – Sharing: providing information to friends/other users
     – Persuasion: trying to influence user behaviour based on extracted information
• Often one or more parts rely on information in external sources
Anatomy of Mobile Sensing Applications: Modes

• Sensing applications can operate in two modes
  • On-demand: sensing performed when user interacts with an application/device
    – Rotate screen when user turns phone, automatically answer phone when performing a specific gesture
    – Location-based services: determine user’s location when user interacts with the application
  • Continuous: sensor data collected and processed continually
    – Requires support for multitasking and background applications (not possible on all operating systems)
    – Resource-efficiency critical requirement, discussed in a separate lecture
Anatomy of Mobile Sensing Applications: Scale and Paradigms

• Two scales:
  • Personal sensing: focus on individual user
  • Community-sensing: focus on a group of users that share a common goal (and trusted relationship)

• Sensing paradigms can be categorized into two based on user involvement
  • Participatory sensing: users actively contribute data
    – Common in community sensing applications
  • Opportunistic sensing: sensor data collected (and processed) without user involvement
    – Most common mode for personal sensing application
Distributing Sensing Apps: Application Stores

• Different *app stores* are main distribution channel for mobile sensing applications
  • Apple App Store, Google Play, Microsoft Mobile Marketplace, etc.
  • Scale changed drastically:
    – First apps 5-50 users
    – Carat (carat.cs.helsinki.fi): over 700,000 users
• Several challenges:
  • Biases in data (population biases, context/usage biases, device biases)
  • Reliability of measurements collected from applications difficult to assess
  • Applications need to be “attractive” to users (good usability, scalability, energy-efficiency, …)
Sensor data processing steps can be divided into a pipeline:

- **Sensor data extraction**
  - Typically extracted in *frames*, e.g., all measurements within a second window (common window sizes within range 0.5 – 4 seconds)

- **Preprocessing**
  - Removing errors from data, filtering
  - Typical operations: low/high-pass filtering, jitter removal

- **Feature extraction**

- **Classification / Regression**
  - Extracting the relevant information from the sensor data
Sensing Pipeline: Example
Developing Sensing Pipeline

- Developing sensing pipelines a multi-stage process
  1. Determine phenomena to sense and sensors to use
  2. Collect sensor data of the phenomena together with ground truth *labels*
  3. Determine appropriate preprocessing steps and features that are good for the phenomena
  4. Train classifiers / regression models based on the features and ground truth
  5. Deploy trained classifiers as part of the application

Repeat steps 2 – 5 as needed
Obtaining Ground Truth

• Developing sensing applications requires *ground truth* of the phenomena that is being sensed
  • Examples: current transportation mode, number of steps taken, personality type of a user
  • Typically the ground truth is given as a discrete category/class or as a numeric value
    • Discrete class: transportation mode (walking, running, train, ...) ➔ classification problem
    • Numeric value: questionnaire-based measures, e.g., extroversion (value between 1 – 48) ➔ regression
    • Labels assigned to frames (not individual measurements)
Obtaining Ground Truth: Labelling / Annotation

• When labels are classes / categories, most common way is to manually assign the corresponding labels
  • Self-annotations: user collecting the data marks the appropriate class / category
    – Diary: periodic labelling
    – Experience sampling: prompting the user to label
  • Shadowing: external person observes the user and assigns the appropriate label
  • Video-annotation: activity recorded and an external person assigns labels based on the recording
• Typically “event-based” annotation
  • Only starting/ending points of activities labelled
Error Sources in Reporting

- Self-reporting biases:
  - Recall: Forgetting to annotate some event or remembering something wrong
  - Self-bias: Intentionally manipulating labels, e.g., leaving out information or modifying information

- Participant observation (shadowing) biases:
  - Behaviour bias: user modifies his/her behaviour due to awareness of being observed

- Coding errors
  - Ambiguity and non-uniqueness of labels/actions (e.g., walking in a train)

- Latency
  - Start/end of activity starts/ends too early/late
Example: Authentication (BlueProximity)

