Assignment 1. Step Count Detection Using Smartphone Sensor

RISHIRAJ ADHIKARY, Indian Institute of Technology, Gandhinagar, India

Step detection or step counting is a classic feature available in all smartphones and smartwatches. This assignment analyses two well-known algorithms, namely peak detection and adaptive peak detection based on Z-score, to estimate step counts. We found that peak detection generally works well for three locations (Hip, Hand and Pocket) of the smartphone in the human body without recalibration of the algorithm. On the other hand, adaptive peak detection gives the best mean absolute error if calibrated to a particular position in the human body. We conclude adaptive peak detection is suitable for wearables such as smartwatches, whereas thresholding peak detection would perform better for smartphones.

Additional Key Words and Phrases: datasets, accelerometer, step detection

ACM Reference Format:
Rishiraj Adhikary. 2021. Assignment 1. Step Count Detection Using Smartphone Sensor. In . ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Smartphone application stores are flooded with applications that claim to count steps accurately. However, even Google Fit Activity applications failed to estimate the step count accurately via a smartphone. We tried several other step counting applications besides Google Fit during this assignment, namely, Leap fitness, StepsApp pedometer, etc., but none could be used as a ground truth system. We, therefore, relied on our own self counting of steps as a ground truth. The objective of this assignment was to estimate the step counts as accurately as possible by collecting the accelerometer data from the smartphone. The accelerometer data were collected using the MATLAB mobile application. The retrieved data was later analysed offline. We implemented two widely known algorithms [1, 2] and reported the mean percentage error. The complete analysis can be found in the GitHub repository.

2 APPROACH

The change in magnitude of acceleration can be used to infer the number of steps. A sudden change in magnitude gives rise to peaks in a time-series waveform. This section will explain the approach to implementing the Windowed Peak detection algorithm and the modified peak detection algorithm.

Baseline Peak Detection [2]:

(1) Mean subtracting and detrending: The mean of the accelerometer signal's magnitude is first subtracted from the signal so that we can count the number of zero crossings. Given that steps occur in a cyclic pattern, we detrend the signal to get rid of non-cyclic trends.
(2) Smoothing: We remove low-frequency noise arising out of the movement of the cellphone itself by using a low pass filter.

Ubicomp Class '21, Semester 1, 2021, Gandhinagar, India

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/1122445.1122456

(3) Step tracking: We use an existing peak detection algorithm to detect peaks in the waveform. Detecting the number of peaks in a waveform is subjective unless the minimum and maximum peak height (concerning the amplitude) and the minimum distance between the peak is defined. While the minimum distance between the peak is a function of the sampling rate, the height is deduced empirically.

Accelerometer signal corresponding to walking may inhibit peak height and width change depending on the stride length. We experimented with an adaptive peak detection algorithm that dynamically adjusts the threshold for the height and width of the peak.

Adaptive Peak Detection [1]: It works on the principle of change in the standard deviation of the signal. A data point is considered a peak if the data point is certain standard deviations away from the moving mean. The z-score of the signal multiplied with the standard deviation forms the threshold of classifying a data point as a peak. This algorithm is robust since it constructs a separate moving mean and deviation. The algorithm has three hyperparameters:

(1) Lag: It determines the smoothing of the signal. The lag should be higher if the signal is stationary. The algorithm adapts to the time-varying trend in the signal depending on the lag parameter.
(2) Influence: The peak detection threshold changes dynamically depending on this parameter. Influence is a discrete random variable. A value of 0 means that the peak does not influence the threshold. It implies that future peaks are detected based on the mean and standard deviation of the signal that is not influenced by the past peaks. A value of 1 implies that the signal leads to a structural break of the long-term average of the time series.

3 EVALUATION

	Location	Number of accelerometer recording
1	Leg-Orientation 1 (Front Pocket)	8
2	Leg- Orientation 2 (Front Pocket)	6
3	Hip (Phone strapped to the belt)	3
4	Hand (Phone carried on the hand)	3

Table 1. Data was collected across two session on two participant. The participants were asked to walk in a hallway corridor by placing the phone in select locations as shown above.

To evaluate the two algorithms on detecting steps, we created a dataset of 20 accelerometer signals retrieved by putting a Samsung Galaxy M20 smartphone on four different positions of the body as shown in Table 1. For the leg position, we had collected the data using two orientations, one by putting the screen of the cellphone towards the leg and another by flipping it 180 degrees. The study was conducted in a hallway. Our objective was to compare the step reported by manual counting and the ones estimated by both algorithms. We conducted a series of experiments as described below.

