Multimedia Tools and Applications
https://doi.org/10.1007/s11042-021-11844-6

1215: MULTIMODAL INTERACTION AND IOT APPLICATIONS

Deep-learning-based unobtrusive handedness prediction for one-handed smartphone interaction

Taizhou Chen[1] · Kening Zhu[1,2] · Ming Chieh Yang[1]

Received: 29 March 2021 / Revised: 7 September 2021 / Accepted: 23 December 2021

Abstract

The handedness (i.e. the side of the holding and operating hand) is an important contextual information to optimise the one-handed smartphone interaction. In this paper, we present a deep-learning-based technique for unobtrusive handedness prediction in one-handed smartphone interaction. Our approach is built upon a multilayer LSTM (Long-Short-Term Memory) neural network, and processes the built-in motion-sensor data of the phone in real time. Compared to the existing approaches, our approach eliminates the need of extra user actions (e.g., on-screen tapping and swiping), and predicts the handedness based on the picking-up action and the holding posture before the user performs any operation on the screen. Our approach is able to predict the handedness when a user is sitting, standing, and walking at an accuracy of 97.4%, 94.6%, and 92.4%, respectively. We also show that our approach is robust to the turbulent noise with an average accuracy of 94.6% for the situations of users in the transportation tools (e.g., bus, train, and scooter). Furthermore, the presented approach can classify users' real-life single-handed smartphone usage into left- and right-handed with an average accuracy of 89.2%.

Keywords Handedness prediction · LSTM · Smartphone interaction · Motion sensor · Single hand

1 Introduction

Users generally prefer one-handed interaction with their smartphone whenever possible [20]. One-handed (i.e. unimanual) interaction frees the other hand for parallel activities such as carrying a bag, holding the handrail in the bus, writing a note, etc. However, one-handed interaction with the smartphone usually involves the input of the thumb only. The limited dexterity of the thumb and the increasing size of the touch screen largely affected

✉ Kening Zhu
keninzhu@cityu.edu.hk

Taizhou Chen
taizhou.chen@my.cityu.edu.hk

1 School of Creative Media, City University of Kong Kong, Hong Kong, SAR, China

2 City University of Hong Kong Shenzhen Research Institute, Shenzhen, China

the user performance in many one-handed smartphone interaction, such as typing [13], text editing [57], target selection [34], and so on. In addition, some users tend to use their phones with their non-dominant hands during other passive activities such as eating or drinking [39]. Therefore, designers and researchers have proposed numerous solutions to facilitate one-handed interaction with smartphones, such as user-interface (UI) adaptation [7, 13, 19], specific thumb gestures [9, 21, 34], and input space expansion (e.g., mid-air and back-of-device interaction) [31, 52, 55]. More specifically in UI adaptation for one-handed smartphone interaction, it is suggested that the UI design should accommodate the situations of left- and right-hand interaction which differ from each other largely in terms of thumb reachability [7]. This consideration often results in different UI layouts for left- and right-hand usage. Therefore, the side of the holding and operating hand, which means the handedness, is one of the crucial contextual factors in one-handed smartphone interaction.

Different approaches have been proposed to support the handedness-aware one-handed smartphone interaction. Generally, industrial manufacturers have integrated the feature of switching to one-handed mode in most commercial smartphones. However, all of them require users to manually and explicitly adjust the system configuration with additional operations, such as double tapping the home key in iPhone 8, swiping down at the bottom of the screen in iPhone 11 Pro, swiping diagonally from bottom corner on Samsung S20, etc. There have been a significant amount of research effort focusing on automatic handedness prediction through unlock behaviours [3, 37] and other user input [12, 14, 15, 39, 44]. However, these existing methods perform the prediction still based on users' explicit operations, such as unlocking on the dial-pad interface, swiping, and tapping, thus requiring additional input prior to the actual intended interaction. Among these required operations for handedness prediction, on-screen swiping and tapping may cause unintended input, especially with high-density contents on the screen. The unlocking behavior requires graphical-user-interface (GUI) interaction, which itself may cause the reachability problem for one-handed interaction and require UI adaptation. On the other hand, many latest commercial smartphones (e.g. iPhone 11 Pro, Samsung Galaxy S20, etc.) support the face-scanning unlock mechanism and the feature of "lift to wake". Thus, users can wake up and unlock the phone without any on-screen operation. Besides unlocking which currently may require explicit user input, it is possible for users to perform the action of hand switching during the smartphone usage. There is a need for more implicit/unobtrusive mechanism for handedness prediction in one-handed smartphone usage.

In this paper, we present a deep-learning-based approach of unobtrusive handedness prediction for one-handed smartphone interaction. Leveraging the sensor data obtained by the built-in accelerometer, gravity sensor, rotation sensor, and gyroscope in the smartphone, our approach is able to detect the user's handedness implicitly in real time. More specifically, we train a Long-Short-Term-Memory (LSTM) neural network with the sensor data of 13 users performing normal smartphone activities, such as lifting/picking up the phone, holding the phone still, and operating the phone, etc. The users perform these activities with their left and right hands while sitting, standing, and walking. The statistical analysis on the collected data set showing that the action of phone lifting with the dominant hand leads to larger and faster movements, while the phone-holding posture with the non-dominant hand yields more unstable movements. We then train the LSTM network with the collected data set, and conduct a series of prediction experiments to validate the prediction accuracy under different situations: sitting (97.4%), standing (94.6%), walking (92.4%), and in transportation (94.6%). We also experiment the performance of our approach on the motion-sensor data collected from 6 participants' daily phone-using routine. The result shows an average accuracy of 89.2%. In terms of the inference time, the presented approach can classify a batch of sixteen data samples within 40ms on the phone, to achieve real-time handedness prediction.

This paper makes contributions in three folds:

- We construct the data set containing 989,597 frames of motion-sensor data of one-handed smartphone usage with 13 participants. The statistical analysis on the data set reveals the user-behavioral patterns of smartphone usage by different hands.
- We train the LSTM-based deep neural network trained by the collected data set, for handedness detection in one-handed smartphone interaction.
- We validate the performance of the presented approach with the data of users taking transportation tools and users' real-life smartphone usage.

2 Related work

In this section, we will discuss prior research that facilitate the unimanual smartphone and explore the smartphone-grasping recognition.

2.1 One-handed smartphone interaction

One important motivation of handedness prediction is to facilitate one-handed interaction which is generally preferred by most smartphone users [20]. However, one-handed interaction usually is less accurate, slower, and more difficult than the two-handed interaction due to the the low dexterity of the thumb [1]. Existing solutions to facilitate one-handed smartphone interaction can be classified into three categories:

Adaptive/dynamic GUI As an early work on dynamic GUI, ThumbSpace [19] utilized the thumb-reachable area on the touch screen as the proxy of the whole screen. By dynamically adjusting the touching offset on the soft-keyboard interface based on the user's hand-grasp posture, Goel et al. [13] introduced ContextType, significantly improving the typing performance while users holding the phone with one hand. Buchek et al. [7] proposed a dynamic approach for mobile GUI adaption, aiming to optimize GUI elements within a comfortable reach for one-handed interaction.

Thumb-based gestures With two set of thumb gestures, AppLens and launch-Title facilitated one-hand interaction in early handheld devices [21]. Chang et al. [9] introduced three tilting gestures for one-hand thumb-based GUI target acquisition on large screen. BezelCursor [34] supported bezel-initiated swipe gestures for thumb-based target selection in the further end of a large screen. Li et al. [35] further explored user-defined bezel-initiated thumb gestures for smartphone interaction.

Interaction beyond the screen BeyondTouch [56] leveraged the motion-sensor data to support the back-of-device tap as additional interaction in one-handed mode. Back-Mirror [52] supported a set of unimanual gestures on the back of device with the mirror reflection. HandSee [55] extended Back-Mirror, introduced a camera-based gesture sensing technique by placing a prism mirror on the front camera to extend the interaction space above the screen. InfiniTouch [31] supports the back-of-device interaction using different fingers with the support of extra capacitive sensors around the smartphone.

Our approach implicitly detects users' handedness of signle-hand smartphone interaction in real time. One direct application of our approach is automatically adjusting the GUI layout on the phone without needing the explicit mode change.

2.2 Handedness prediction

Researchers have proposed various solutions on detecting the handedness/posture of smartphone grasping, based on either external hardware or the built-in sensors of the smartphone.