- Example of experience sampling-based labelling
  - Prompts the user and asks for appropriate label at pre-defined intervals
  - Binary labels: close / far
- In this case, no logging of motion sensors → user interactions have no effect on collected data
  - Hence prompting the user valid approach
Example: Transportation Mode Detection

- Discrete set of different classes (10 + 1)
  - Separate NULL class for data that does not conform to any of the activities of interest
- Assigning labels *during* the activity not desirable
  - Requires interactions with the device, which influence collected data
- Alternatives
  - Assigning labels post activity
  - Separate person (and device) responsible for labelling
Multiple Annotators

- “Two heads are better than one”
  - Reliability of annotations can be (potentially) improved by combining measurements from multiple annotators
  - But annotators need to agree on the “scheme” that is used to assign labels
- Coding scheme
  - Definition of classes/categories and instructions on when to assign them
  - Widely used especially in linguistics (including speech annotations)
Multiple Annotators

- *Inter-rater agreement* measures the rate at which annotators agree on labels
  - High agreement ➔ reliable labels
  - Low agreement ➔ unreliable labels
- Cohen’s kappa coefficient
  - One of the most widely used measures of agreement between annotators (for categorical data)
  - $\kappa = \frac{\Pr(u) - \Pr(e)}{1 - \Pr(e)}$
  - $\Pr(a)$ – relative observed agreement
  - $\Pr(e)$ – hypothetical probability of agreeing by chance
Example

- Consider the given “confusion” matrix for two raters
  - \( P(\text{walking} \mid A) = 95 / 202, P(\text{running} \mid A) = 107 / 202 \)
  - \( P(\text{walking} \mid B) = 99 / 202, P(\text{running} \mid B) = 103 / 202 \)
  - \( P(\text{walking} \mid A \text{ and } B) = 99 * 95 / (202 * 202) \)
  - \( P(\text{running} \mid A \text{ and } B) = 103 * 107 / (202 * 202) \Rightarrow \Pr(e) = 0.5 \)

- Agreement: \( \Pr(a) = (92 + 100) / 202 = 0.95 \Rightarrow \kappa = (0.95 - 0.5) / (1 - 0.5) = 0.9 \)

- Overall agreement is high \( \Rightarrow \) annotations reliable and those where annotators agree can be included in analysis
Numeric Labels: Psychometric Instruments

- Psychometrics measures *psychological* aspects of the user (personality, mood, emotion)
- One of the most important categories of applications where labels are numeric (instead of categorical)
- Ground truth collected using *standardized questionnaires*
  - Consist of several *items* or questions
  - Responses typically elicited using *Likert-scales* (e.g., 1-5, 1-7 or 1-9)
  - Typically multiple items grouped together to form a *scale*
    - Example: Big-5 (personality) questionnaire has 5 scales measuring different personality traits
<table>
<thead>
<tr>
<th>Question</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Are you a talkative person?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Are you rather lively?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Do you enjoy meeting new people?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Can you usually let yourself go and enjoy yourself at a lively party?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Do you usually take the initiative in making new friends?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Can you easily get some life into a dull party?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Do you tend to keep in the background on social occasions? (Reversed)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Do you like mixing with people?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Do you like to plenty of action and excitement around you?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Are you mostly quiet when you are with other people? (Reversed)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Do other people think of you as being very lively?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Can you get a party going?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Numeric Labels: Consistency and Cronbach’s Alpha

- Measure of *reliability* and *internal consistency*
  - How well different measures correlate with each other
  - Used to validate questionnaire items in scales

- Cronbach’s alpha:
  - $\delta_X^2$ variance in observed total test scores
  - $\delta_{Y_i}^2$ variance of component (questionnaire item) $i$
  - $K$ is the number of items

$$\alpha = \frac{K}{K-1} \left( 1 - \sum_{i=1}^{K} \frac{\sigma_{Y_i}^2}{\sigma_X^2} \right)$$
### Cronbach’s Alpha: Example

