DePedo: Anti Periodic Negative-Step Movement Pedometer with Deep Convolutional Neural Networks

Wenhua Shao*, Haiyong Luo†, Fang Zhao*, Cong Wang*, Antonino Crivello‡, Muhammad Zahid Tunio*
* School of Software Engineering, Beijing University of Posts and Telecommunications, Beijing, China
†Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
‡Institute of Information Science and Technologies, CNR, Italy
‡University of Siena - Department of Information Engineering and Mathematics, Italy
E-mails: {shaowenhua, yhluo}@ict.ac.cn, {zfsse, wangc, zahid.tunio}@bupt.edu.cn, antonino.crivello@isti.cnr.it

***Abstract*—Pedometer is an enabling technique for smartphone-based pedestrian positioning systems. Because the sensor drifts, these algorithms can only estimate moving distances from step counts. In order to detect step events, researchers have tried to leverage the peak detection and the periodicity attribute of step acceleration signals. However, many human behaviors are having acceleration peaks and periodic, causing traditional detectors error-prone when the phone is shaken periodically leading state-of-the-art system to high false positive ratio and consequently to big mistake of distance estimations. Based on the acceleration feature analysis of step events, we present a deep convolution neural network based step detection scheme to improve the pedometer robustness. Finally, the proposed step detection algorithm is tested in a realistic situation, showing a high anti periodic negative-step movement capability.**

Keywords—context awareness, pedometer, deep learning, smartphone

I. INTRODUCTION

The development of smartphones has opened up many new application fields, where traditionally the use of inertial measurement unit (IMU) has been too costly, or the sensors too bulky. One important application is the pedometer counting the number that a user has walked, which is usually seen in a sport or healthy management software.

A pedometer is also an indispensable module for smartphone-based pedestrian dead reckoning (PDR), which is an important positioning strategy [1, 2]. PDR can reach a high accuracy if the IMU mounted on the shoes because the user moving distance can be calculated through the double integral of acceleration [3]. However, smartphones have little chances to be mounted on shoes. Therefore, positioning researchers first utilize pedometers to detect a step event and then estimate an average step length as the moving distance [4, 5].

Step detection techniques mainly have two classes: foot mounted and hand-held like method. For the foot-mounted case, it is relatively easy to implement step detection by the fact that a step must contain a phase that a foot contacts with the floor for a few seconds, which is also known as zero velocity update (ZUPT) [6]. Therefore, researchers can use acceleration-variance, acceleration-magnitude, or angular rate energy detectors to detect a step event [3].

For the hand-held like case, users tend to hold the phone in their swing hand, bag, or phone call posture. When users walk, phones at these positions have no zero velocity phenomena, so it cannot use the detection method in the foot-mounted case. In order to detect steps, researchers take the advantage of the low frequency of steps, that is, step acceleration signal contains low-frequency peaks for each period [4], and hence step detectors detect these peaks as steps. In order to reduce false alarm rates, step detectors further remove false step peaks by comparing the similarity of several continuous step periods [7-11]. This method reduces the false positive rates for true negative steps, but the initial multi-period comparison brings in several seconds lag for the pedometer first response.

On the other hand, except steps, many human movements are periodic and in low frequency, like shaking hands or legs. Therefore, the abovementioned multi-period comparison has little effect on these periodic movements. Traditional step detectors tend to error-prone again when it comes to periodic negative-step movements. Definitely, for a PDR based positioning system, the two drawbacks will cause the system to be unfriendly to users. For example, the initial lag problem makes the system seems stutter, especially when users frequently stop. On the other hand, the error-prone to periodic movement drawback cause the position result moves when a static user shakes hands.

The main aim of this paper is to improve the robustness of pedometers reducing false negative-step detection especially during periodic movements and to decrease the initial response time.

A robust pedometer is challenging because the step features are different when a user holds the smartphone in different ways. On the other hand, it is necessary to utilize more of these step features to improve pedometer anti-interference ability and decrease the dependency of periodicity. With this purpose, we analyze the steps features and propose deep convolutional neural networks (CNN) approach to learn more of these step features automatically, rather than designing a sophisticated artificial step detection logic.

Another challenge is how to design and to train a CNN for pedometers application. Based on the feature analysis of steps, we adopt acceleration strength as input features, and then design a five-layer pedometer CNN. In order to train the mass network parameters, the learning process needs substantial labeled

training data. Therefore, we present a step-data automatic extraction and labeling method for the network training.