2.2.1 Handedness prediction using external hardware

As an early work, Kim et al. [22] adopted external capacitive touch sensor to detect the hand-grasp posture of a feature phone. Taylor et al. [49, 50] developed a hand-held prototype covered with a matrix of capacitive sensors to infer the hand-grasp posture from the touching signal. Similarly, HandSense [51] detected the unimanual hand-grasp postures through a capacitive-sensor array. Le et al. [30–32] extended HandSense and developed InfiniTouch, a finger- and hand-aware touch-sensing mechanism with the touch-sensor array, supporting handedness prediction.

Researchers have also explored handedness and grasping-posture prediction based on the smartphone data. WhichHand [36] utilized the relative orientation and motion between the smartphone and the smartwatch to determine the handedness while the user is wearing a smartwatch and using a smartphone simultaneously. Kubo et al. [28] conducted an in-depth investigation on cross-device posture recognition between the smartwatch and the smartphone through the built-in accelerometers in both devices. Their prediction model can achieve 94% accuracy. However, such handedness/grasp-prediction methods strongly relied on sensor data from other devices, requiring the smartphone to equip extra external sensors or work along with a smartwatch. Our approach only utilizes the motion data from the built-in sensors in the phone. Our experiments show that it can predict the handedness of single-handed smartphone usage in different environment across different users.

2.2.2 Handedness prediction using built-in sensors

As one of the early works in grasp prediction using built-sensors, GripSense [14] adopted the decision-tree model to classify the gyroscope data, the touch size, and the swipe shape on the screen into different hand-grasping postures, with an overall accuracy of 84.3%. Extending GripSense, Park et al. [44] included additional accelerometer data to train a support-vector-machine (SVM) model for grasp prediction in different situations (e.g., sitting, standing, in room, and in train, etc.), resulting in 87.7% overall accuracy for five postures and 92.4% for four postures. Löchtefeld et al. [37] propose a kNN-based classification method to detect the user's interacting hand (left/right) from his/her unimanual unlocking behaviors, based on the data from the IMU (Inertial Measurement Unit) and the touch screen. Similarly, Guo et al. [15] trained two random-forest models, one to detect the operating hand (left/right) based on the on-screen swipe shape, and the other to detect the hand-changing process based on the IMU data, with the overall accuracy of 94%. Fernandez et al. [12] also experimented a list of supervised classification model for operating-hand prediction based on the on-screen swipe and click data, and found that the AdaBoost model achieved the accuracy of 96%. Nelavelli et al. [39] modeled the screen-unlocking swipe-up trace as a polynomial curve, and adopted polynomial regression to classify the side of operating hand for adaptive user interfaces. Similarly processing the swipe-up gesture for unlocking, Avery et al. [3] presented a dynamic-time warping (DTW) approach to determine the handedness of holding and operating the phone, with 83.6% accuracy. Tan et al. [48] utilized the framework of LSTM neural network to classify the statuses of left-hand and right-hand holding while users are still, and achieved an average accuracy of 98.5%.

While GripSense and the works discussed above offer important insights on utilizing the built-in sensor data for smartphone handedness classification, they still require additional user input, such as swiping and tapping, which are prior to and may not be related to the user-intended operation (e.g., GUI manipulation and target selection). Apart from the touch and the motion data, Kim et al. [23, 24] proposed a grasp-prediction system based on acoustic signal propagated from the built-in speaker to the microphone. However, the acoustic-based method often relies on the signal within a special frequency range, thus occupying the speaker and the microphone for signal transmission and detection. Therefore, it may affect the normal usage of the audio functions in the phone. HandSee [55] installs a right-angle prism mirror on the front camera of the phone, to detect the mid-air gesture above the screen. It is also possible to detect the handheld posture, yet occupying the front camera. In addition, research showed that the concurrent usage of camera, speakers, and microphone in the phone often consume more power than the motion sensors do [8]. In our approach, the handedness of single-handed phone interaction is inferred from the motion-sensor data of users' phone picking and holding actions, without the need of either extra prior input or occupying other input/output components in the phone. To this end, we are inspired by the work of LSTM-based hand posture detection by Tan et al. [48], and differentiates itself by considering different smartphone-usage contexts (e.g., standing, sitting, walking, and in transportation). Furthermore, we collect the real-life user data to evaluate the effectiveness of our approach.

3 Handedness prediction from device motion

In this section, we describe the operational theory behind our approach. We will also describe the process of collecting the user data and the statistical analysis on the data set to support our implementation.

3.1 Operational theory

The presented approach infers the handedness primarily from the phone movement, based on the data of the built-in motion sensors on the phone (i.e. the gyroscope, the accelerometer, the rotation sensor, and the gravity sensor). Everyday one-handed smartphone motion

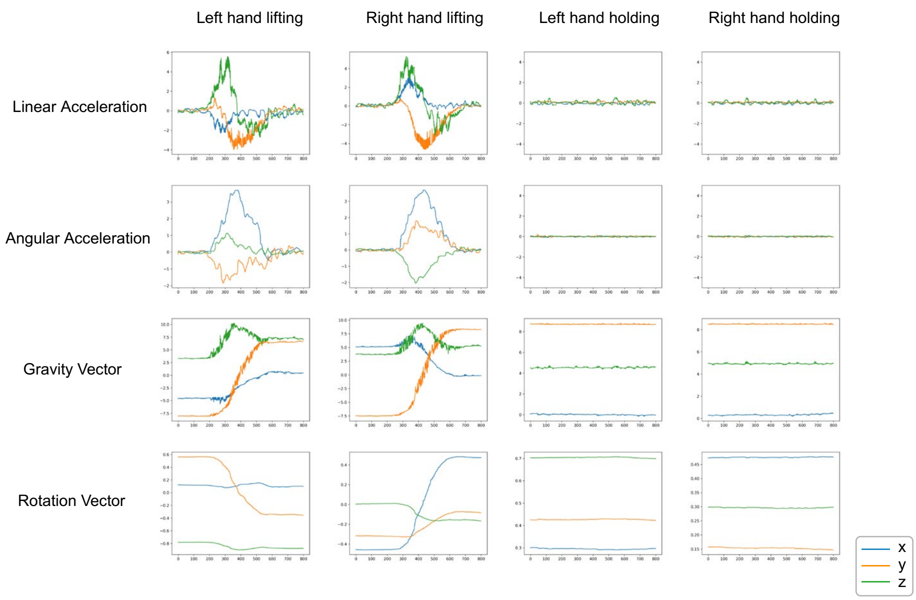

Fig. 1 Visualization of the motion data with user lifting and holding the phone with different hands (from left to right). The linear acceleration, the angular acceleration, the gravity vector, and the rotation vector, were sampled from the accelerometer, the gyroscope, the gravity sensor, and the rotation sensor, respectively

can be grouped into three categories: 1) being picked up; 2) being held still; and 3) being operated, by one hand. Previous research [14] has shown that the device's rotational movements during the user touching the screen could be distinguishable between the left- and the right-hand mode. With the aim of predicting the handedness in real time without extra user input, our approach mainly focuses on the phone movement before the user's on-screen operation, that is, when the user is picking up the phone or holding the phone still in his/her hand.

Picking up the phone either from a flat surface (e.g., table) or from the pocket is the essential prior activity before operating it. As we normally pick up our phone with one hand only, we aims to make a prior handedness prediction based on the user's pick-up motion. Due to the biomechanical principles and the effect of gravity, human kinematic motion always shows symmetries if we perform the motion using different hands [11]. For example, when we pick up the phone with the right hand from the right side, we keep the phone gradually rotate toward our face, from right to left, vice versa. As an example, we plot and visually compare four data sequence samples (i.e. right/left hand lifting, right/left hand holding) from four different sensors (i.e. the accelerometer for the linear acceleration, the gyroscope for the angular acceleration, the gravity sensor for the gravity vector, and the rotation sensor for the rotation vector) collected from one user (Figure. 1). Note that The linear acceleration plot and the angular acceleration plot when picking up the phone with one hand shows the motion symmetrical difference.

