An Intelligent Session Transition System Towards Low Power Walking Step Estimation

Dipankar Das
Health and Fitness
Samsung R & D Institute India, Bangalore
Bangalore, India
dipankar.d@samsung.com

Prakhyath Kumar Hegde
Health and Fitness
Samsung R & D Institute India, Bangalore
Bangalore, India
prakhyath.h@samsung.com

Vishal Bharti
Health and Fitness
Samsung R & D Institute India, Bangalore
Bangalore, India
visu.bharti@samsung.com

Moonbae Song
Health R&D Group
Samsung Electronics
Suwon, South Korea
moonbae.song@samsung.com

***Abstract*—Activity tracking has gained prominence with the phenomenal growth of fitness devices and ever increasing fitness awareness and step count has emerged as the primary fitness parameter. Step count is typically derived from continuous motion signal analysis. Either a dedicated hardware processing unit or a software signal processing unit, continuously counting repetition in motion signal is the common implementation choice. While dedicated hardware increases the BOM (Bill of Material), the software pedometer incurs high power consumption. In this paper, we proposed a power efficient step estimation algorithm without continuous high power requirement even in motion at an acceptable accuracy using smartphone's tri-axial accelerometer data.**

Keywords—Software pedometer; step count; step estimation; accelerometer; fitness device; smartphone

I. Introduction

A. Motivation

With the recent advent of fitness awareness among users, demand for fitness devices and activity tracking applications are on the rise. Step count has emerged as the primary fitness parameter. But repetitive nature of step counting results in continuous power consumption. A dedicated hardware processing unit solves the problem of higher continuous power need but increases the production cost. So, most of the low-end smartphones employ software pedometers, which in turn increases the average continuous power consumption. In this work, we are proposing a new approach of step count for low-end devices without hardware unit.

B. Prior Work

A lot of studies and research has been done for Software pedometer optimization. However, the majority of pedometer optimization focuses on identification of walk action and limit execution of step count algorithm only to the identified walk episode. Though these approaches reduce the power requirement during non-motion, it still incurs higher power consumption during motion. Haik Kalantarian and Majid Sarrafzadeh demonstrated a foot-mounted energy-harvesting system which uses a beacon to send digital heartbeats to a mobile device for the user's step estimation [1]. For this hardware based experiment setup, an energy harvesting hardware consisting of a piezoelectric transducer, a capacitor bank and an energy harvesting circuitry was used.

The study by Agata Brajdic, Robert Harle evaluated various step detection and counting algorithms [2]. In the work by A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen proposed a normalized auto-correlation (NAR) based continuous step-counting algorithm [3]. A more generalized activity classification using optimized K-Nearest Neighbor (KNN) based clustering is successfully reported by Mustafa Kose, Ozlem Durmaz Incel and Cem Ersoy [4].

Wang Hongman, Zhu Xiaocheng and Chang Jiangbo [5] developed an Android based pedometer using multi sensors setup. In this paper, the authors demonstrated about using real time dynamic threshold for removing invalid wave crest and troughs. As claimed by the authors, using accelerometer and orientation sensors provided better accuracy than the single sensor based configuration. Daisuke Sugimori, Takeshi Iwamoto and Michito Matsumoto [6] presented a 2 step walking authentication system using smartphone accelerometer sensors. In this paper, the authors first detected the mobile phone's position and then classified the pedestrian in the two stage process. In this experiment, 50 data points per 100 cycle for each person was captured for peak and valley as well as for time shifted FFT features. The authors used C4.5 decision tree, Naive Bayes classifier and SVM for comparisons of both device position and pedestrian classification using individual feature amounts.

Though above mentioned systems yielded good accuracy, power efficiency was not considered in those papers.

For efficient training of data, many studies have been done. But in most scenarios, training data is captured offline. These approaches work well for systems which behave in a known manner. The research done by Dan Pelleg. Andrew Moore [7] reported a more generic unsupervised algorithm termed as X-Means having the intelligence of selecting suitable model automatically without prior knowledge of the system behavior.

C. Main Contribution

In this paper, we introduce a power efficient step counter, which estimates count per session. A session is detected through an intelligent pace transition algorithm. In the training mode, we used X-Means algorithm [7] for step pace cluster formation using the derived features and generated a Step-Info map for each cluster instance. Step count values in each Step-Info map are captured using Samsung Health application [8]. Cluster transition is detected and then confirmed by respective features using an optimized K-Nearest Neighbor algorithm [4]. The transition confirmation and cluster identification initiated only when transition is detected, resulting in less drainage of power from the device. It is observed that the efficiency-accuracy trade-off favors the proposed algorithm over the conventional step counters. Lastly, the algorithm provides a gradual increase in accuracy through learning the walking pattern of the user.