The third challenge lies in the real-time step detection. The proposed pedometer leverages a sliding window to extract real-time input features from smartphone sensors. The abandon of peak detection provides more flexibility to the system but, this operation, also causing multiple positive step outputs when a true step event occurs, because acceleration features around true step-events are similar. Therefore, we consider a filtering method to improve the prediction.

In conclusion, the major contributions of this work include: i) analyzing step features and presenting a pedometer CNN to learn step features automatically; ii) proposing a step data extraction and labeling method to get massive network training data; iii) presenting a filtering method to merge the original continuous multiple CNN outputs to a single step event judgement.

The plan of the rest of the paper is as follows: Section II analyses the acceleration features of steps. Section III presents the proposed step detection solution. It first introduces the system architecture and workflow, and then introduces the true step-scope extraction method. Successively, it proposes the pedometer CNN, introducing its detail design. At last, it explains the real-time CNN results filtering method. Section IV extensively evaluates the proposed scheme. Finally, section V discusses results of the proposed work.

II. STEP FEATURES

This section first analyses the gait cycle then introduces the acceleration features used to detect steps. The following parts reveal the step acceleration patterns at different positions of the body, and finally, the possible interferences from negative-step movements.

Walking is a form of periodic movement between lower extremities. As biomechanics defined [12], the bipedal locomotion can be divided into two phases for each leg: the stance phase and the swing phase, shown in Figure 1. When the left leg is in contact with the ground for restraining, supporting, and propulsion, the right leg is in swing phase for creating a new step forward. The two phases alternate with each other, therefore, moving a person ahead. This alternating action affects other parts of the body, and therefore is reflected by the other parts.

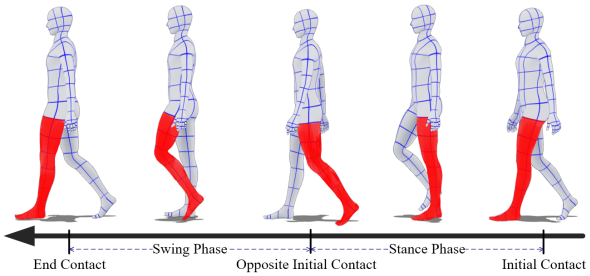

Figure 1. Positions of the legs during a gait cycle. The timeline is for the left leg (red leg).

Generally, smartphones equip with tri-axial accelerometers, an integrated circuit used to measure the accelerations of itself. The output of an accelerometer is a triad vector in the reference of smartphone coordinates. The smartphone's attitude changes when the user moves and consequently the vector direction change. However, the acceleration strength is irrelevant to directions. Therefore, the paper leverages this attitude-free scalar—the strength of acceleration—as feature of steps.

As Figure 2 reveals, the acceleration of a walking foot consists of flat and fluctuation areas, corresponding to the stance and moving phase in the gait cycle. Take the left foot for example: when it contacts the ground at the sample index one, the foot is in a static state, so the acceleration closes to zero. When the following step two starts, left leg swings the left foot, therefore the left foot acceleration changes violently in step two. Noticing that for a pedestrian, his two feet can keep in stance phase simultaneously, but it is impossible to keep both of them in swing phase. Therefore, the paper defines the start of a step as the end of a static phase, and the end of the step as the following start of a static phase.

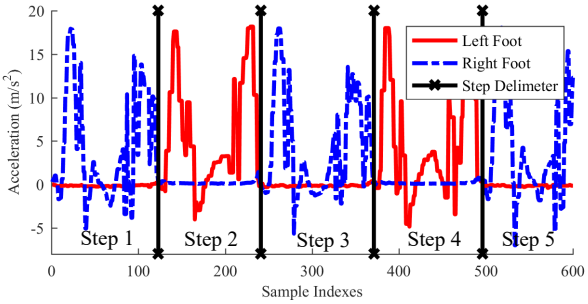

Figure 2. The variation of accelerations for the left and right foot of a pedestrian.

For every position of a user body, the accelerations are periodic when the user is walking, but the acceleration patterns are different between these positions. Smartphones have many carrying manners when a user walks, including putting it into a bag, carrying it in a swing hand, or holding it for sending a message. That is, a smartphone measures different acceleration patterns when it is at different positions. As Figure 3 shows, the figure records the accelerations of three body positions for five steps. For example, acceleration patterns of every step, gathered from a smartphone in a pant pocket, are very similar each other. Furthermore, the pant pocket pattern, swing hand pattern and texting hand pattern are different in the first step cycle.