When we are holding the phone with one hand, as our arm is stretched out against gravity, the normal physiological tremor will always occur, due to normal breathing, heart beat, and muscle activation [2]. This kind of normal tremor is usually hard to be visible to eye. On the other hand, since the shaking frequency of normal physiologic tremor falls in the

Fig. 2 The single-handed smartphone-usage scenarios during the data-collection procedure. (a) and (b): a user picks up the phone from a table using his right hand and left hand while standing. (c) and (d): a user picks up the phone from the table using his right hand and left hand while sitting. (e) and (f): a user lifts up the phone using his right hand and left hand while standing. (g) and (h): a user lifts up the phone using his right hand and left hand while walking. (i) illustrates different initial positions when the phone is initially laying on the table. For each position, the phone has two initial status: screen facing up and facing down

range of 7-12Hz [38], it is possible to capture such subtle movement with the state-of-the-art in-phone motion sensors in a high sampling frequency (see Figure 1 for the linear-acceleration and the angular-acceleration plots when holding left/right). Research also shows that the statistical characteristics of the normal physiological tremor are significantly different between the right and the left hand [6]. Therefore, although not directly noticeable by eye, it is possible that such "deep" difference in the format of motion-sensor data could be extracted by the deep-learning models and used for handedness recognition.

3.2 Data collection

To construct the data set for handedness analysis and classification, we developed an Android application for capturing the smartphone's motion data, including its linear and angular acceleration, gravity vector, and rotation vector. The application ran on a Huawei Nexus 6P smartphone with an Android OS version 8.1. The sample rate for the gravity sensor, the accelerometer, the rotation sensor, and the gyroscope were set to maximum which are 200Hz, 400Hz, 200Hz, and 400Hz respectively in Huawei Nexus 6P.

The data-collection procedure aimed to simulate a real-life smartphone-usage scenario. The phone was initially laying on the table or being kept in the pocket. User then picked up the phone with one hand (left/right) and turned the screen towards his/her face before operating the phone. The users were asked to repeat this procedure while standing, sitting, and walking (Fig. 2).

13 smartphone users (11 male, 2 female, and average age was 23.3 years old, and all right-handed) were recruited for data collection. To thoroughly cover the real-life scenario, a user needed to pick up the phone using different hands from the left of the table, from the middle of the table (120cm * 40cm), and from the right of the table (Fig. 2(i)), when the phone screen was initially either facing up or facing down. The user was asked to perform three times for each combination of hand (left/right), phone position (middle/left/right), and screen direction (up/down). Thus, there were 2 hands * 3 positions * 2 screen direction

* 3 repetitions= 36 combinations in total, for the scenarios of a user sitting/standing and picking the phone on the table. All the combinations appeared in a random order. In the walking condition, the user was instructed to pick up the phone in the left/right pocket on his/her trousers. Similarly, he/she needed to repeat each action for three times. One may argue that there could be other situations of phone picking, such as picking the phone from the handbag while sitting/standing/walking. In such particular cases, the motion of taking the phone from the bag, especially in the later part of orienting the phone toward his/her face, could yield similar features that can be covered by the scenario of picking the phone from the pocket/table.

The order of sitting, standing, and walking was randomized for each user. Before the each data-recording trial, the user was asked to place the phone at the indicated position as casually as they do in their daily life. During the data-collection process, the user was asked to pick up the phone in the same way as they do daily. To inform the user for phone lifting/picking up, the mobile application signified a beeping sound. During the lifting, the user was instructed not to do any screen touching. In order to simulate a real-life phone usage, the user needed to perform a list of on-screen operations after picking up the phone, including holding, tapping, and swiping, following the icons appearing on the screen in a random order. To indicate the start and the end of a particular action (e.g., lifting, holding, operating), the user needed to press a button on the screen before and after the action. The button-pressing actions were used to indicate the start and the end of the automatic data labelling in the data-collection application. Since the sample rates are different across four different built-in sensors, we aligned the collected data by down-sampling the data collected in the high sample rate to the low sample rate. As a result, we collected 989,597 frames of motion-sensor data in total, including the actions of phone being picked up, held still, and operated by either the left or the right hand, from 13 users. More specifically, there were 92,903 frames for being picked up, 173,870 for being held still, and 229,205 for being operated by the left hand, and 86,078, 169,549, 237,992 by the right hand respectively. Each user contribute 76,123 frames of data averagely. Here, we defined a "frame" of data as one sample from all sensors, namely a 12-dimensional vector containing four types of motion data (i.e., linear and angular acceleration, gravity vector, and rotation vector). The motion-sensor data frames were labeled by the data-collection application at the triggers of user actions, according to the handedness (left or right) and the operation (lifting, holding, or operating).

3.3 Analysis on collected data

Before implementing and experimenting the motion-data-based unobtrusive handedness detection, we performed a series of statistical analysis on the collected data, to investigate the user-behavioral patterns revealed by the handedness of phone holding and lifting. For the data on each axis of each motion sensor, we calculated three dependent variables: 1) the difference between the maximum and the minimum values (i.e., range), 2) the mean value, and 3) the standard deviation (i.e., std) in every 100 frames for each label. These variables are intended to indicate the amplitude and the stability of the user motion. For example, the larger values of range and mean could imply a larger motion, and a smaller std indicates that the motion within the selected time period is more stable and consistent [29].

The repeated measure ANOVA tests are performed on these values with the data label as the independent variable. The results show that the data label significantly affects all the three dependent variables on all the axes for all the motion sensors. Figure 3 shows

(a) Linear Acceleration

(b) Angular Acceleration

(c) Gravity Vector

(d) Rotation Vector

Fig. 3 The range, the mean, and the std of the sensor data on each axis: (a) linear acceleration, (b) angular acceleration, (c) gravity vector, (d) rotation vector. The negative signs indicate the direction of the movement

the descriptive results of the data. Post-hoc pairwise comparisons reveal that for the phone-lifting action, the right-handed data yield significantly larger range, mean, and std than the left-handed data on all the axis of all the sensors (all pairwise comparisons: $p < 0.001$). In addition, Fig. 3 shows the opposite direction of acceleration for the left-handed and right-handed lifting motions. On the other hand, the comparison between the data of left-handed and right-handed holding reveal the opposite. That is, the left-handed holding posture leads to significantly larger range, mean, and std than the right-handed holding posture does for all the sensors (all pairwise comparisons: $p < 0.001$).

Table 1 Comparison of different prediction models

	Left	Right	Overall
LSTM	94.96%	97.45%	96.89%
FC-NN	85.82%	90.51%	88.14%
2D-CNN	80.74%	75.37%	78.34%
1D-CNN	52.63%	78.04%	66.28%
SVM	63.85%	73.51%	66.39%
RF	57.03%	80.07%	65.82%
KNN	59.72%	73.19%	64.32%
LR	52.87%	75.37%	60.22%

Such results may imply that our participants tend to have larger motion of phone lifting while using their right hands, and they have more stable postures when holding the phone with right hands. This could be due to the fact that all these participants are right-handed. Previous physiological research show that the dominant limbs usually maintain higher muscle strength [5], and people can achieve higher speed and larger motion range while using their dominant hands for target reaching [4]. In addition, as discussed above, Beuter et al's studies show that the right-handed persons show different patterns of normal physiological tremor while holding objects with their dominant and non-dominant hands [6], with the dominant hands being more stable. Our collected data set echos with these previous results, revealing that users tend to have larger and faster phone-lifting motion, and more stable holding with their dominant hands. The significant difference between the left-handed and the right-handed phone-lifting/holding further suggests the feasibility of using machine-learning models for unobtrusive handedness detection.

4 Implementation

We first conduct an experiment to compare 8 different machine-learning models on classifying our collected data. According to the experimental result, we design a LSTM-based sliding-window algorithm to process the real-time sensor data. The multi-layer LSTM network is trained on the motion-sensor data set described above.

4.1 Experiment on different machine-learning models

With the data set collected from 13 smartphone users, we experiment the feasibility of detecting the handedness of phone lifting and holding using machine-learning models. More specifically, We experiment 8 machine-learning models for handedness-detection, including LSTM [17], fully-connected neural network (FC-NN) [42], 2D convolutional neural network (2D-CNN) [27], 1D convolutional neural network (1D-CNN) [54], support-vector machine (SVM) [40], k-nearest neighbors (KNN) [10], logistic regression (LR) [53], and random forest (RF) [47], as shown in Table 1. For the structures of the deep-learning-based models, we experiment 2 layers with 32 cells per layer for LSTM, 4 fully connected layers with 10,000, 5000, 1000, and 100 units per layer respectively for the fully-connected dense network, 3 convolution layers with filter number 64, 128, 256 and 2D kernel size 7, 5, 3 respectively for 2D-CNN, and the same architecture as 2D-CNN but 1D kernel for

1D-CNN. Note that we didn't consider the common models for image-based object recognition, such as VGG [45] and ResNet [16], since these network models couldn't directly process our time-series data without any format conversion which may affect the data-feature representation. In addition, in order to enable the potential implementation on mobile devices, we designed and compared a series of relatively small models, such as LSTM model with 1,156,612 parameters, 1D-CNN with 1,278,724 parameters, and 2D-CNN with 1,258,596 parameters, instead of the common image-processing models with much larger numbers of parameters.

