Machine Learning for Makers: Interactive Sensor Data Classification Based on Augmented Code Examples

David Mellis, Ben Zhang, Audrey Leung, Björn Hartmann
EECS and School of Information, University of California, Berkeley
Data vs. High-level Information

- Accelerometer
  - gesture
- Gyroscope
  - posture
- Humidity
- Temperature
  - building comfort level
- Odometer
  - driving behavior
- Speedometer
- Wearables
  - activity
- Color
- GPS
  - semantic location
- Microphone
  - speaker

Image from Invention Lab [http://invent.citris-uc.org](http://invent.citris-uc.org)
Machine Learning To Help, but . . .
Machine Learning To Help, but . . .

- Build Physical Sensing Systems
- Read / Visualize Sensor Data
- Define / Modify ML Pipeline
- Add / Modify Training Data
- Train Model
- Run Classification

Tune Parameter

Data → Pre-Processing → Feature Extraction → Classification → Post-Processing → Classification

- Decision Tree
- AdaBoost
- GMM
- LDA
- SVM
- Naive Bayes
- Random Forest
- Knn
- DTW
- …
Existing Approaches That Increase ML's Accessibility for Novices

• Domain-specific Interfaces
  - Crayons / EyePatch for images, Exemplar / MAGIC for gestures
  - Do not scale and require novices (experts) learning (building) different tools for different problems

• General Machine Learning Libraries
  - scikit learn, Wekinator, ml.lib
  - Powerful / developer-friendly, but require configuring details

• Augmented Code and Interface
  - Jupyter / iPython notebook, but not focusing on interactive ML
Existing Approaches That Increase ML's Accessibility for Novices

• Domain-specific Interfaces
  - Crayons / EyePatch for images, Exemplar / MAGIC for gestures
  - Do not scale and require novices (experts) learning (building) different tools for different problems

• General Machine Learning Libraries
  - scikit learn, Wekinator, ml.lib, GRT, …
  - Powerful / developer-friendly, but require configuring details

GRT (Gesture Recognition Toolkit) Documentation for DTW (Dynamic Time Warping)
Existing Approaches That Increase ML's Accessibility for Novices

• Domain-specific Interfaces
  - Crayons / EyePatch for images, Exemplar / MAGIC for gestures
  - Do not scale and require novices (experts) learning (building) different tools for different problems

• General Machine Learning Libraries
  - scikit learn, Wekinator, ml.lib, GRT, …
  - Powerful / developer-friendly, but require configuring details

• Augmented Code and Interface
  - Jupyter / iPython notebook, but not focusing on interactive ML

Jupyter Notebook
Augmented Code Examples for Interactive ML

Writing Examples in Code
- Definition of ML pipeline
- Encoding of expertise
- Reference training data

Adapting Examples
- With their own data
- Tuning systems behavior
- Incorporating into a larger project
ESP: Example-based Sensor Predictions

- Build Physical Sensing Systems
- Read / Visualize Sensor Data
- Define ML Pipeline
- Add / Modify Training Data
- Train Model
- Run Classification
- Tune Parameter
- Pre-Processing
- Feature Extraction
- Classification
- Post-Processing
- Classification

Data

- AdaBoost
- Decision Tree
- GMM
- LDA
- SVM
- Naive Bayes
- Random Forest
- Knn
- DTW
- ...

ML Pipeline

Pre-Processing

Feature Extraction

Classification

Post-Processing
Ingredients for Successful Transfers

Sensor Calibration
Tunable Parameters
Domain Knowledge
Data Management
ML Pipeline

Experts

ESP

Makers
Calibration: Supporting Portability Across Sensor Models

<table>
<thead>
<tr>
<th>Accelerometer</th>
<th>Microphone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert</td>
<td>Expert</td>
</tr>
<tr>
<td>MMA8451</td>
<td>Mac OS built-in</td>
</tr>
<tr>
<td>±8g</td>
<td></td>
</tr>
<tr>
<td>Maker</td>
<td>Maker</td>
</tr>
<tr>
<td>ADXL335</td>
<td>MAX9814</td>
</tr>
<tr>
<td>±3g</td>
<td></td>
</tr>
</tbody>
</table>

```java
calibrator.addCalibrateProcess("Resting", "Rest accelerometer on flat surface.", restingDataCollected);
```
Tunable Parameters: A Smaller, App-specific Parameter Space

```java
registerTunable(null_rej, 0.1, 5.0, "Variability", "How different from the training data a new gesture can be and "
"still be considered the same gesture. The higher the number, the "
"more different it can be.", updateVariability);
```
Domain Knowledge and Heuristics