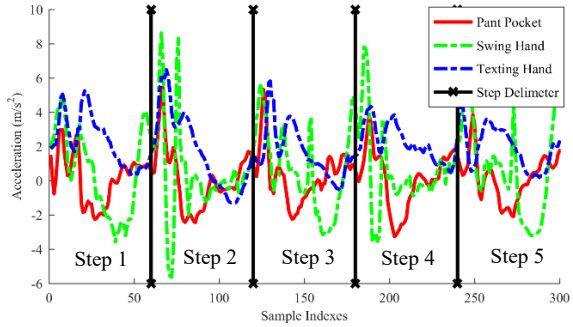

Figure 3. Accelerations at different body positions of a pedestrian. The figure illustrates five steps. The five steps are divided by four step delimiters.

Traditional smartphone based pedometer algorithms mainly rely on the peak detection techniques [7-11]. The basic idea is using a low pass filter to remove high-frequency signal, then detecting peaks as steps. Some researchers further improve detecting accuracy by adding state machines [7], using decision trees to adjust parameters under different poses [8], or using periodic similarity to reduce the short-term negative-step movements [4].

However, except steps, many human movements are pendulum-like. These negative-step motions tend to generate low-frequency acceleration peaks, causing false positive steps detection from traditional algorithms. The motion system of the human being can be abstracted as a series of kinematic chains [13]. Joints connect bones together, and bones limit the body moving scope, just like the pivot and rod of a pendulum. When muscle contracts, it drives relative bones to start a pendulum-like motion, then the motion is stopped when the muscle stops contraction or due to the bone reaches motion limits. This process generates acceleration peaks. For example, in order to answer a phone call, a user moves a smartphone from a table to his ear. The smartphone accelerates from zero velocity to maximum velocity and then returns to zero velocity at the ear. There must be a low-frequency acceleration peak during this process. In order to reduce the false positive rate, some researchers introduce the dynamic time warping (DTW) method. In general, they first detect an acceleration peak and then hold it until a third peak occurs. If the DTW distance between the first and the third peak is low enough, then the first and the third peaks are labeled as true steps [4]. This method reduces the number of false positive steps, but it delays the first step output. On the other hand, this method has little effect on periodic negative-step movements, like shaking hands and legs.

III. The Proposed Solution

This section explains the proposed pedometer based on convolutional neural network in detail. Part A introduces the system architecture and briefly explains how it works, then the following parts detail the key algorithms in the architecture. Part B presents the workflow of the proposed step-scope extraction method in pseudo-code. Successively, part C proposes the pedometer neural network, introducing its structures. Finally, part D proposes the detection-result filtering algorithm.

A. System Architecture

As Figure 4 shows, the smartphone-based pedometer system consists of seven modules: the true steps extraction, feature normalization, acceleration extraction and labeling, and network training module used in the training phase. As well as the sliding window, CNN step detection and result filtering module are used in the online detecting phase. In addition, a sensor module provides basic acceleration signals of the left foot, right foot, and the target position to upper modules.

In the training phase, users collect acceleration signals from the two feet and target positions, then training the pedometer CNN. A target position is a place that you wish the system being able to detect steps, for example, a swing hand, a phone call posture, or a bag. The training-data collection system is shown in Figure 5, including one target-sampling smartphone and two foot-sampling devices. The foot-sampling device consists of one smartphone and a container. The smartphone is fixed into the container, and then the container is tightly fastened to the bottom of the shoes because this place provides the most similar acceleration signal of the foot. When data collection starts, the target position smartphone opens a Wi-Fi access point connection, working as a master. Then the two foot-sampling devices connect to the master, synchronizing with it using the smallest round trip time (RTT). Based on our experiment, the synchronization accuracy is less than 2.5ms, less than the data collection period of 5ms. After data collections phase, the system extracts true steps from the foot-sampling data, then it labels acceleration data of the target position. Because the foot-sampling data and the target-sampling data are synchronous, the acceleration signal segment of one-step in the target position can be confirmed by the step scope of the foot data. Therefore, this signal segment is extracted and labeled with this step, generating a piece of training data. Successively, the network training module trains pedometer neural networks with these labeled training data.