II. PROPOSED SYSTEM

A. Data Collection

Two months of data were collected for training purpose from 15 volunteers. All the volunteers were healthy and their ages were between 25 and 33. After an informed consent was obtained, participants were asked to use this data collection setup for a period of 2 months. They were also asked to walk at different paces and different patterns. Volunteers were allowed to keep the smartphone in different places, for example, in the front and back pockets of the trouser, shirt pocket, in hand, in backpack, etc. For Step evaluation during the training phase, we used Samsung Health application [8].

However, to evaluate and test the power consumption of our algorithm, we generated dummy pedometer steps by observing actual patterns. Data from multiple samples were merged together to simulate large real-time data. After collecting all the variants as observed in the real Volunteer, they were merged together in a randomized way to simulate walking/running of the user in real time. The same pattern was repeated for a long interval of time. Thus, more than a million records of data was generated.

We collected 3-axial linear acceleration data without gravity from Samsung galaxy smartphone with the 50Hz sampling frequency.

B. Pre-Processing

We calculated the resultant acceleration (A_{R_i}) as in (1) and acceleration sum ($A_{sum(i)}$) as in (2) from the accelerometer raw data (shown in Fig. 1).

$$A_{R_i} = \sqrt{\left({A_{x_i}}^2 + {A_{y_i}}^2 + {A_{z_i}}^2\right)} \tag{1}$$

$$A_{sum(i)} = A_{x_i} + A_{y_i} + A_{z_i} \tag{2}$$

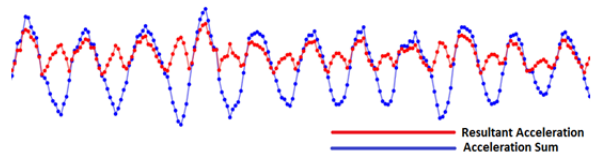

Fig. 1. Pre-processing options.

As in Fig. 1, Resultant Acceleration (A_{R_i}) is always positive, which suits pace transition identification. On the other hand, acceleration sum ($A_{sum(i)}$) shows more distinguishable periodic sinusoidal nature than A_{R_i}, which is favorable for pace evaluation.

C. Feature Vectors Identification

From the pre-processed data we identified our feature vectors based on the experimental result. Fig. 2 shows plot for windowed average and standard deviation. As shown in Fig. 2, while walking at a particular speed, windowed average (W_{Avg}) and Windowed Standard deviation (W_{Sd}) maintains a nearly constant vertical level. However, the figure clearly shows a better transition sensitivity for W_{Avg}, which prompted us to select the same for transition detection. We combined it with Normalized Auto-Correlation (NAR) to strengthen the prediction accuracy.

D. Feature Vectors Derivation

The time domain accelerometer signal is (A_{R_i} and $A_{sum(i)}$) passed through below operations to derive the identified feature vectors.

1) Windowed Average (W_{Avg}): We calculated average for all moving 40 samples present in the 0.8-second window using A_{R_i}value as in (3).

$$WAvg[i] = \frac{1}{Wsize}\sum_{j=i}^{i+Wsize} A_{R_j} \tag{3}$$

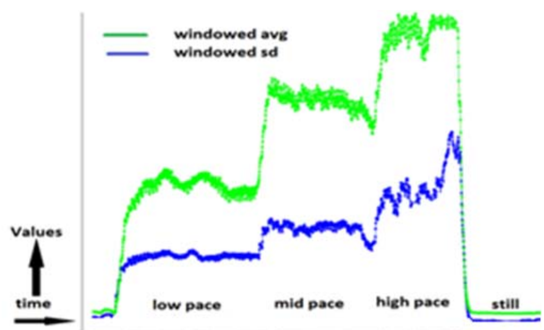

Fig. 2. Experimental data for features identification.

2) Windowed Standard Deviation (W_{Sd}): We calculated W_{Sd} for sliding window of size 0.8 second which shows similar behavior as that of W_{Avg}.

3) Normalized Auto Correlation (NAR): We calculate the NAR for each lag Ʈ at the m[th] samples of A_{sum}. From the autocorrelation values for each lag ranging from 0.8-2 seconds, we calculate the optimum time lag which corresponds to the highest normalized auto-correlation value. This Ʈ opt is the period of the sinusoidal signal of walking pattern. For our experiment, we kept 5 seconds sliding window for measurement for a user moving very slow; and selected the 30^{th} (m) sample.