(1) **Step Detection:** The objective of this experiment was to estimate the number of steps by fixing the phone's location and orientation. We choose the front pocket as the location and orientation were such that the phone's screen faces the leg. Firstly, the number of steps was estimated using the baseline algorithm with an experimentally chosen minimum peak height of 3 and a maximum height of 5.5. We fixed the kernel size at 5. We observed a loss of cyclic trend in the signal for a larger kernel size. The hyperparameters chosen in this experiment are fixed for the subsequent experiments, where we consider the accelerometer signal

from different body locations.

We repeated the above process for the peak detection algorithm using Z-scores. We choose a specific value of lag and influence parameter to reduce the mean percentage error. Unlike the baseline algorithm, we choose the best influence parameter for each of the smartphone locations.

(2) **Location sensitivity:** This experiment is similar to experiment 1, but with different locations of the smartphone.
(3) **Orientation sensitivity:** We considered a fixed location (Front Pocket) but with two different orientations.
(4) **Sampling frequency sensitivity:** The objective of this experiment was to observe the change in mean percentage error as we reduce the sampling rate of the data.
(5) **Feature set sensitivity:** We did not use any learning algorithm for this project and relied only on signal processing to deduce the step. Therefore, we performed two sensitivity analyses. First, we fixed the influence parameter to unity, implying that our signal is non-stationary and contains structural break.

4 RESULT

Now we discuss the result obtained using both the algorithm for detecting the step. We performed the first experiment by putting the phone in the users pocket in a fixed orientation. With the baseline algorithm, we achieved a mean percentage error (MPE) of 2.5%. For the adaptive peak detection algorithm, the lowest MPE was 0.71% for a threshold of 6 and an influence of 0.8.

Location sensitivity: Figure 1 shows the result obtained by placing the smartphone at different locations. The lag and influence parameter of the adaptive algorithm was determined for each of the location before computing the result.

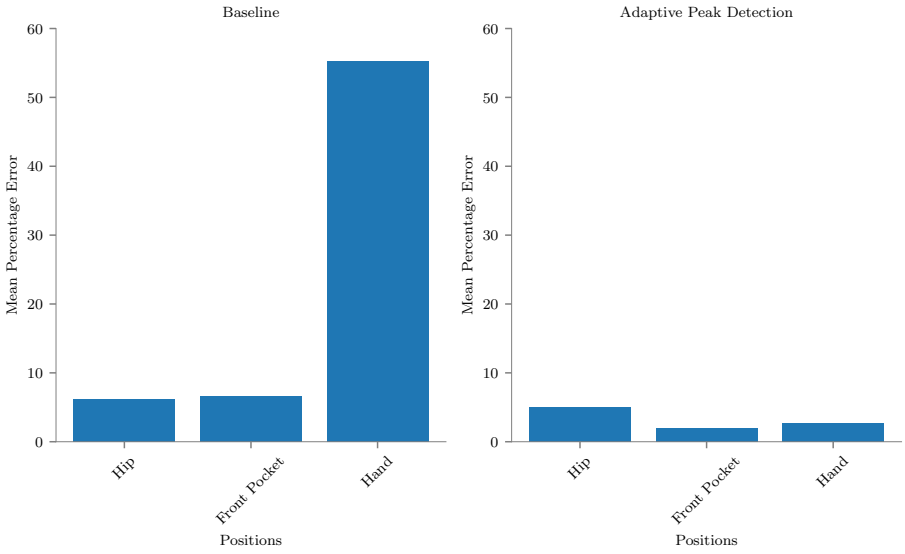

Fig. 1. Adaptive peak detection outperforms the baseline algorithm (on left) on all the three location of the smartphone

Orientation sensitivity: Figure 2 shows the result obtained by placing the smartphone in the front pocket . The adaptive algorithm outperforms baseline in both the orientation.