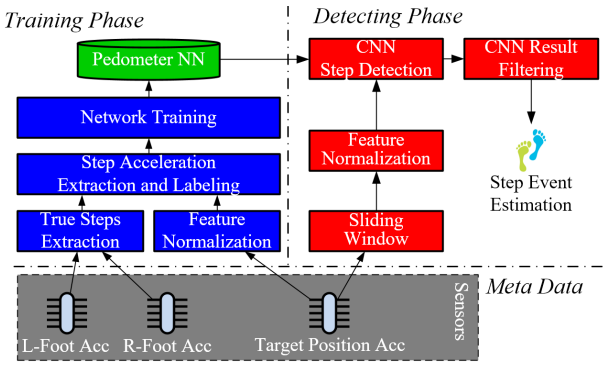

Figure 4. Step detection system architecture. In the sensors module, Acc means a triaxial accelerometer. L-Foot stands for the left foot. R-Foot stands for the right foot. NN means neural networks. And CNN means convolution neural networks.

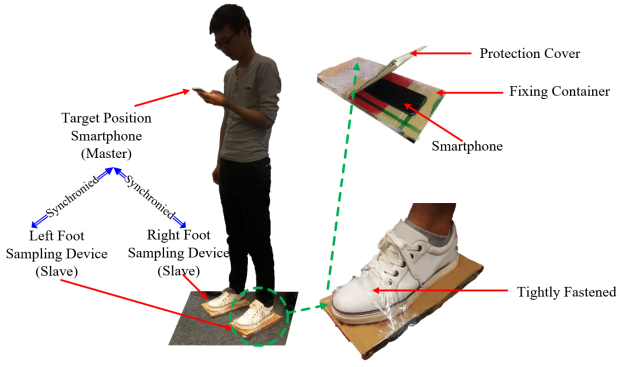

Figure 5. Training data collection system. The figure enlarges a foot sampling device to clarify its working process. In this figure, the target position is a texting hand.

In the online detecting phase, users only collect acceleration signals from target positions. Unlike traditional step detecting methods that select candidate steps by the peak detection technique, the proposed system utilizes a sliding window. The window length is equal to the length of training step segments. The detecting frequency is 20 Hz. Then the CNN step detection module detects the input features and output raw predictions.

Finally, a result-filtering module processes the raw predictions and generate the final output.

B. True Step Scopes Extraction

The proposed system extracts true step scopes from the two foot-sampling devices, so this part presents the algorithm in the true steps extraction module. As Figure 2 shows, a foot acceleration consists of flat and fluctuation areas, therefore, to detect the scope of a step is to find the end of flat acceleration areas as the step start index and start of the next flat acceleration areas as the step stop index. Algorithm 1 details the whole process, it first calculates the moving standard deviations of accelerations, detecting flat areas with small standard deviations, and then finding edges as possible steps. The method further removes too short steps in case of outliers. Algorithm 1 also provides key parameters for our experiment.

