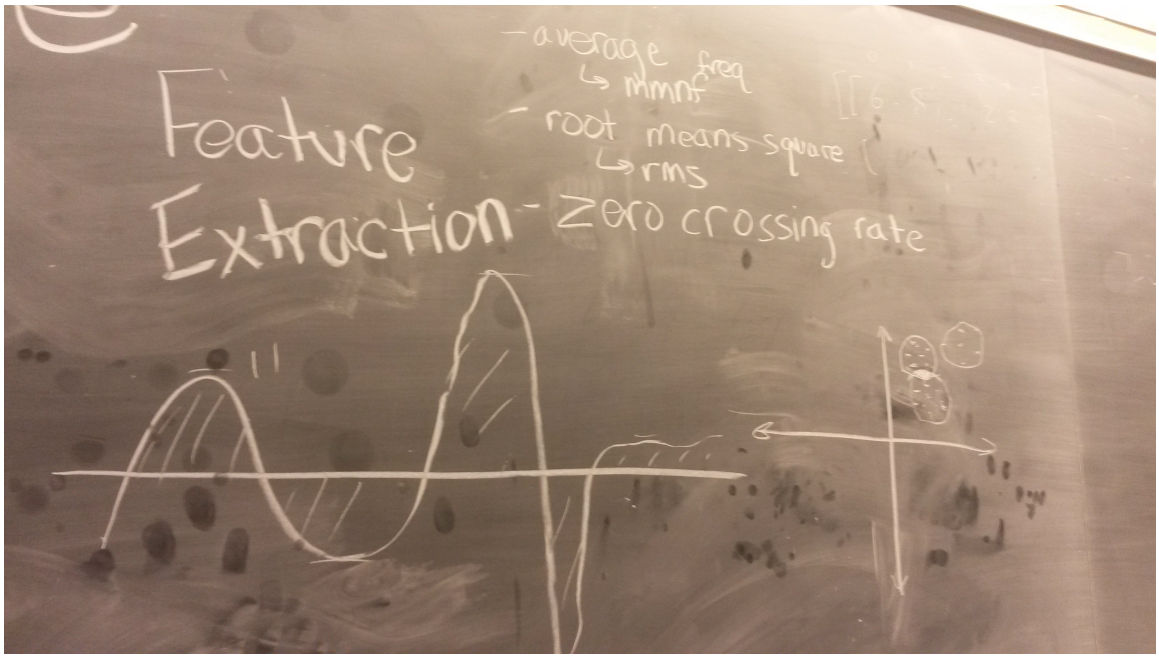
Hack the North Part 2 - Feature Extraction

Rushi Shah

20 September 2015



So what is feature extraction? Basically, you want to take a curve and represent the entire curve as one value. That value is called the feature. We needed to do this at Hack the North because we wanted to compare the output of a sensor with past data to classify it. It is harder to compare two curves than it is to compare two values, and thus we just treated the curves as their extracted feature and compared the features. How might one extract a feature from a curve, you ask?

1 Options

- Root mean square (RMS): square every value (to make sure everything is positive), then calculate the mean of the list, and calculate the square root (to undo the initial squaring).
- Mean Absolute Value (MAV): similar to RMS, this just takes the mean absolute value of the list rather than squaring square-rooting every element

- Zero crossing rate: represent the curve as the number of times it crosses the x-axis
- Slope sign change (SSC): basically the zero crossing rate of the derivative of a curve made by the list values.
- Integral: the area between the curve and the x-axis

2 What we needed

After analyzing some previous literature (1 and 2) we decided to use the RMS method because of it's past successes, ease of implementation, and relevance to the Myo data.

The Myo itself will provide us with eight sensors worth of data. Each sensor provides it's own curve that we want to extract a feature from. Thus given an array of eight arrays (which contain data points that represent a curve), we need to return an array of eight number values (eight features).

3 Implementation

For all you imperative programmers out there, this just screams for loop, right? Well old habits die hard so initially I whipped this simple implementation up:

```
import numpy as np
def extractFeatures(arr):
  res = []
  for x in arr:
    res.push(np.sqrt(np.average(np.square(np.array(x)))))
  return res
```

But at this point in the hackathon, I had been using Haskell for the whole weekend, so I improved it by mapping an `rms` function on each sensor's curve:

```
import numpy as np
def rms(arr):
  return np.sqrt(np.average(np.square(np.array(arr))))
def extractFeatures(arr):
  # list because http://stackoverflow.com/a/1303354/3861396
  return list(map(rms, arr))
```

That's really all we needed as far as Feature Extraction goes. With that being said, I stumbled on some very intriguing papers while researching so if you're interested, check them out. This one outlines an experiment on user-independent and user-specific EMG Hand Gesture recognition for solving virtual Rubik's Cubes. This one gives a nice overview of the different types of Feature Extractions like `SSC` and `MAV`.