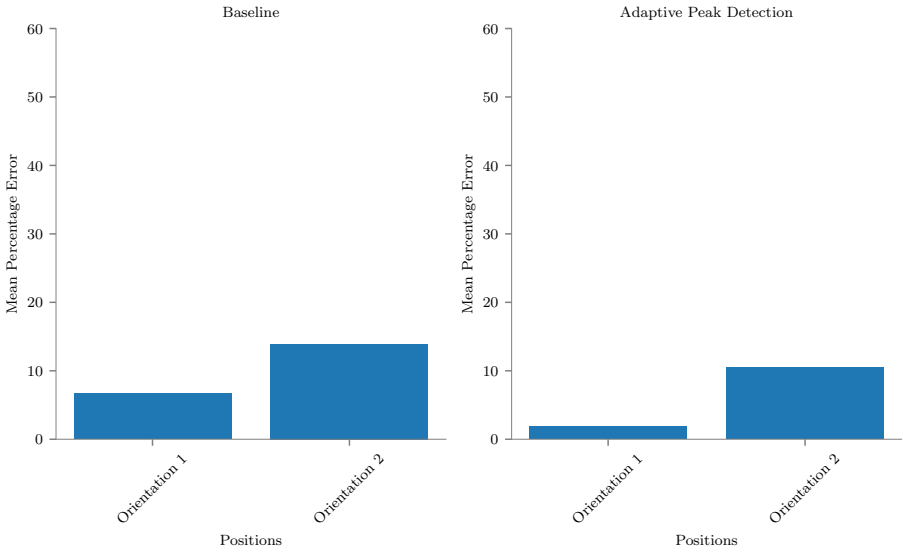

Fig. 2. Adaptive peak detection outperforms the baseline algorithm (on left) on both the orientation of the phone kept inside the front pocket

Sampling Frequency Sensitivity: Figure 3 shows the result obtained by placing the smartphone at different locations for the adaptive algorithm. we observe a drastic increase in MPE as the frequency is changed. This observation could be explained by the change of structural break as data-points are reduced in the signal. We believe that the error can be reduced if the lag and influence parameter are recomputed for the reduced sampling rate.

Feature set sensitivity: Figure 4 shows that the mean percentage error changes from 1.89% to 38% for the front pocket (leg) location if the lag is changed to unity for each location. Figure 5 shows the MPE changes to 22% from 1.89% when running median is considered instead of the running mean.

The above results show that Adaptive step detection can only be used if the smartphone location is known apriori. Generally, the baseline method works better in estimating the steps for all locations with the same set of hyperparamters.

REFERENCES

[1] JP van Brakel. 2014. Robust peak detection algorithm (using z-scores). *Stack Overflow: New York, NY, USA* (2014).

[2] Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. 2020. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature methods* 17, 3 (2020), 261–272.

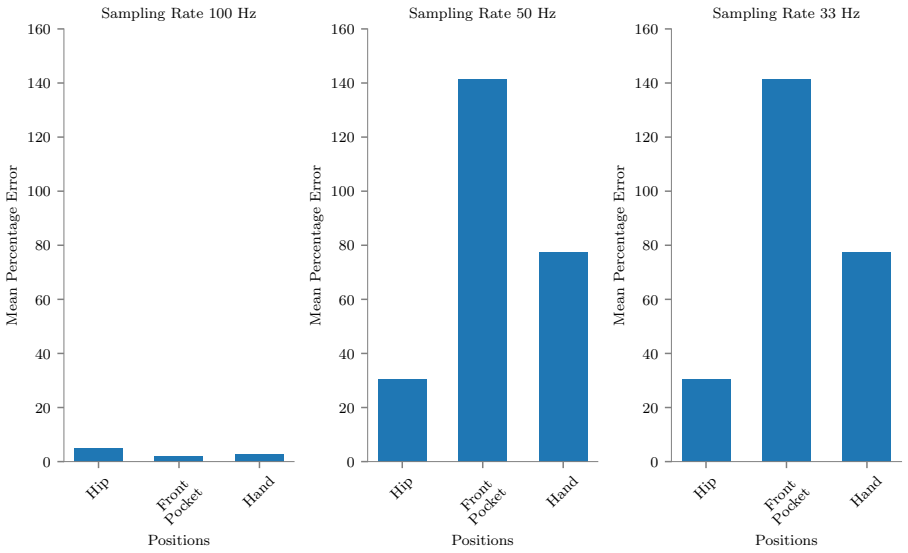

Fig. 3. The high error in reduced sampling rate can be explained by the change of structural break as data-points are reduced in the signal

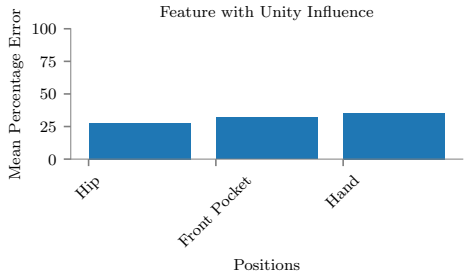

Fig. 4. Change in lag significantly increases the MPE of estimated count.

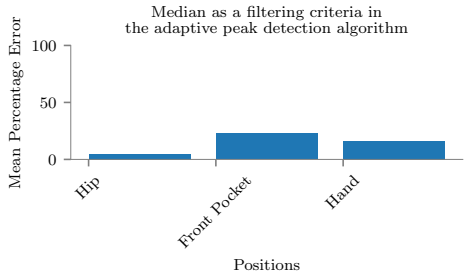

Fig. 5. Moving median cannot be used in place of moving average filter due to increase in MPE.