Algorithm 1 True Step Scope Extraction Workflow

Initialization
Synchronize sampling devices at the left, right foot and target position.
Collect the tri-axial acceleration vectors of the three devices at 200 Hz. The acceleration at time t of device i is $\boldsymbol{a}_i^t = (a_{ix}^t, a_{iy}^t, a_{iz}^t)$.
Extracting true steps
for each foot device i **do**
  Calculate strength vector $\boldsymbol{s_i}$. For each entry at time t
  $s_i^t = \sqrt{{a_{ix}^t}^2 + {a_{iy}^t}^2 + {a_{iz}^t}^2}$.
  Calculate moving standard deviation vector $\boldsymbol{m_i}$.
  For each entry $m_i^t = \sqrt{\frac{1}{2N}\sum_{j=t-N}^{t+N}(s_i^j - \frac{1}{2N+1}\sum_{k=t-N}^{t+N} s_i^k)^2}$.
  Sliding window length $N = 5$, 25ms.
  Calculate small moving standard deviation index vector $\boldsymbol{v_i}$.
  If $m_i^t < T_m$, set $v_i^t = 1$, else set $v_i^t = 0$.
  Threshold $T_m = 0.4\ m/s^2$.
  Initializing parameters
  Set having found a start entry flag $f = false$.
  Set having found step count $c = 0$.
  Extracting potential step start and end indexes
  for each entry v_i^t in $\boldsymbol{v}$ at time t **do**
    if $v_i^{t-1} == 0$ **&&** $v_i^t == 1$ **then**
      Increase having found step count $c = c + 1$.
      Record this entry into potential step begin vector $\tilde{b}_i^c = t$.
      Set having found a start entry flag $f = true$.
    end if
    if $f == true$ **&&** $v_i^{t-1} == 1$ **&&** $v_i^t == 0$ **then**
      Record this entry into potential step end vector $\tilde{e}_i^c = t$.
    end if
  end for
  Removing too long or too short potential steps.
  Calculate potential step sample counts $\tilde{\boldsymbol{l}}_i = \tilde{\boldsymbol{e}}_i - \tilde{\boldsymbol{b}}_i$.
  for each entry $\tilde{l}_i^t$ in $\tilde{\boldsymbol{l}}_i$ at time t **do**
    if $\tilde{l}_i^t < T_{l\text{-min}}$ || $\tilde{l}_i^t > T_{l\text{-max}}$ **then**
      Remove this entry from $\tilde{\boldsymbol{l}}_i, \tilde{\boldsymbol{e}}_i, \tilde{\boldsymbol{b}}_i$.
      Threshold $T_{l\text{-min}} = 60$, $T_{l\text{-max}} = 260$, that is, 300ms and 1300ms.
    end if
  end for
  Output step lengths $\boldsymbol{l_i}$, step start indexes $\boldsymbol{e_i}$, step end indexes $\boldsymbol{b_i}$. $\boldsymbol{l_i}, \boldsymbol{e_i}, \boldsymbol{b_i}$ are the short steps removed $\tilde{\boldsymbol{l}}_i, \tilde{\boldsymbol{e}}_i, \tilde{\boldsymbol{b}}_i$.
end for

C. Convolutional Neural Networks for Pedometer

As Figure 6 shows, the proposed deep pedometer CNN has four stacked layers: convolution layer, batch normalization (BNorm) layer, rectified linear unit (ReLu) layer, and the fully connected multi-layer perceptron (MLP) [14]. In detail, the convolution layer slides a convolution window along the input acceleration features, generating a new feature map. In the proposed system, this layer has 100 convolution windows, so the layer generates 100 feature maps in the feature maps 1. Then, the BNorm layer normalizes the activations of the previous layer at each batch by applying a transformation that maintains the average activation close to zero and the activation standard deviation close to one, to accelerate network convergence [21]. Successively, the ReLu layer is a nonlinear activation function, which adds non-linearity to the network. It helps in accelerating network convergence and improves network performance [19, 20]. Finally, the fully connected MLP calculates prediction values of all labels. The CNN scheme uses a softmax loss function [15], at end of the stack, to evaluate the error of predictions.

This paragraph introduces the data size of the proposed pedometer neural network. The input feature is an acceleration strength vector with 154 elements, equals to the mean sample data count of one-step, considering the sampling frequency of 200Hz, the mean time of one-step is about 770ms. The convolution window has 20 randomly-initialized parameters. It slides along the input features, calculating convolution values within the window at a stride interval of one. Therefore, the convolution layer outputs feature maps 1 with feature-length of 135.

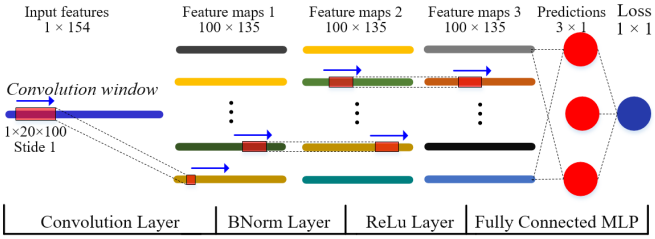

Figure 6. Deep pedometer neural network. BNorm stands for batch normalization. ReLu means rectified linear unit. MLP stands for multi-layer perceptron.

The training data consist of three labels: left step, right step, and negative-step. The training data for left and right steps are acquired by the above-mentioned method, and then be resampled to the length of input features. For the negative-step case, we first ask a user to use the smartphone without walking, collecting negative-step acceleration signals. Then, the system utilizes a fixed-length sliding window to extract negative-step training data. The window length is also the same as input features.