- **Number of items** \( K = 5 \)
- **Sum of item variances** = 9.9
- **Variance in observed test scores (variance of user sums)** = 34.3
- **Cronbach’s alpha** = \( \frac{5}{4} \ast (1 - \frac{9.9}{34.9}) = 0.89 \)
- **Rule-of-thumb**: 0.7 minimum acceptable value for consistency

<table>
<thead>
<tr>
<th>Item</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>User 1</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>19</td>
</tr>
<tr>
<td>User 2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>User 3</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>23</td>
</tr>
<tr>
<td>User 4</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>User 5</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>12</td>
</tr>
<tr>
<td>Variance</td>
<td>3.8</td>
<td>1.8</td>
<td>2.3</td>
<td>0.7</td>
<td>1.3</td>
<td></td>
</tr>
</tbody>
</table>
Evaluation Criteria

• Performance of sensing algorithms:
  • Standard classification/regression metrics (precision, recall, F1-score, mean absolute error)
  • Event-based metrics (underfill and overfill rate, fragmentation rate)
• End-user performance criteria (covered in a later lecture):
  • Energy-efficiency / battery drain
  • Usability
Classification Metrics

- Most common approach for evaluation sensing algorithms
- Consider the matrix on the left:
  - Precision = TP / (TP + FP)
  - Recall = TP / (TP + FN)
  - F1-score = 2TP / (2TP + FP + FN)
    - Harmonic mean of precision and recall
    - Equal to 2 * (precision * recall) / (precision + recall)
Classification Metrics and Frame-Based Scoring

- Precision and recall examples of *frame* based metrics
  - Classifier applied separately for each frame and metrics calculated by comparing output to ground truth
  - Scoring: for each class and sample, determine whether output is true positive (TP), false positive (FP), true negative (TN), or false negative (FN)
  - Precision/recall/F1-score calculated from scoring

- Example:

<table>
<thead>
<tr>
<th>Ground truth</th>
<th>W</th>
<th>W</th>
<th>W</th>
<th>W</th>
<th>W</th>
<th>W</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classifier I</td>
<td>W</td>
<td>S</td>
<td>W</td>
<td>S</td>
<td>W</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>Walking (W)</td>
<td>TP</td>
<td>FN</td>
<td>TP</td>
<td>FN</td>
<td>TP</td>
<td>FN</td>
<td>TN</td>
</tr>
<tr>
<td>Standing (S)</td>
<td>TN</td>
<td>FP</td>
<td>TN</td>
<td>FP</td>
<td>TN</td>
<td>FP</td>
<td>TP</td>
</tr>
</tbody>
</table>
Often classification metrics are combined with a *confusion matrix*:
- Table that shows what kinds of misclassifications occur
- Can be used for calculating precision, recall and F1-score

**Example:**
- Precision(Still) = \(\frac{35654}{35654 + 14 + 564 + 976 + 1500 + 3187}\) = 0.85
- Recall(Still) = \(\frac{35654}{35654 + 58 + 218 + 235 + 433 + 1784}\) = 0.929

<table>
<thead>
<tr>
<th>Predictions</th>
<th>Still</th>
<th>Walking</th>
<th>Bus</th>
<th>Train</th>
<th>Metro</th>
<th>Tram</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Still</td>
<td>35,654</td>
<td>58</td>
<td>218</td>
<td>235</td>
<td>433</td>
<td>1,784</td>
<td>85.1</td>
<td>92.9</td>
<td>88.8</td>
</tr>
<tr>
<td>Walking</td>
<td>14</td>
<td>12,823</td>
<td>371</td>
<td>10</td>
<td>11</td>
<td>61</td>
<td>96.2</td>
<td>96.5</td>
<td>96.3</td>
</tr>
<tr>
<td>Bus</td>
<td>564</td>
<td>318</td>
<td>3,811</td>
<td>207</td>
<td>356</td>
<td>1,540</td>
<td>58.9</td>
<td>56.1</td>
<td>57.4</td>
</tr>
<tr>
<td>Train</td>
<td>976</td>
<td>50</td>
<td>238</td>
<td>319</td>
<td>423</td>
<td>540</td>
<td>23.4</td>
<td>12.5</td>
<td>16.3</td>
</tr>
<tr>
<td>Metro</td>
<td>1,500</td>
<td>11</td>
<td>444</td>
<td>347</td>
<td>823</td>
<td>621</td>
<td>34.6</td>
<td>22.0</td>
<td>26.9</td>
</tr>
<tr>
<td>Tram</td>
<td>3,187</td>
<td>70</td>
<td>1,393</td>
<td>244</td>
<td>332</td>
<td>2,762</td>
<td>37.8</td>
<td>34.6</td>
<td>36.1</td>
</tr>
</tbody>
</table>
Classification Metrics: Macro-Averaging