4.1.1 Training

The models were trained and tested on a Desktop PC with one GTX 1080Ti NVIDIA GPU, 32GB RAM, and one Intel i7-8700 CPU. We adopted the leave-one-user-out training scheme for the experiments. While we mainly focus on the hand-lifting action and the holding-still posture, we included the data of on-screen operation in the training process, to achieve a more general classification model. More specifically, as the users may hold phone still some times, such as in between two actions. However, it is challenging to extra these data from the collected raw data of phone operation. Therefore, we label the data of on-screen operation the same as the data of holding still for training. Based on the labels created by the data-recording mobile application, we further label the data with four classes: left-hand pick up, left-hand holding/operating, right-hand pick up, and right-hand holding/operating.

Each deep-learning-based model took 100 time steps of motion-sensor data as input, and was trained for 100 epoch at this stage. In addition, we adopt an Adam optimiser [25] ($\beta_1 = 0.9$, $\beta_2 = 0.999$) with learning rate of $1e-5$ to optimise the model, and train with a batch size of 16. During training the deep-learning-based models, we applied the dropout technique [46] with dropout rate 0.5 to avoid over-fitting. For the classic models (e.g., SVM, RF, KNN, and LR), we calculated the maximum and the minimum values, the absolute difference between the maximum and the minimum, the mean value, the standard deviation, the gradient kurtosis, and the skewness on each axis of each motion sensor in a sliding window of 100 time steps for training and testing. For all chosen models, we experimented their performance on different combinations of the four types of motion data, and selected the optimal combination that yielded the best testing performance for each model (i.e., LSTM: The four types of motion data in a 2D-stacked format for each cell in the first layer; 1D-CNN: the 1D concatenation of the four types of motion data in 100 frames/time-steps; FC-NN and 2D-CNN: the 2D stack of the four types of motion data in 100 frames/time-steps; SVM, RF, KNN, and LR: the 84-dimensional handcrafted feature vector).

4.1.2 Performance

The comparison on the model performance (Table 1) shows that the LSTM model outperforms the others in terms of the overall average accuracy of handedness detection with 13 rounds of leave-one-user-out training and testing. Our results of model comparison echo with the previous research on deep-learning-based human-activity recognition which shows the advantage of the LSTM-based model over the standard CNN model on capturing the time-series features in the motion-sensor data [41]. In our case of handedness prediction while picking up or holding, the current hand status (e.g., speed, acceleration, orientation, etc.) is dependent on the previous status, and could sequentially influence the hand status in the next frame. Compared to CNN, LSTM could better extract these time-based sequential features. Thus, we adopt the LSTM framework for our further detailed implementation.

4.2 Model architecture

LSTM network [17] is proposed to capture long-range dependencies in the data sequence, and it have been proved to outperform traditional recurrent neural networks (RNNs) in sensor-signal processing [18, 41]. We implement a multi-layer forward LSTM structure (Fig. 4) with the hyperbolic tangent function as the activation function to process the motion-sensor data. The sensor measurement at a time stamp t, denoted as $S_t = \{a_t, g_t, r_t, \theta_t\}$, contains a_t as the linear acceleration captured by the accelerometer (*Acc*), g_t as the gravity vector captured by the gravity sensor (*Gravity*), r_t as the rotation captured from the rotation sensor (*Rot*), and θ_t as the angular acceleration captured from the gyroscope (*Gyro*). Take the gyroscope *Gyro* as an example, it samples the data of angular acceleration θ_t at a particular time stamp t. θ_t can be represented by $\boldsymbol{\theta}_\mathbf{t}$, a $d^{(Gyro)}$-dimensional vector (e.g., here $d^{(Gyro)} = 3$ for measuring the data on x, y, z axes for the gyroscope). As all the sensor data are sampled synchronously, we concatenate the measurement vectors of all the sensors (i.e., $\mathbf{a_t}, \mathbf{g_t}, \mathbf{r_t}, \boldsymbol{\theta}_\mathbf{t}$) to form a longer measurement vector $\boldsymbol{X} = [a_t, g_t, r_t, \theta_t]^T \in \mathbb{R}^{(d \times 1)}$ where $d = d^{(Acc)} + d^{(Gravity)} + d^{(Rot)} + d^{(Gyro)}$ is the accumulated measurement of all the sensors. $\boldsymbol{X}$ is then taken as the input vector for one cell in the first LSTM layer corresponding to the time stamp t. As time passes with new sensor data streaming in, the new $\boldsymbol{X}$ is formed and fed as the input for the following LSTM cell. In summary, the input matrix for the first LSTM layer can be represented as:

$$\boldsymbol{\Phi} = [\boldsymbol{X}_\mathbf{1}, \boldsymbol{X}_\mathbf{2}, \boldsymbol{X}_\mathbf{3}, \boldsymbol{X}_\mathbf{4}, ..., \boldsymbol{X}_{\boldsymbol{\omega}}] \tag{1}$$

where ω is the number of measurement vectors to be taken as input for the first LSTM layer and also indicates the number of units in the first LSTM layer.

As one trial of phone-lifting/holding (either in real time or pre-recorded) may contain more than ω sets of measurements over time, we applied a sliding-window mechanism on the sensor-data stream with the slide step as 1. The window size can be represented by ω which also denotes the number of measurement vectors to be taken as input for the first LSTM layer. The sliding-window process extracts a series of the input matrices $\boldsymbol{\Phi}$ with $(\omega - 1)$ overlapping time intervals in two consecutive input matrices for the continuous data processing with the LSTM network.

In addition, we consider the handedness-prediction problem as an open-set recognition problem (with the class of one-handed phone usage and the class of non-one-handed phone usage). We design our approach to make prediction continuously on the sequence of sensor data no matter if the phone is being used in the one-handed mode or in other situations, such as being used by two hands, laying still on the table or in the pocket, etc. Therefore, there is a need to recognize the one-handed data and process accordingly, and ignore the non-one-handed data. To this end, we stacked an additional set of LSTM layer with the same structure as the first LSTM layer, followed by a fully connected network with a sigmoid activation to predict if the label is belong to the known one-handed classes (1 - single-handed, 0 - others). For training this part of the network, we collected the same amount of the non-single-handed motion-sensor data of the phone while it being held and used by two hands, being held by one hand and used by the index finger of the other hand, being placed in the pocket while the users are walking, and laying still on the table.

Table 2 Accuracy and inference time under different hyper-parameters. Number in () denotes the average inference time for one batch (16 samples) of 100-frames data on Samsung Galaxy S8+ device, in milliseconds

		Number of Units for LSTM cells		
		16	32	64
Number of LSTM Layers	1	90.21% (28)	94.22% (35)	94.51% (56)
	2	92.87% (35)	95.95% (44)	95.66% (111)
	3	93.02% (47)	96.00% (59)	95.88% (198)

We implemented the model using Python 3.5 with TensorFlow 1.15 framework. We also implemented an Android application by running the trained model on a Samsung Galaxy S8+ android device (Android OS version 8.1) for real-time handedness detection.

4.3 Parameter experiment

To optimize the performance of the LSTM-based network for handedness prediction in single-handed phone usage, we experimented the network structure with different value combinations of hyper-parameters, including the number of LSTM layers, the number of units in each LSTM cell, and the number of time steps (i.e. the side of the sliding window ω). We first test different sets of the layer numbers and the unit numbers, by fixing the number of time steps as 100. Taking the same data-splitting mechanism and train scheme in the initial experiment (Section 4.1), we trained the LSTM models with each set of hyper-parameters for 100 epoch. We measured the prediction accuracy on the validation set as well as the inference time on an android phone. The result (Table 2) shows that the setting with 2 LSTM layers and 32 LSTM units per cell lead to the optimal training accuracy and inference time. We then experimented the effect of different numbers of time steps on the training process, and found that the training accuracy reaches the highest with 100 time steps (80: 93.37%, 100: 95.95%, 120: 95.73%). Therefore, we finalized the LSTM-based neural network as 2 LSTM layers with 32-units LSTM cells and 100 time steps (see Fig. 4 for details) for further training and testing.