The proposed pedometer CNN has more than 2,000 parameters. The network randomly initializes these parameters and uses backpropagation algorithm [14] for training. When training the network, parameters are tuned.

D. Real-Time CNN Step Detection Result Filtering

In the online detecting phase, the paper utilizes the trained neural network to predict steps against real-time acceleration signals. We utilize a sliding window to extract real-time input features, therefore it generates prediction results at every time the window slides, as Figure 7 reveals. When it nears a true step event, the pedometer CNN outputs a series of discontinuous positive step predictions. The original CNN predictions is confusing to users, so the system filters them before the output.

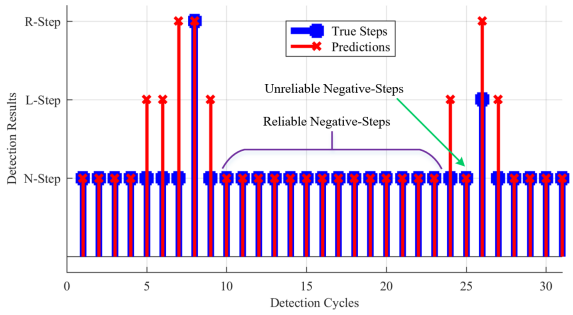

Figure 7. The real-time prediction results from pedometer CNN. The true step event is defined as the time that a foot contacts the ground from a swing phase. L-Step means left step event. R-Step stands for right step event. N-Step means negative-step event. The time between detection cycles is 50 ms.

Algorithm 2 details the filtering algorithm. The system filters the original pedometer CNN predictions basing on the fact that the number of reliable negative-step cycles is much larger than the unreliable ones near step events, as Figure 8 shows. The threshold to distinguish the two type of cycles in our experiment is six. On the other hand, because the features of left step and right step are similar, the system usually confuses the left step and the right step. Therefore, the filtering system merges the two type of events as one type. The system outputs step events immediately when the pedometer outputs a positive step prediction. Then the system keeps outputting negative-step events until it passes a long enough continuous negative-step period.

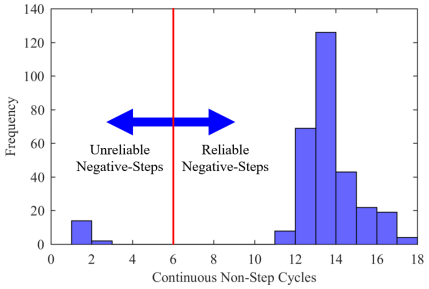

Figure 8. Continuous negative-step cycle count statistics.

Algorithm 2 CNN Step Detection Result Filtering Workflow

Initialization
Start the CNN pedometer, receiving real-time accelerometer strengths.
The pedometer outputs results at 20Hz, outputting $L\text{-}Step$ for the left step, $R\text{-}Step$ for the right step and $N\text{-}Step$ for the negative-step movement.
Set minimum negative-step duration count $MinNDC = 6$, 300ms.
Set step event output ready flag $ReadyForNewStep = False$.
Filtering Results
while true **do**
  Pedometer outputs real-time results $R_i \in (L\text{-}Step, R\text{-}Step, N\text{-}Step)$ at time i.
  if $R_i == N\text{-}Step$ **then**
    Increase continuous negative-step count $C = C + 1$.
    if $C > MinNDC$ **then**
      Set $ReadyForNewStep = True$.
    end if
  end if
  if $R_i == (L\text{-}Step || R\text{-}Step)$ **then**
    Reset negative-step count $C = 0$.
    if $ReadyForNewStep == True$ **then**
      Set $ReadyForNewStep = False$.
      Output a step event.
    end if
  end if
end while

IV. Experiments and Performances

This section exhibits the performance of the proposed step detection method. The section starts with a description of the real-world experimental settings. Then it shows evaluation and performance comparison with traditional pedometer techniques.

A. Instrument Specifications and Experimental Scenario

In our experiments, we collect acceleration signals using five commercially available smartphones, specifications shown in Table 1. We utilize the One X and S4 collecting left and right foot data, the Mate-8 collecting target position data. The S5, P9, and Mate-8 have inherited pedometer software, so we compare our algorithm with them. The smartphones send collected data to a PC server to train the pedometer neural network. Then the smartphone runs the trained neural network to detect real-time steps. The paper trains the network with MatConvNet software in Matlab and then runs online step detection on Android devices with Deeplearning4j, a java open source software. The step count of training data, testing data at each target positions are all 200. Before the experiment, we have trained the pedometer CNN with data from different target positions.