- Standard classification metrics ignore the distribution of the different classes (biased towards most frequent class)
- Alternative is to use macro-averaging where precision/recall/F1-score is calculated as a weighted average using distribution of the class as weight
- Example:
  - $P(\text{still}) = 0.527$, $P(\text{walking}) = 0.183$, ...
  - $F_1^M = \frac{\sum(P(\text{class}) \times \text{Precision(\text{class})})}{\sum P(\text{class})} = 0.747$

<table>
<thead>
<tr>
<th>Predictions</th>
<th>Still</th>
<th>Walking</th>
<th>Bus</th>
<th>Train</th>
<th>Metro</th>
<th>Tram</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Still</td>
<td>35,654</td>
<td>58</td>
<td>218</td>
<td>235</td>
<td>433</td>
<td>1,784</td>
<td>85.1</td>
<td>92.9</td>
<td>88.8</td>
</tr>
<tr>
<td>Walking</td>
<td>14</td>
<td>12,823</td>
<td>371</td>
<td>10</td>
<td>11</td>
<td>61</td>
<td>96.2</td>
<td>96.5</td>
<td>96.3</td>
</tr>
<tr>
<td>Bus</td>
<td>564</td>
<td>318</td>
<td>3,811</td>
<td>207</td>
<td>356</td>
<td>1,540</td>
<td>58.9</td>
<td>56.1</td>
<td>57.4</td>
</tr>
<tr>
<td>Train</td>
<td>976</td>
<td>50</td>
<td>238</td>
<td>319</td>
<td>423</td>
<td>540</td>
<td>23.4</td>
<td>12.5</td>
<td>16.3</td>
</tr>
<tr>
<td>Metro</td>
<td>1,500</td>
<td>11</td>
<td>444</td>
<td>347</td>
<td>823</td>
<td>621</td>
<td>34.6</td>
<td>22.0</td>
<td>26.9</td>
</tr>
<tr>
<td>Tram</td>
<td>3,187</td>
<td>70</td>
<td>1,393</td>
<td>244</td>
<td>332</td>
<td>2,762</td>
<td>37.8</td>
<td>34.6</td>
<td>36.1</td>
</tr>
</tbody>
</table>
Event-Based Metrics

- In practice sensing focuses on *events* that continue for longer period of time
  - Walking typically lasts longer than 1-4 seconds (i.e., several frames)
- *Event*-based metrics provide additional information about how well the different events are recovered
- Event defined as continuous sequence with the same label assignment
Event-Based Metrics: Motivation

Consider the example below

<table>
<thead>
<tr>
<th>Ground truth</th>
<th>W</th>
<th>W</th>
<th>W</th>
<th>W</th>
<th>W</th>
<th>W</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classifier I</td>
<td>W</td>
<td>S</td>
<td>W</td>
<td>S</td>
<td>W</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>Classifier II</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
</tr>
</tbody>
</table>

- Equal precision and recall, but very different behaviour:
  - Classifier I has high *fragmentation rate*, i.e., the walking event is split into multiple pieces
  - Classifier II *underfills* walking in the beginning as it has delay in starting to detect it
  - Classifier II *overfills* walking at the end as it continues to detect it even if activity changed
Event-Based Metrics: Segmentation