4.4 Training

For training and validating the finalized LSTM model, we divided the data collected from the 13 users into the training set and the validation set in the ratio of 8:2, with a random split of 80% for the training set and 20% for the validation set. We first trained a 2-layer stacked LSTM network to classify the recorded data into four single-handed classes. After convergence, we froze the weight of the first LSTM layer, and stacked an additional LSTM block layer on the first LSTM layer to train a binary classifier for open-set classification.

5 Evaluation

We evaluated the performance of our approach with the validation data set which is a random 20% split from the 13-users data, and an additional data set collected from two new users in different transportation tools. Lastly, we investigated the prediction accuracy of

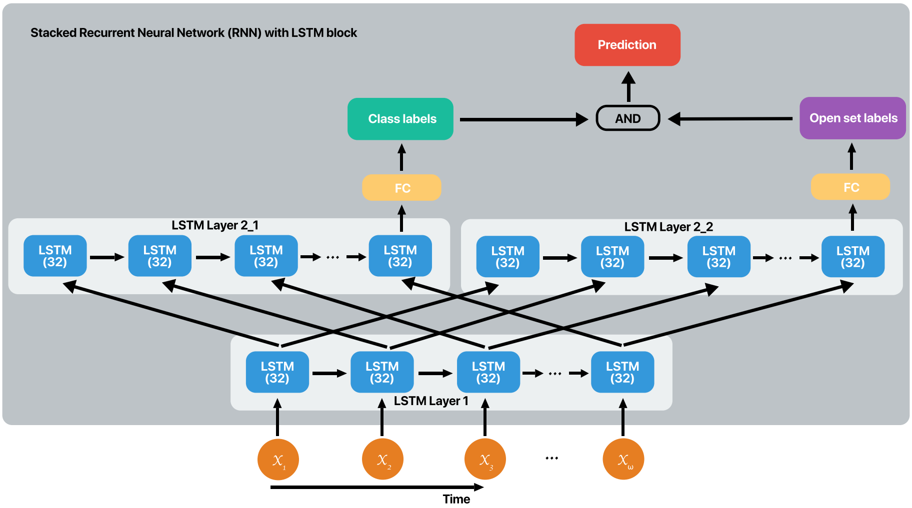

Fig. 4 Data processing pipeline. Dark gray box denotes the architecture of Recurrent Neural Network (RNN) with LSTM blocks that yield two outputs. LSTM Layer 1 and LSTM Layer 2_1 are for class labels prediction. We stack an additional LSTM layer (LSTM Layer 2_2) above LSTM Layer 1 for open set labels prediction. Numbers in () are number of units in the respective layer. We take the bitwise boolean "AND" operation output between "class labels" and "openset labels" as the final output prediction

our approach upon the motion-sensor data caused by users' daily single-handed smartphone usage.

5.1 Validation accuracy

5.1.1 Validation data set

As described above, the validation set consists of 20% of the total data collected from the 13 users. Although we included the data of operating smartphone in the training process, we deleted these data from the validation data set, to evaluate the performance of our approach on the action of picking up and holding the phone. This results in 65,071 100-frames data in the validation set. The offline experiment is done on the same desktop PC used for training.

5.1.2 Accuracy

The validation results show that for open-set classifcation, we can achieve an accuracy of 90.32% of classifying the non-single-handed data (i.e., the open-set data collected in two-handed and cradled usage, with the phone on the table and in the pocket), and 96.93% on the single-handed data. For the four classes of single-handed smartphone usage, we can achieve an overall accuracy of 95.35%. The confusion matrix (Fig. 5) shows that our approach performs more accurately in predicting the handedness of picking up the phone than holding. This could be due to the less motion feature while holding the phone still.

Since we collected the data with the users performing single-handed phone manipulation in the scenarios of sitting, standing, and walking, we are also interested in the

Fig. 5 Confusion matrix of our approach predicting the validation set. Row: the ground truth, Colomun: the predicted label

	Left Pick	Left Hold	Right Pick	Right Hold
Left Pick	94.54%	0.08%	3.61%	1.76%
Left Hold	0.03%	86.45%	0.27%	13.24%
Right Pick	0%	0.04%	99.66%	0.30%
Right Hold	0%	9.09%	0.99%	90.00%

Table 3 Prediction performance in users sitting, standing, and walking

	Left	Right	Overall
Sitting	94.47%	98.69%	97.44%
Standing	92.15%	96.44%	94.64%
Walking	92.51%	92.26%	92.38%

performance of our prediction approach in these three scenarios respectively. Table 3 shows that we can achieve above 90% accuracy in handedness prediction across three scenarios. In addition, we observed that the prediction accuracy decreases as the users shift from sitting to standing, and to walking. This could be due to the increasing level of motion noise introduced by standing and walking.

While our work is closely related to Tan et al.'s research [48], we took the situation of walking into consideration. We further compared our approach with theirs by training the classification model with the sitting and the standing data and leaving the walking data as the test set. The results showed that the LSTM model trained by the sitting and the standing data could only achieve the accuracy of 53% on classifying the walking data. This indicates the comprehensiveness of our dataset and approach.

5.2 Experiment with new users in transportation

While the validation result shows a considerable accuracy above 95% for handedness detection, we note that the training data and the validation data are from the same users, but the data from different users may yield a different result. Therefore, we further test the performance of our approach with a group of different users. In addition, we consider the situation of users walking during the process of data collection, to include the data that may be affected by the turbulent noisy movement. However, there could be many other types of noise in our daily activities, especially while taking

(a) Bus (b) Subway (c) Scooter

Fig. 6 Data collection in transportation

transportation tools (e.g., sitting on bikes or scooters, taking bus or train). The momentum of the moving transportation tools may affect the motion-sensor-based prediction.

To this end, we collected 150,412 frames of motion data from two right-handed users, one female and one male, both 25-years old, who didn't participate in the previous data-collection process. They are asked to perform the same sets single-handed smartphone lifting and holding actions while taking the bus (Fig. 6a) and the subway (Fig. 6b), and sitting on the back seat of the scooter (Fig. 6c).

The prediction results (Fig. 7) shows that we can achieve averagely above 93% accuracy in four-classes single-handedness prediction. In addition, with the collected noisy data, our approach achieves a constant prediction accuracy above 92% for left and right hands in the three different noisy scenarios (Table 4). More specifically, we achieves the highest accuracy in the scenario of scooter (Left: 98.02%, Right: 96.33%). We suspect that this could be because sitting on a scooter introduces more turbulence than in a bus or train, so the users need to hold the phone more tightly, and this may amplify the features for classifying the left and the right hand.

5.3 Experiment with real-life phone-using motion data

While collecting data through a controlled user study is relatively easier to execute as well as more efficient, it could be bias-prone [26] since it couldn't represent a real-life usage scenario. Therefore, we collect the motion-sensor data of everyday smartphone usage, from 6 participants going through their daily smartphone-usage routines, to evaluate the performance of our approach in a real-life usage scenario.

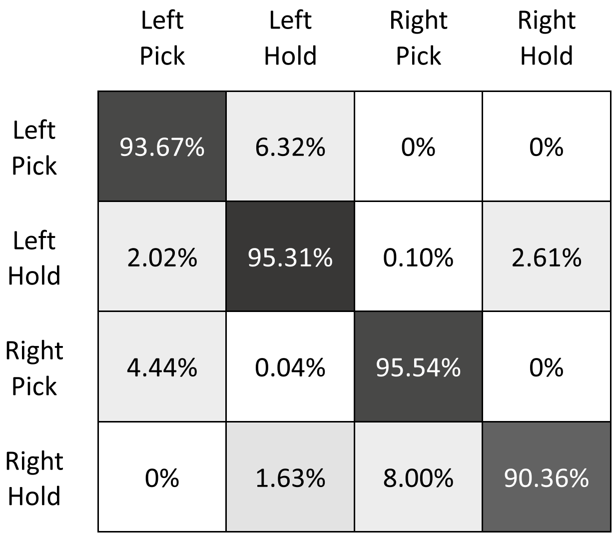

Fig. 7 Confusion matrix of predicting the noisy data. Row: the ground truth, Column: the predicted label

Table 4 Prediction accuracy in three different transportation tools

	Left	Right	Overall
Bus	94.78%	92.34%	93.56%
Subway	92.62%	92.45%	92.54%
Scooter	98.02%	96.33%	97.18%

5.3.1 Procedure

We developed an Android application that run in the background of the smartphone operating system as a service. The service will surreptitiously capture 2 seconds of the phone's motion-sensor data (i.e. the accelerometer, the gravity sensor, the gyroscope, and the rotation sensor) after a random time interval (40% chance of popping up per minute) when the user is using the phone. After capturing the data, the service pops up a graphical user interface (Fig. 8) for the user to label the captured data sequence. The service is also in active in the background when the phone is in the sleep mode (screen off), to detect if the user is picking up the phone. The same interface may pop up after the user picks up and unlocks the phone, in 50% chance. The user can not close the interface unless he/she makes a choice, or the interface will close automatically after 10 seconds if the user doesn't make any selection. Since we only focus on portrait mode in the experiment, we ask the user to choose the "None of above" label if their phone is in landscape mode.