Table 1. Experiment Smartphones

Brand	HTC	Samsung		Huawei	
Series	One X	S4	S5	Mate-8	P9
CPU	1.5G 4 cores	1.6/1.2G 8 cores	2.5G 4 cores	2.3/1.8G 8 core	1.4/1.6G 4 cores
RAM	1 GB	2 GB	2GB	4 GB	4 GB
Function	L-Foot Sampling	R-Foot Sampling	Compare	Target/ Compare	Compare

B. True Step Extraction Algorithm Performance

In order to examine the influence of user's height and gaits, the experiment asks six users to walk 200 steps naturally, wearing the foot-sampling device. Then the system counts detected steps and calculates the correct detection ratio using the number of true step counts divides the absolute value of the difference between true steps and the detected step counts. As Table 2 shows, the true step extraction method achieves high accuracies.

Table 2. Correct Ratios of Different Users on Normal Speed

Users	#1	#2	#3	#4	#5	#6
Gender	F	M	M	M	M	M
Height	1.61m	1.95m	1.74m	1.67m	1.76m	1.69m
Ratio	99%	99.5%	99%	98%	100%	99.5%

C. CNN Pedometer Performance

This part evaluates the performances of the proposed CNN pedometer from various aspects, as well as comparing it with the three algorithms available in three different smartphones. The pedometers in the smartphones need to acquire data from seven steps before they can count the initial step. The proposed method has no such limitations.

True Positive-Step Evaluation of Different Users: Users have different walking speeds and gaits, so the experiment asks six users to walk 200 steps, and then calculates the correct step ratio. As Table 3 shows, although users are of different genders and heights, the true step ratios are approximate. Therefore, the height and gender has little influence to the system performance.

TABLE 3. TRUE POSITIVE RATIOS ON NEW USERS

Users	#1	#2	#3	#4	#5	#6
Gender	F	M	M	M	M	M
Height	1.61m	1.95m	1.74m	1.67m	1.76m	1.69m
TP Ratio	97.5%	99.5%	99.5%	98.5%	98.5%	98%

True Positive-Step Evaluation of Different Target Positions: This experiment asks the user to put the phone in different target positions, including a swinging hand, phone call posture, and a bag. Figure 9 shows the proposed method performs well in the calling and bag case, but poor in the swing hand scenario comparing to other three algorithms. Because the step patterns of a swing hand are more random than the relative static calling and bag case, but we only use 200 piece of training data to train the network for each case. Therefore, we need more training data of swing hands to improve its performance.

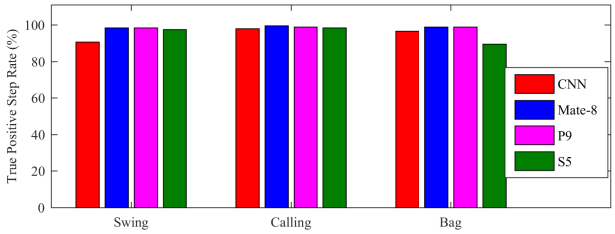

Figure 9. True positive step ratios of different target positions.

False Negative-Step Evaluation: Periodic motion is a serious inference to periodicity-based pedometers. This experiment simulates periodic motion in four manners, horizontal shaking, vertical shaking, mild shaking, and violent shaking. The tester sustains the periodic motions for three minutes, then counting the false positive steps. Figure 10 shows that the proposed CNN pedometer performs rather better than other methods. Its false positive step count is less than a tenth of other pedometers, revealing its strong ability to overcome negative-step periodic motions.

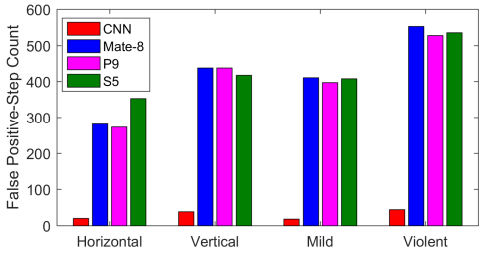

Figure 10. False positive step counts of different negative-step movements.