The normalized autocorrelation is calculated for lag τ at the m^{th} sample as in (4) [3].

$$X(m, \tau) = \frac{\sum_{k=0}^{\tau-1}\left[\begin{array}{c}(a(m+k)-\mu(m,\tau)) \\ (a(m+k+\tau)-\mu(m+\tau,\tau))\end{array}\right]}{\tau\sigma(m,\tau)\sigma(m+\tau,\tau)} \qquad (4)$$

In $\mu(k,\tau)$ and $\sigma(k,\tau)$ are the mean and standard deviation of the samples $< a(k), a(k+1), \cdots a(k+\tau-1) >$ [3].

$$\Psi(m) = \max_{\tau_{min} \le \tau \le \tau_{max}}(X(m,\tau))$$

$$\tau = \tau_{opt} \text{ corresponds to } \Psi(m)$$

E. Personalized Experience and Training Mode

For personalized experience, a first time user will use the system in training mode. The training will gradually capture the personal walking pattern. A standard pedometer algorithm will run in parallel while features are recorded and classified as shown in Fig. 3. This will help us generate a personalized Step-Info map for different pace (low, mid and high) with steps info recorded in Steps/time. This map will be used in real data phase later.

1) Initial Step Pace Cluster Formation using X-Means Algorithm: In our system, we lack any prior knowledge regarding the number of step pace clusters required. We need to use a model which itself optimizes by varying its number of labels. So we shifted our focus towards unsupervised learning algorithm and chose K-Means clustering algorithm as our first choice. But K-Means has major shortcomings in terms of defining an initial number of clusters as well as a dependency on the local minima. On top of these, K-Means algorithm is a relatively slow algorithm due to its iterative nature. There are also several polarization noises of the feature vectors in our definition space. Hence there is a need to splits or shrinks the clusters to arrive at the most cost-effective model. Fortunately, X-Means does not require an initial number of clusters for its operation. So we chose X-Means algorithm over K-Means algorithm.

The algorithm searches the space of cluster locations and number of clusters to optimize the Bayesian Information Criterion (BIC).The BIC information helps select the most cost-effective model using suitable K value. Along with X-Means, we used multi-Resolution Kd-tree for faster convergence towards global minima, by storing sufficient statistical information at each tree node.

In our experiment, we defined the maximum clusters to be formed as 10.

2) Cluster Transition Identification: After clusters are formed from the walking patterns of a user for a month, our next step is to find the steps per unit time for those individual clusters. So, our proposed algorithm continuously identifies the current cluster state from our derived features (which we have calculated per 5 seconds data) using the KNN algorithm.

a) Cluster Identification using KNN Algorithm: After initial clusters are formed, from the received new sample points comprising of our features, the algorithm classifies the label.

For our experiment, we set the K value for KNN as 47. As in optimized KNN [4] search space is pruned for each dimension, the chosen K value is efficient without much computational complexity.

Once the Cluster label is identified, we find the maximum distance ($Dist_{max}$) of that particular cluster from that evaluated sample point.

b) Outlier Identification: As KNN algorithm is always bound to assign the new unidentified data samples to one of the clusters, we enhance the KNN algorithm so that we can identify the outlier points not classifying them to any cluster label. Later using these sample points we build new clusters again.

Below is the method for the outlier identification:

- First the Standard deviation (Sd_{WAvg}, Sd_{Wsd},Sd_{NAR}) for all the features (W_{Avg}, W_{Sd}, NAR) using all cluster instances is calculated.
- The maximum distance of the feature points from its identified cluster is found out. This distance is the maximum of all the other distance contributor of the K votes supporting that particular cluster. These distances are $Dist_{max\_WAvg}$, $Dist_{max\_Wsd}$, $Dist_{max\_NAR}$.
- The identified cluster centroid tuple is (C_{WAvg}, C_{Wsd}, C_{NAR}).
- Now, If $Dist_{max\_WAvg} > C_{WAvg} + 3 * Sd_{WAvg}$ or $Dist_{max\_Wsd} > C_{Wsd} + 3 * Sd_{Wsd}$ or $Dist_{max\_NAR} > C_{NAR} + 3 * Sd_{NAR}$, then Outlier point is detected and store for further processing. Otherwise, the cluster identification is confirmed.