5.3.2 Participants

Through advertising in online social networks and by word of mouth, we recruit 6 voluntary participants (4 male, 2 female) who use Android phones. They live in three different regions globally. Table 5 shows detailed information of the participants. We sent

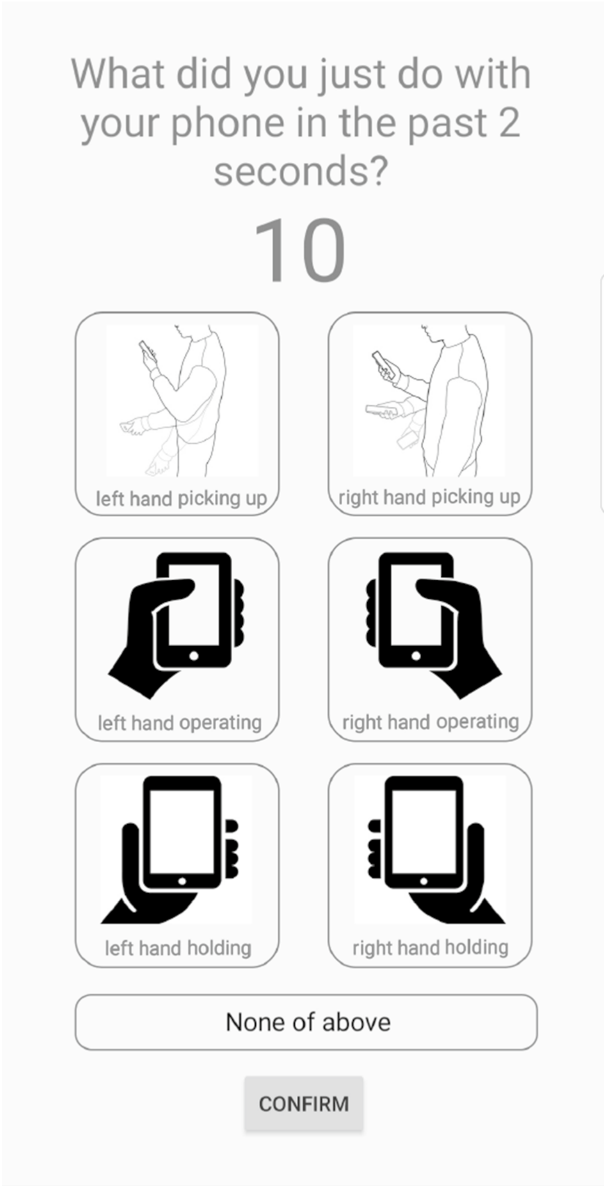

Fig. 8 The labeling interface that will pop up on the user's phone

them the experiment app via Email, asked them to install it on their phones and switch on the app to collect data for one day (10AM - 5PM in the local time). We also asked them add the application into system-optimization white list to prevent it from being killed by the system optimizer. The participants are told to use their phones as normal as they do daily. We also told the participants to select the label as accurately as possible, to avoid mistaken labelling. After the experiment, we instructed the participant through online video conferences to extract the data file from their phones.

Table 5 Participant information for real-life experiment. F denotes female, M denotes male

User	Dominant Hand	Phone Model	Amount of Data (Frame)	Location
P1 (F, 32y/o)	Right	Samsung Galaxy S9+	54,400	Indoor
P2 (M, 26y/o)	Right	Samsung GalaxyS8+	42,400	Indoor, Outdoor
P3 (M, 56y/o)	Left	Samsung Galaxy S8+	36,800	Indoor
P4 (M, 24y/o)	Right	Huawei Nexus 6P	24,000	Indoor
P5 (F, 26y/o)	Right	Huawei Nova 3	34,400	Indoor, Outdoor
P6 (M, 26y/o)	Right	Samsung Galaxy S8	52,800	Indoor, Outdoor

Table 6 Prediction accuracy on the real-life data across 6 participants. "N/A" indicates that there is no data under the particular label from the user

	Left				Right			
	Pick	Hold	Operate	Overall	Pick	Hold	Operate	Overall
P1	N/A	73.45%	89.38%	82.30%	N/A	82.19%	98.69%	94.52%
P2	69.84%	93.11%	92.29%	86.25%	90.56%	N/A	93.04%	92.27%
P3	89.76%	100%	90.37%	95.25%	N/A	N/A	N/A	N/A
P4	N/A	89.26%	92.65%	91.08%	N/A	90.00%	96.54%	92.69%
P5	82.77%	89.22%	91.45%	88.73%	92.83%	79.43%	88.38%	86.33%
P6	84.38%	89.74%	84.33%	85.92%	82.75%	89.47%	82.23%	83.66%

5.3.3 Results

In total, we collected 244,800 frames of data from 6 participants' one-day smartphone usage, including picking up, holding, and operating their phones. For P3 who is left-handed, we didn't receive any data of right-handed phone usage from him. On the other hand, all other participants who are right-handed provide labeled data from both left-hand and right-hand phone usage. In overall, 47% of the data are from the dominant hands, and 53% are from the non-dominant. We didn't receive any report about mistaken labelling from the participants, and we found no pop-up labelling interface was ignored by the participants. We also observe that the amount of data for phone lifting is largely smaller than the amount of phone holding and operating. This could be due to the fact that the duration of phone lifting is usually much shorter than holding and operating in daily smartphone usage, therefore there is a lower chance for the data-collection app to be triggered during the lifting motion.

To evaluate the performance of our approach, we investigated the prediction on left-hand and right-hand phone usage based the labels the participants provided. Table 6 shows the prediction accuracy across different labels and participants. For these participants, We can predict the phone usage in their right hands with an overall accuracy of 89.89%, and 88.26% for the left-handed usage. Specifically for left-handed P4, though our approach is not trained with left-handed users' data at the current stage, it can still achieve overall 95.25% accuracy for his daily left-hand smartphone usage.

6 Discussion

In this section, we will discuss the potential applications of our approach, the possible solutions to deal with failure predictions, and also the limitation and future work.

6.1 Potential application

The most straightforward application of our approach and unobtrusive handedness prediction is automatic GUI adjustment to accommodate different sides of single-handed smartphone usage. Note that we didn't explicitly predict the two-handed usage which is considered as one type of non-single-handed/open-set situations. Our experiments show that the accuracy of classifying single-handed and non-single-handed data is 93.5%, indicating the potential of using our approach for GUI adjustment while the user is using the phone with two hands.

Our unobtrusive handedness prediction could also potentially support adaptive thumb gestures. For instance, in some of the current mobile applications, the function menu could be brought up by swiping from the left bezel. However, this gesture may not be suitable and comfortable for the right thumb in the single-handed mode. Our approach could automatically shift the gesture to swiping from the right bezel when the user is holding his/her phone with the right hand, without explicit mode setting. Another potential application could be toggling different modes, such as general and personal/secure modes, with the handedness of phone lifting. Our approach could offer a design option upon the current common solution of using the fingerprints of different fingers for mode switching[1].

6.2 Correcting failure predictions

One potential issue with any prediction technique is addressing or correcting the errors. Even an overall accuracy around 95% could possibly result in a failure prediction happening once in 20 times. One possible solution is to introduce a larger sliding window to process the prediction results from multiple 100-time-step windows. By adopting the mechanism of maximum voting or top-k classification, the prediction result from the larger window could be stabilized and possibly eliminate the prediction error which may receive lower voting number. According to our empirical test, we can predict a batch of sixteen 100-frames sample data within 40ms on a Samsung Galaxy S8+, thus employing a larger sliding window would place small influence on the real-time response of the system. Another possible solution could be introducing explicit correction operation. For example, under the application of automated GUI adjustment, the user can explicitly touch or swipe on the screen or flip the device for extra motion data when a prediction error happens. Yet, a high accuracy of handedness prediction could reduce the need of explicit error correction.