V. CONCLUSIONS

This paper presents a smartphone based robust pedometer algorithm with the help of deep convolutional neural networks. First, the paper introduces the acceleration features of steps, which is the basis of step detection algorithms. However, the step pattern changes when a smartphone is put in different body positions and traditional pedometers are error-prone to periodic negative-step movements. Therefore, the paper proposes the deep learning based algorithm that learns the step patterns automatically. The paper also presents the foot-aided training data extracting method and the detection area based result-filtering method to train the network and refine the results. Finally, extensive experiments prove the negative-step preventing ability and the accuracy of the proposed scheme.

ACKNOWLEDGMENT

Supported by BUPT Excellent Ph.D. Students Foundation (CX2017404), the National Key Research and Development Program (2016YFB0502004), the National Natural Science Foundation of China (61374214), the International S&T Cooperation Program of China (2015DFG12520), and the Open Project of the Beijing Key Laboratory of Mobile Computing and Pervasive Device.

REFERENCES

[1] X. Wang, X. Wang, and S. Mao, "CiFi: Deep convolutional neural networks for indoor localization with 5 GHz Wi-Fi," in *2017 IEEE International Conference on Communications (ICC)*, 2017, pp. 1-6.

[2] L. Ma, Y. Fan, Y. Xu, and Y. Cui, "Pedestrian dead reckoning trajectory matching method for radio map crowdsourcing building in WiFi indoor positioning system," in *2017 IEEE International Conference on Communications (ICC)*, 2017, pp. 1-6.

[3] I. Skog, P. Handel, J. O. Nilsson, and J. Rantakokko, "Zero-Velocity Detection—An Algorithm Evaluation," *Biomedical Engineering, IEEE Transactions on,* vol. 57, pp. 2657-2666, 2010.

[4] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, and F. Zhao, "A reliable and accurate indoor localization method using phone inertial sensors," presented at the Proceedings of the 2012 ACM Conference on Ubiquitous Computing, Pittsburgh, Pennsylvania, 2012.

[5] Y. Shu, C. Bo, G. Shen, C. Zhao, L. Li, and F. Zhao, "Magicol: Indoor Localization Using Pervasive Magnetic Field and Opportunistic WiFi Sensing," *Ieee Journal on Selected Areas In Communications,* vol. 33, pp. 1443-1457, Jul 2015.

[6] P. H. Veltink, H. B. J. Bussmann, W. De Vries, W. L. J. Martens, and R. C. Van Lummel, "Detection of static and dynamic activities using uniaxial accelerometers," *IEEE Transactions on Rehabilitation Engineering A Publication of the IEEE Engineering in Medicine & Biology Society,* vol. 4, p. 375, 1996.

[7] J. Qian, J. Ma, R. Ying, P. Liu, and L. Pei, "An improved indoor localization method using smartphone inertial sensors," in *Indoor Positioning and Indoor Navigation (IPIN), 2013 International Conference on*, 2013, pp. 1-7.

[8] M. Susi, V. Renaudin, and G. Lachapelle, "Motion Mode Recognition and Step Detection Algorithms for Mobile Phone Users," *Sensors,* vol. 13, pp. 1539-1562, 2013.

[9] W. Kang and Y. Han, "SmartPDR: Smartphone-based pedestrian dead reckoning for indoor localization," *IEEE Sensors journal,* vol. 15, pp. 2906-2916, 2015.

[10] N. Ho, P. H. Truong, and G. Jeong, "Step-Detection and Adaptive Step-Length Estimation for Pedestrian Dead-Reckoning at Various Walking Speeds Using a Smartphone," *Sensors,* vol. 16, p. 1423, 2016.

[11] H. Xie, T. Gu, X. Tao, H. Ye, and J. Lu, "A Reliability-Augmented Particle Filter for Magnetic Fingerprinting based Indoor Localization on Smartphone," *IEEE Transactions on Mobile Computing,* vol. PP, pp. 1-1, 2015.

[12] D. Levine, J. Richards, and W. Mw, *Whittle's Gait Analysis*, 2012.

[13] J. M. Mccarthy, *Introduction to theoretical kinematics*: MIT Press, 1990.

[14] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard*, et al.*, "Backpropagation applied to handwritten zip code recognition," *Neural computation,* vol. 1, pp. 541-551, 1989.

[15] C. M. Bishop, "Pattern recognition and machine learning," *Journal of Electronic Imaging,* vol. 16, p. 049901, 2006.