3) Step-Info Map: The step accuracy is most dependent on this chapter of training mode. This section will create a look-up table (Step-Info map) for various pace captured. This Step-Info map will help to calculate step count in real data phase later without much intervention of expensive algorithms. After clusters are built, we start monitoring derived feature vectors to evaluate the cluster where it belongs. When the current label changes, we register the time stamps. We query step counts from Samsung Health database for the duration between two timestamps representing instances when the cluster label has changed. We store this step value in our Step-Info Map for corresponding previous cluster instance.

Assume our cluster label changes at time instance t_1 to C_i and the again transit to another label at time instance t_2, and number of steps from Samsung Health between the time frame t_1 and t_2 is n steps, then Step-Info (SI_{Ci}) is denoted as

$$SI_{Ci} = n / (t_2 - t_1) \qquad (5)$$

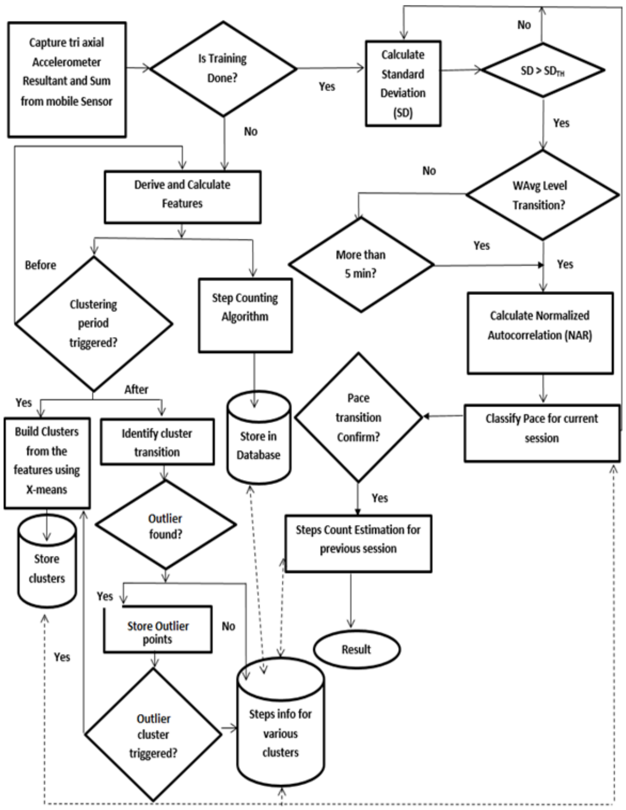

Fig. 3. System block diagram.

F. Real Data phase

During this phase, we compute step counts depending on the walking pace of a user. In this phase, pace transition is continuously monitored from less expensive W_{Avg} value as in (3). If walking pace transition is identified from W_{Avg} value as shown in Table 1, we proceed further to KNN clustering [4] classification algorithm; computing NAR value to confirm and then evaluate the pace from Pace detection data as shown in Fig. 3.

Once walking pace is evaluated from the classification algorithm for current session, we halt classification until next transition. We have kept this step evaluation duration as small as possible for better power efficiency. For these we reduced our search space to (Number_of_cluster)*K values by reducing the K value; where K is the number representing nearest neighbors from each cluster centroids.

TABLE I. STEP PACE TRANSITION RELATION WITH W_{AVG} LEVEL

Windowed Average(W_{Avg})	**Step pace transition category**
1.2 – 1.4 m/s^2	I
1.4 – 1.6 m/ s^2	II
1.6 – 1.8 m/ s^2	III
1.8 – 2.0 m/ s^2	IV
….	

Meanwhile, we compute the steps for the previous session using the Step-Info map generated during training and time duration of previous session. However, we will keep on calculating non-expensive W_{Avg} for further pace transition detection. For better accuracy, we introduced one session validation step (as shown in Fig. 3) if W_{Avg} transition does not happen for longer duration of time.

III. RESULT

Fig. 4 depicts the various steps per time for each cluster. The clusters are nicely formed supporting the diverse movement pace where Step-Info is in Steps/second unit.

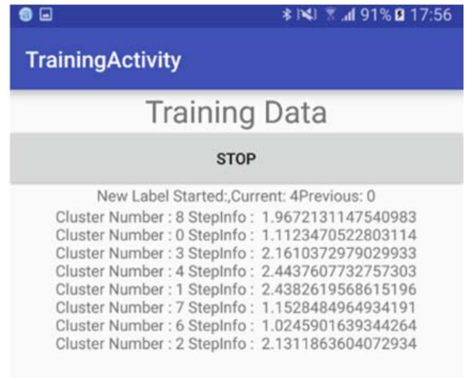

Fig. 4. Representation of clusters centroids and Step-Info for 10 clusters.