6.3 Limitations and future work

Firstly, during the evaluation process, we observe that the accuracy of predicting right-handed pick-up/holding is generally higher than predicting left-handed usage in the current

[1] https://www.samsung.com/uk/support/mobile-devices/what-is-the-secure-folder-and-how-do-i-use-it/

stage. This could be possibly because all the training data were collected from right-handed users. It is more challenging to recruit the left-handed persons whose world-wide population is indeed much smaller than the right-handed (10% vs 90%) [43]. It could be more intuitive and natural for the users to perform the actions with their dominant hands. Using the non-dominant hands may lead to fatigue and motion distortion. We plan to expand the data set by collecting data from more left-handed users, and optimize the training and the testing performance of our approach in the future.

Secondly, we didn't explicitly test the performance of our approach in predicting the situation of changing hand or one user passing the phone to another user. The process of changing hand or passing the phone to others is usually followed by a short period of holding or adjusting the screen orientation towards the user. This could be possibly inferred by our approach. As a future work, we plan to evaluate the prediction performance in these situations in a controlled manner.

Thirdly, extensive usage of the motion sensors built in the smartphone can lead to high power consumption. We didn't observe any significant reduction in battery during our empirical test. However, there is a need in the future on detailed analysis and optimization on the energy usage in our approach while running in the background.

Fourth, we observe a difference in the prediction performance across different users in the real-life data experiment. Existing research shows that personal identity can be inferred by users' phone-lifting movement [33]. Therefore, the patterns of picking or holding the phone could alter across different users. This suggests that a personal model for handedness prediction may outperform the current general model. As an important future work, we will investigate the data collection and the training strategy of the personal handedness-prediction model, such as adaptive transfer learning from the general model using minimum amount of personal data.

Last but not the least, in the experiment with real-life motion data, the participants mostly stay in their homes, due to the known global health incident. This results in the fact that most of these real-life data are collected in still postures (e.g., sitting and standing) or slow walking. There are three participants who went outdoor for a short while, but only a small part of their data is collected during walking and jogging nearby their homes. Furthermore, to bother the users' normal phone usage as little as possible, we set the questionnaire to pop up with a 50% chance. This mechanism may miss users' lifting or holding actions some times. To this end, we would like to test the performance of our approach with more diverse real-life data in the future.

7 Conclusion

In this paper, we present an LSTM-based technique for unobtrusive handedness prediction in one-handed smartphone interaction. The LSTM-based neural network is trained upon the motion-sensor data from 13 users' single-handed smartphone picking, holding and operating in the setting of sitting, standing, and walking. Compared to the existing approaches, Our approach doesn't rely on the extra user actions (e.g., on-screen interaction), and predicts the handedness based on the hand-lifting action and the holding posture before the user performs any operation on the screen. Our offline experiments show that we can achieve an average accuracy of 92.6% towards the collected validation data set. In addition, it can detect the handedness under the noisy situations of taking transportation tools, with 94.56% accuracy. Our experiments with the motion data from 6 participants' real-life smartphone usage, shows that our approach can detect the handedness with an average accuracy of 89.2% across different users. It further calls for the investigation of personal prediction model in the future. With these results, we aims to place an important step towards unobtrusive context-aware interaction for smart devices.

Acknowledgements This research was partially supported by the Young Scientists Scheme of the National Natural Science Foundation of China (Project No. 61907037), the Applied Research Grant (Project No. 9667189), and the Centre for Applied Computing and Interactive Media (ACIM) of School of Creative Media, City University of Hong Kong. This work was also partially supported the National Natural Science Foundation of China (Project No. 62172346), and the Guangdong Basic and Applied Basic Research Foundation (Project No. 2021A1515011893)

References

1. Alexander N, Stephen B, John W (2014) Investigating the effects of encumbrance on one- and two-handed interactions with mobile devices. In: Proceedings of the ACM CHI'14 conference on human factors in computing systems
2. Allum J, Dietz V, Freund H (1978) Neuronal mechanisms underlying physiological tremor. Journal of Neurophysiology 41(3):557–571
3. Avery J, Vogel D, Lank E, Masson D, Rateau H (2019) Holding patterns: Detecting handedness with a moving smartphone at pickup. In: IHM 2019 - Actes de la 31e conference francophone sur l'Interaction Homme-Machine, pp 1–7. https://doi.org/10.1145/3366550.3372253
4. Bagesteiro LB, Sainburg RL (2002) Handedness: dominant arm advantages in control of limb dynamics. Journal of neurophysiology 88(5):2408–2421
5. Balogun JA, Onigbinde AT (1992) Hand and leg dominance: do they really affect limb muscle strength? Physiotherapy Theory and Practice 8(2):89–96
6. Beuter A (2000) Physiological tremor: Does handedness make a difference? International Journal of Neuroscience 101(1–4):9–19. https://doi.org/10.3109/00207450008986489, http://www.ncbi.nlm.nih.gov/pubmed/10765987
7. Buschek D, Hackenschmied M, Alt F (2017) Dynamic UI adaptations for one-handed use of large mobile touchscreen devices. In: Lecture notes in computer science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol 10515 LNCS, Springer, pp 184–201, https://doi.org/10.1007/978-3-319-67687-6_13
8. Carroll A (2017) Understanding and reducing smartphone energy consumption (doctoral dissertation, university of new south wales sydney, australia)
9. Chang Y, L'Yi S, Seo J (2014) Reaching targets on discomfort region using tilting gesture. In: UIST 2014 - Adjunct publication of the 27th annual ACM symposium on user interface software and technology, Association for Computing Machinery, Inc, New York, New York, USA, pp 115–116. https://doi.org/10.1145/2658779.2658803, http://dl.acm.org/citation.cfm?doid=2658779.2658803
10. Dudani SA (1976) The distance-weighted k-nearest-neighbor rule. IEEE Transactions on Systems, Man, and Cybernetics **SMC-6**(4):325–327
11. Durgun B (2018) Symmetry in human motion: The secret behind biomechanical principles. In: 19th national anatomy congress & 1st international mediterranean anatomy congress
12. Fernández C, Flores J, Zylich B, González-Rodríguez M, Labrador M, Lanvin DF, Andrés JD (2017) Real-time operating hand detection for the optimization of mobile web interfaces. In: Proceedings - 14th IEEE international conference on mobile Ad Hoc and sensor systems, MASS 2017 pp 520–524, https://doi.org/10.1109/MASS.2017.74
13. Goel M, Jansen A, Mandel T, Patel SN, Wobbrock JO (2013) ContextType: Using hand posture information to improve mobile touch screen text entry. In: Conference on human factors in computing systems - Proceedings, pp 2795–2798, https://doi.org/10.1145/2470654.2481386
14. Goel M, Wobbrock J, Patel S (2012) GripSense: Using built-insensors to detect hand posture and pressure on commodity mobile phones (January 2016), 545, https://doi.org/10.1145/2380116.2380184
15. Guo H, Huang H, Huang L, Sun YE (2016) Recognizing the operating hand and the hand-changing process for user interface adjustment on smartphones. Sensors (Switzerland) 16(8):1–29. https://doi.org/10.3390/s16081314
16. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778