- Example knowledge for gesture recognition:
  - In calibration, accelerometer is not flat
  - In calibration, accelerometer readings are noisy
  - In collecting samples, gesture contains little movement
  - In collecting samples, gesture is too short or too long

- ESP provides
  - Callback functions for expertise embedding
  - Warning and error messages on the GUI

```cpp
TrainingSampleCheckerResult checkTrainingSample(const MatrixDouble &in) {
    VectorDouble stddev = in.getStdDev();
    if (*max_element(stddev.begin(), stddev.end()) < 0.1) {
        return TrainingSampleCheckerResult(TrainingSampleCheckerResult::WARNING, "Warning: Gesture contains very little movement.");
    }
    return TrainingSampleCheckerResult::SUCCESS;
}
```
Training Data Management

- Basic Functionalities
  - rename, delete, trim, relable
  - data loading / saving

- Scoring to assist sample collection
  - confusion score (should I delete a bad sample?)
  - information gain (should I continue to add samples?)
ESP Modules For Successful Transfers

- Calibration: supporting portability across sensor models
- Tunable parameters: a smaller, app-specific parameter space
- Domain knowledge and heuristics for sample collection
- Training data management to help collect good samples
Gesture Recognition Using Accelerometer and ESP (DTW algorithm)

https://tinyurl.com/esp-gesture
=> https://create.arduino.cc/projecthub/mellis/gesture-recognition-using-accelerometer-and-esp-71faa1
The ESP generates an interface based on specific pipeline configuration to visualize live data.
The maker collects calibration samples according to instructions provided by experts.

For accelerometer, we are calibrating the bias (0g) and sensitivity (1g).
The maker can either use training data from experts or add his own training data.

For tennis gestures, he records forehand, backhand and serve gestures.
The maker then trains the model and sees classification on live data.

The gesture recognition example uses the DTW classifier.
The ESP interface provides further analysis such as showing the likelihood of each class and the distance to each class for live data.
To further improve the classification performance, the maker can tune pipeline parameters.

For gesture recognition, the maker can tune variability that control how different a new gesture can be.
Once the model is trained, each training sample is scored. The user can modify the training data interactively within ESP.

- rename, delete, trim, relable
- confusion score and information gain
The Life of an ESP-based Project

ML Implementation

- ESP File
- Download Example
- Build Physical Sensing System

Build Physical Sensing System → ESP File → Download Example → Calibrators → Calibrate Sensors → Add / Modify Training Data → Train Model → Change Pipeline Parameters → Data Management & Scoring Interface → Pipeline Parameter Editor

ML Analysis

Run Classifier on Live or Stored Data → ESP Runtime Sensor Data Visualization → Classifier Visualization

Observe Performance
Embedding in Interactive Projects

Serial
TCP
Built-in Mic
OSC*

Serial
TCP
Keyboard Emulation

*OSC: Open Sound Control protocol
The experts provide,

- ML pipeline definition (F)
- Calibration across sensors (A, E)
- Tunable parameters (C, G)
- Expert knowledge and heuristics (B, H)
- Reference training data (*no code)

*The code is not meant to be read; check out the paper and our repository.*
## Example Applications

<table>
<thead>
<tr>
<th>Example</th>
<th>Sensor</th>
<th>Calibration</th>
<th>Features</th>
<th>Classifier</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gesture Recognition</td>
<td>Accelerometer</td>
<td>Bias &amp; Sensitivity</td>
<td>Raw XYZ</td>
<td>DTW</td>
<td>Variability &amp; Timeout</td>
</tr>
<tr>
<td>Touché [1]</td>
<td>Electrode</td>
<td>-</td>
<td>Frequency Response</td>
<td>SVM</td>
<td>-</td>
</tr>
<tr>
<td>Speaker Identification</td>
<td>Microphone</td>
<td>Bias &amp; Sensitivity</td>
<td>MFCC</td>
<td>GMM</td>
<td>Noise Level</td>
</tr>
<tr>
<td>Color Detection</td>
<td>Color Sensor</td>
<td>-</td>
<td>Raw RGB</td>
<td>Naive Bayes</td>
<td>Scaling &amp; Variability</td>
</tr>
</tbody>
</table>