Fig. 5 suggests a comparable accuracy of the proposed algorithm with Samsung Health Step Count [8].

The confusion matrix in Table 2 represents a data of 3 hours duration with varied paces. It shows around 98% pace estimation accuracy (average) which results in more than 95% step count accuracy compared to the reference application's step count.

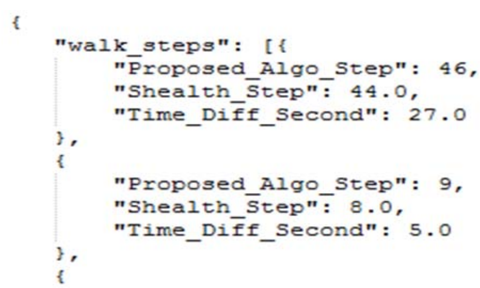

Fig. 5. Experimental comparison result between our proposed algorithm step and samsung health application step count.

TABLE II. WALKING STATE DETERMINATION CONFUSION MATRIX

Targets / Output	**Pace 1**	**Pace 2**	**Pace 3**
Pace 1	99	1	0
Pace 2	2	97	1
Pace 3	0	1	99

TABLE III. POWER MEASUREMENT DURING DIFFERENT STATE OF MOTION

Device Mode	**Power consumption**	
	Conventional Software Pedometer	***Proposed Software Pedometer***
Still	4.0 – 5.0 mAh	4.0 – 5.0 mAh
In transition(1-2 s)	NA	8.0 – 9.0 mAh
In motion	14.0 – 15.0 mAh	4.0 – 5.0 mAh

Table 3 shows three times power saving during Motion in our proposed solution against conventional software pedometer.

For the same time duration, a standard Software pedometer will consume more energy than our proposed algorithm. For example, for 1 hour duration, it will consume around three times more energy than the proposed algorithm when device is in motion.

IV. Conclusion

There are many software pedometer algorithms with good accuracy but very few consider power efficiency which is a concerning factor for smartphones and wearable devices. In this work, we described an intelligent algorithm step estimation using a pace session detection and identification. The efficiency of our algorithm not only brings fitness tracking on low-power, low-cost devices but also enables removal of dedicated hardware processing unit on higher cost devices without compromising much on accuracy.

The proposed algorithm has not been tested enough for the movements like strolling and wheelchair usage. For our future work, we will try to analyze all these patterns to enhance our algorithm's efficiency. Also, if the users change their step pattern very frequently, which is very rare in a real life scenario, duration of the maintained steady state can decrease which may lead to some power consumption. To solve this limitation we are trying to study the pace change pattern on top of the proposed pace detection and identification of the user as our future work.

References

[1] Haik Kalantarian, Majid Sarrafzadeh, "Pedometers Without Batteries: An Energy Harvesting Shoe," IEEE Sensors Journal, Volume: 16 Issue: 23, Dec.1, 2016.

[2] Agata Brajdic, Robert Harle, "Walk Detection and Step Counting on Unconstrained Smartphones," Ubicomp '13 Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing, pp. 225-234, September 2013.

[3] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen. Zee, "Zero-effort crowdsourcing for indoor localization," Mobicom'12 Proceedings of the 18th annual international conference on Mobile computing and networking,pp 293-304, August 2012.

[4] Mustafa Kose, Ozlem Durmaz Incel, Cem Ersoy, "Online Human Activity Recognition on Smart Phones, " Elsevier Journal on Pervasive and Mobile Computing, January 2013.

[5] W. Hongman, Z. Xiaocheng, and C. Jiangbo, "Acceleration and Orientation Multisensor Pedometer Application Design and Implementation on the Android Platform," 1st Int'l Conf. on Instrumentation, Measurement, Computer, Communication and Control, pp. 249-253, 2011.

[6] D. Sugimori, T. Iwamoto, and M. Matsumoto, "A Study about Identification of Pedestrian by Using 3-Axis Accelerometer," 17th Int'l Conf. on Embedded and Real-Time Computing Systems and Applications (RTCSA), pp. 134-137, 2011.

[7] Dan Pelleg. Andrew Moore, "X – means: Extending K-means with Efficient Estimation of Number of Clusters," ICML '00 Proceedings of the Seventeenth International Conference on Machine Learning, pp 727-734, June 2000.

[8] Samsung Electronics Co. Ltd., Samsung Health, https://health.apps.samsung.com.