- Segment: continuous segment where **BOTH** ground truth and classifier output remain same
  - Basic unit for calculating event-based metrics
  - Segmentation NOT unique for a given ground truth sequence
- Example:
  - Classifier I: 7 segments (either ground truth or output change)
  - Classifier II: 3 segments

<table>
<thead>
<tr>
<th>Ground truth</th>
<th>W</th>
<th>W</th>
<th>W</th>
<th>W</th>
<th>W</th>
<th>W</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classifier I</td>
<td>W</td>
<td>S</td>
<td>W</td>
<td>S</td>
<td>W</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>Classifier II</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
</tr>
</tbody>
</table>
Event-Based Metrics: Segment Scoring

- Types of errors
  - **Insertion**: FP segment that corresponds exactly to an inserted return
  - **Merge**: FP that occurs between two TP segments
  - **Overfill**: FP that occurs at the or end of a partially matched return

<table>
<thead>
<tr>
<th>Ground truth</th>
<th>W</th>
<th>W</th>
<th>S</th>
<th>W</th>
<th>W</th>
<th>W</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classifier III</td>
<td>S</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
</tr>
<tr>
<td>Walking (W)</td>
<td>FN</td>
<td>TP</td>
<td>FP</td>
<td>TP</td>
<td>TP</td>
<td>TP</td>
<td>FP</td>
</tr>
<tr>
<td>Standing (S)</td>
<td>FP</td>
<td>TN</td>
<td>FN</td>
<td>TN</td>
<td>TN</td>
<td>TN</td>
<td>FN</td>
</tr>
</tbody>
</table>
Event-Based Metrics: Segment Scoring

Types of errors (continued):

- **Deletion**: FN that corresponds exactly to a deleted event
- **Fragmentation**: FN that occurs between two TP segments
- **Underfill**: FN that occurs at the start or end of an event

<table>
<thead>
<tr>
<th>Ground truth</th>
<th>W</th>
<th>W</th>
<th>W</th>
<th>W</th>
<th>W</th>
<th>W</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classifier III</td>
<td>S</td>
<td>W</td>
<td>S</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
</tr>
<tr>
<td>Walking (W)</td>
<td>FN</td>
<td>TP</td>
<td>FN</td>
<td>TP</td>
<td>TP</td>
<td>TP</td>
<td>FP</td>
</tr>
<tr>
<td>Standing (S)</td>
<td>FP</td>
<td>TN</td>
<td>FP</td>
<td>TN</td>
<td>TN</td>
<td>TN</td>
<td>FN</td>
</tr>
</tbody>
</table>
Event-Based Metrics

- Segment scores typically converted to rates
  - Fraction of events that manifest given behaviour
  - Most widely used (and most beneficial) rates:
    - *Fragmentation rate*: Number of (true) events that contain fragmented returns
    - *Underfill rate*: Number of true events that have delay in detecting them
    - *Overfill rate*: Number of true events that are recognized after the event has ended
  - Note: underfill rate NOT same as latency, simply tells how often there is latency in detection
Other (sensing) metrics

- Regression metrics (continuous/numeric targets)
  - Mean absolute error (MAE): \( \sum_i |\text{predicted}_i - \text{value}_i| \)
- Latency
  - Difference in the start time of event and the first true positive of classifier
  - Effectively the extent of underfills
- Energy and computational cost
  - Resource requirements of the algorithms
  - Computational overhead
Summary

• Mobile Sensing
  • The use of mobile devices for acquiring information about phenomena (user/device/environment)
  • Sensing pipeline, scale, and paradigms
• Ground truth collection: several options with different pros and cons
• Several evaluation criteria:
  • Sensing performance: classification metrics and event-based metrics
  • Usability, end-user factors, energy-efficiency
<table>
<thead>
<tr>
<th>Reference</th>
<th>Title</th>
</tr>
</thead>
</table>
References