Touché - Swept Frequency Capacitive Sensing
by Sato M, Poupyrev I, Harrison C, CMU & Disney Research, CHI’12
Successful Transfer of Provided Examples

adapted from our gesture recognition example
Successful Transfer of Provided Examples

adapted from our Touché example
Workshop Lessons

+ All users are able to follow the expert-authored tutorials

+ ESP provided makers with insight into real-world use of machine learning
  + "I could incorporate machine learning into an interactive project if I wanted to"

+ Successful transfer of provided examples to users' tasks (interface level)

+ Workshop suggested the accessibility of example modification (source-code level)
  + Two participants (P5 and P7) modified one of the provided examples to work with their own sensor --- a light sensor (P5) and a simple capacitive electrode (P7)

- Workshop facilitated by ESP authors
  - makers received immediate feedback; unrealistic in comparison with real-world online community
This was totally real and raw and I was generating the data but then I was training model and seeing how well the classifier worked. And that whole process for me of being really hands-on with it really helped my way of thinking about it, thinking about all the steps along the way.

- P5, in closing discussion
Limitations and Future Directions

• We focused on classification problems, not regression

• ESP is best suited for "small data" (interactiveness)

• Cost of generality (ESP vs. interface with custom visualization)

• No embedded runtime (ESP port to embedded systems work in progress)
ESP is open source

- BSD 3 Clause
- Repo: https://github.com/damellis/ESP
- Wiki: https://github.com/damellis/ESP/wiki
- Tutorial: https://tinyurl.com/esp-gesture
- Video: http://tinyurl.com/esp-youtube-video
- Processing mode: https://github.com/damellis/processing-esp-mode
ESP (Example-based Sensor Prediction) leverages expert-authored examples to scaffold makers for machine learning projects.

<table>
<thead>
<tr>
<th>Domain</th>
<th>ESP</th>
<th>Experts</th>
<th>Makers</th>
</tr>
</thead>
<tbody>
<tr>
<td>ML Pipeline</td>
<td>Visualization / Predictions</td>
<td>Codes the definition</td>
<td>Downloads and uses</td>
</tr>
<tr>
<td>Calibration</td>
<td></td>
<td>Specifies required samples</td>
<td>Collects samples</td>
</tr>
<tr>
<td>Tunable Parameters</td>
<td>Specifications Interfaces Instructions</td>
<td>Selects and documents tunable parameters</td>
<td>Tweaks the parameter</td>
</tr>
<tr>
<td>Domain Knowledge</td>
<td></td>
<td>Assesses sample quality</td>
<td>Handle warning / error messages</td>
</tr>
<tr>
<td>Data Management</td>
<td></td>
<td>Provides reference data</td>
<td>Collects samples</td>
</tr>
<tr>
<td>Input/Output</td>
<td>Implementation</td>
<td>Selects appropriate streams</td>
<td>Adjusts streams</td>
</tr>
</tbody>
</table>

https://github.com/damellis/ESP
Backup Slides

- Slides Table of Contents
  - Motivation
  - Existing Approaches
  - ESP Intro Video
  - ESP Modules
  - ESP Workflows
  - Examples
  - Workshop
  - Limitation and Future Work

- Maker + ML
  - Touché
  - Drinke
  - Lamello

- QA
  - Why P in ESP?
DRINKE: TRACK YOUR HYDRATION

**Anatomy**
- **LCD Screen**: Displays readings and statistics
- **Temperature**
- **RGB LED and Light Sensor**: Detects luminescence to determine beverage type
- **The Brain of the Operation**: PCB & Arduino
- **Force Sensor**: Detects weight, calculates the beverage volume
- **Client Mobile Website**
  - Home & Alerts
  - Overview Chart
  - History Chart

**Light Readings by Drink**

**Drink Types**
- Water
- Peppermint Tea
- San Pellegrino Orange
- Snapple Tea
- Coke
- Coffee
- Coffee with Cream
Lamello: Passive Acoustic Sensing for Tangible Input

Related Work


Gestalt, Patel et al., 2010.
Q & A: Prediction in ESP

• Why the P in ESP is "prediction"? You are not predicting the future.
  - The term common in supervised learning: predicting an output variable from observations.
  - You have APIs like "knn.predict(iris_X_test)" in scikit learn
Q & A: Can ESP run on embedded / cloud?

• Current ESP runs on a wide variety of platforms

• Not yet on embedded systems with limited RAM / CPU
  - A work-in-progress https://github.com/damellis/grt/tree/arm

• ESP works on Linux, so easily on the cloud
Q & A: How would the maker change the calibration?

• Makers have direct access to the code
• If no changes on the code, the maker can only follow the instructions