17. Hochreiter S, Schmidhuber J (1997) Long short-term memory 9(8):1–32
18. Huang Y, Kaufmann M, Aksan E, Black MJ, Hilliges O, Pons-Moll G (2018) Deep inertial poser: Learning to reconstruct human pose from sparse inertial measurements in real time. ACM Trans. Graph. **37**(6). https://doi.org/10.1145/3272127.3275108
19. Karlson AK, Bederson BB (2007) ThumbSpace: Generalized one-handed input for touchscreen-based mobile devices. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **4662 LNCS**(PART 1):324–338. https://doi.org/10.1007/978-3-540-74796-3_30
20. Karlson AK, Bederson BB, Contreras-Vidal JL (2011) Understanding one-handed use of mobile devices. Handbook of research on user interface design and evaluation for mobile technology, pp 86–101, https://doi.org/10.4018/978-1-59904-871-0.ch006
21. Karlson AK, Bederson BB, SanGiovanni J (2005) AppLens and launchTile: Two designs for one-handed thumb use on small devices. In: Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '05, Association for Computing Machinery (ACM), New York, New York, USA, p 201, https://doi.org/10.1145/1054972.1055001, http://portal.acm.org/citation.cfm?doid=1054972.1055001
22. Kim KE, Chang W, Cho SJ, Shim J, Lee H, Park J, Lee Y, Kim S (2006) Hand grip pattern recognition for mobile user interfaces. Proceedings of the National Conference on Artificial Intelligence 2:1789–1794
23. Kim N, Lee J (2017) Towards grip sensing for commodity smartphones through acoustic signature. In: UbiComp/ISWC 2017 - Adjunct proceedings of the 2017 ACM international joint conference on pervasive and ubiquitous computing and proceedings of the 2017 ACM international symposium on wearable computers, Association for Computing Machinery, Inc, New York, New York, USA, pp 105–108, https://doi.org/10.1145/3123024.3123090, http://dl.acm.org/citation.cfm?doid=3123024.3123090
24. Kim N, Lee J, Whang JJ, Lee J (2019) SmartGrip: grip sensing system for commodity mobile devices through sound signals. Personal and Ubiquitous Computing pp 1–12, https://doi.org/10.1007/s00779-019-01337-7
25. Kingma DP, Ba J (2014) Adam: A method for stochastic optimization
26. Kopec JA, Esdaile JM (1990) Bias in case-control studies. A review. Journal of Epidemiology and Community Health **44**(3):179–186. https://doi.org/10.1136/jech.44.3.179
27. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp 1097–1105
28. Kubo Y, Takada R, Buntarou S, Takahashi S (2017) Exploring Context-Aware User Interfaces for Smartphone-Smartwatch Cross-Device Interaction. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 1(3):1–21. https://doi.org/10.1145/3130934
29. Laudani L, Casabona A, Perciavalle V, Macaluso A (2006) Control of head stability during gait initiation in young and older women. Journal of Electromyography and Kinesiology 16(6):603–610
30. Le HV, Mayer S, Bader P, Henze N (2017) A smartphone prototype for touch interaction on the whole device surface. In: Proceedings of the 19th international conference on human-computer interaction with mobile devices and services, MobileHCI 2017, https://doi.org/10.1145/3098279.3122143
31. Le HV, Mayer S, Henze N (2018) InfiniTouch: Finger-Aware Interaction on Fully Touch Sensitive Smartphones, pp 779–792, https://doi.org/10.1145/3242587.3242605
32. Le HV, Wolf K, Mayer S, Henze N (2016) Finger placement and hand grasp during smartphone interaction. Conference on Human Factors in Computing Systems - Proceedings **07-12-May-**, 2576–2584. https://doi.org/10.1145/2851581.2892462
33. Lee WH, Liu X, Shen Y, Jin H, Lee RB (2017) Secure pick up: Implicit authentication when you start using the smartphone. In: Proceedings of the 22nd ACM on symposium on access control models and technologies, pp 67–78
34. Li WHA, Fu H, Zhu K (2016) Bezelcursor: bezel-initiated cursor for one-handed target acquisition on mobile touch screens. International Journal of Mobile Human Computer Interaction (IJMHCI) 8(1):1–22
35. Li WHA, Zhu K, Fu H (2017) Exploring the design space of bezel-initiated gestures for mobile interaction. Int J Mob Hum Comput Interact 9(1):16–29. https://doi.org/10.4018/IJMHCI.2017010102
36. Lim H, Lee K, An G, Suh B, Cho Y (2016) WhichHand: Automatic recognition of a Smartphone's position in the hand using a smartwatch. In: Proceedings of the 18th international conference on human-computer interaction with mobile devices and services adjunct, MobileHCI 2016, Association for Computing Machinery, Inc, New York, New York, USA, pp 675–681, https://doi.org/10.1145/2957265.2961857, http://dl.acm.org/citation.cfm?doid=2957265.2961857

37. Löchtefeld M, Schardt P, Krüger A, Boring S (2015) Detecting users handedness for ergonomic adaptation of mobile user interfaces. ACM International Conference Proceeding Series **30-Novembe**(Mum): 245–249. https://doi.org/10.1145/2836041.2836066
38. McAuley JH (2000) Physiological and pathological tremors and rhythmic central motor control. Brain 123(8):1545–1567. https://doi.org/10.1093/brain/123.8.1545
39. Nelavelli K, Ploetz T (2018) Adaptive App Design by Detecting Handedness http://arxiv.org/abs/1805.08367
40. Noble WS (2006) What is a support vector machine? Nature Biotechnology 24(12):1565–1567. https://doi.org/10.1038/nbt1206-1565
41. Ordóñez FJ, Roggen D (2016) Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. Sensors 16(1):115
42. Pal SK, Mitra S (1992) Multilayer perceptron, fuzzy sets, classifiaction
43. Papadatou-Pastou M, Ntolka E, Schmitz J, Martin M, Munafò MR, Ocklenburg S, Paracchini S (2020) Human handedness: A meta-analysis. Psychological Bulletin
44. Park C, Ogawa T (2015) A study on grasp recognition independent of users' situations using built-in sensors of smartphones. UIST 2015 - Adjunct Publication of the 28th Annual ACM Symposium on User Interface Software and Technology (Figure (2)), 69–70. https://doi.org/10.1145/2815585.2815722
45. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556
46. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: A simple way to prevent neural networks from overfitting. J Mach Learn Res **15**, 1929–1958. http://jmlr.org/papers/v15/srivastava14a.html
47. Svetnik V, Liaw A, Tong C, Culberson JC, Sheridan RP, Feuston BP (2003) Random forest: A classification and regression tool for compound classification and QSAR modeling. J Chem Inform Comput Sci 43(6):1947–1958. https://doi.org/10.1021/ci034160g
48. Tan SL, Ng HF, Ooi BY, Tan HK, Ang JLF (2019) Hand posture detection of smartphone users using lstm networks. In: 10th international conference on robotics, vision, signal processing and power applications, Springer, pp 19–25
49. Taylor B, Bove VM (2009) Graspables: Grasp-recognition as a user interface. In: Conference on human factors in computing systems - Proceedings pp 917–925, https://doi.org/10.1145/1518701.1518842
50. Taylor BT, Bove VM (2008) The bar of soap: A grasp recognition system implemented in a multifunctional handheld device. In: Conference on human factors in computing systems - Proceedings, pp 3459–3464, https://doi.org/10.1145/1358628.1358874
51. Wimmer R, Boring S (2009) HandSense - Discriminating different ways of grasping and holding a tangible user interface. In: Proceedings of the 3rd international conference on tangible and embedded interaction, TEI'09, pp 359–362, https://doi.org/10.1145/1517664.1517736
52. Wong PC, Fu H, Zhu K (2016) Back-mirror: Back-of-device one-handed interaction on smartphones. In: SA 2016 - SIGGRAPH ASIA 2016 mobile graphics and interactive applications, Association for Computing Machinery, Inc, New York, New York, USA, pp 1–5, https://doi.org/10.1145/2999508.2999522 , http://dl.acm.org/citation.cfm?doid=2999508.2999522
53. Wright RE (1995) Logistic regression. In: Reading and understanding multivariate statistics., pp 217–244. American Psychological Association, Washington, DC, US
54. Yao S, Hu S, Zhao Y, Zhang A, Abdelzaher T (2017) DeepSense: A unified deep learning framework for time-series mobile sensing data processing. 26th International World Wide Web Conference. WWW 2017:351–360. https://doi.org/10.1145/3038912.3052577
55. Yu C, Wei X, Vachher S, Qin Y, Liang C, Weng Y, Gu Y, Shi Y (2019) Handsee: Enabling Full Hand Interaction on Smartphones with Front Camera-based Stereo Vision. In: Conference on human factors in computing systems - proceedings. Association for Computing Machinery, https://doi.org/10.1145/3290605.3300935
56. Zhang C, Guo A, Zhang D, Southern C, Arriaga R, Abowd G (2015) Beyondtouch: Extending the input language with built-in sensors on commodity smartphones. In: International conference on intelligent user interfaces, Proceedings IUI, vol. 2015-January, Association for Computing Machinery, pp 67–77, https://doi.org/10.1145/2678025.2701374
57. Zhu K, Ma X, Chen H, Liang M (2017) Tripartite effects: exploring users' mental model of mobile gestures under the influence of operation, handheld posture, and interaction space. International Journal of Human-Computer Interaction 33(6):443–459