Detecting a User’s Interacting Hand for Ergonomic Adaptation of Mobile User Interfaces

Markus Löchtefeld, Phillip Schardt, Antonio Krüger
German Research Center for Artificial Intelligence (DFKI)
Saarbrücken, Germany
firstname.lastname@dfki.de

Sebastian Boring
Department of Computer Science
University of Copenhagen
Copenhagen, Denmark
sebastian.boring@di.ku.dk

ABSTRACT
Often, we operate mobile devices using only one hand. The hand thereby serves two purposes: holding the device and operating the touch screen with the thumb. The current trend of increasing screen sizes however, makes it close to impossible to reach all parts of the screen (especially the top area) for users with average hand sizes. One solution is to offer adaptive user interfaces for such one-handed interactions. These modes have to be triggered manually and thus induce a critical overhead. They are further designed to bring all content closer, regardless of whether the phone is operated with the left or right hand. In this paper, we present an algorithm that allows determining the users’ interacting hand from their unlocking behavior. Our algorithm correctly distinguishes one- and two-handed usage as well as left- and right handed unlocking in 98.51% of all cases. This is achieved through a k-nearest neighbor comparison of the internal sensor readings of the smartphone during the unlocking process.

Author Keywords
Handedness, Ergonomics, Adaptive Interfaces, Unlocking, Sensor Fusion

CCS Concepts
•Human-centered computing → Interactive systems and tools;

INTRODUCTION
With the introduction of Apple’s iPhone, capacitive touch-screens became the de-facto standard input on mobile devices. Unlike earlier devices featuring different form-factors and layouts of hardware buttons, touch screens offer – compared to hardware buttons – a flexible layout of information and input. Yet, they also have several drawbacks: besides the lack of proper tactile feedback, their main problem is that the screen and its desired size dictates the size and shape of the mobile device. This is even more amplified with the current trend towards devices with screen sizes of up to six inches (e.g., Google Nexus 6). Operating them comfortably (i.e., reaching all parts of the screen) with one hand while keeping a firm grip is virtually impossible [14].

The problem is that – while holding the device with a steady grip – the thumb is supposed to select objects on the touch screen. This is unique to the interaction with a mobile phones’ touch screens and results in a physical overhead. However, with increased screen sizes, users with average hand-sizes are unable to reach all parts of the display comfortably – unless they change their grip. Yet, the grip is known to have a significant effect on the performance of the user [14]. For this reason, knowing which hand operates the device and the number of hands involved in the interaction is crucial. Figure 1a depicts the problem based on the iPhone 6: reaching the top left corner to initiate a search is impossible without adaptation. This comes at the cost of possibly dropping the device, and will likely affect interaction performance.

With the increase in screen size, manufacturers began to introduce user interface adaptations. For example, the iPhone 6 allows to temporarily shift down the user interface (see Fig-
ure 1b). This adaption, however, has several limitations: (1) half of the user interface is temporarily invisible; (2) after an action is performed, the adaptation has to be invoked again, potentially requiring many actions that are not directly related to the interaction itself – on the contrary, keeping the adaptation **alive** forces users to manually switch it off again; and (3) even with this generic adaptation, users are forced to slightly changing the grip of the device. Despite their limitations and potentially complicated manual process, these adaptations allow for maintaining known affordances.

To take these adaptations to the next level, we propose an algorithm that **automatically** detects whether the user is operating the device with one or two hands as well as with which hand the user holds the device. Our algorithm is based on sensor data (here: accelerometer, device orientation, and touches) during the unlock procedure. It uses a k-nearest neighbour (kNN) comparison with a weighted dynamic time warping (DTW) as distance function. Based on minimal training sets \( n = 5 \) for each of the hand conditions, our algorithm achieves an accuracy of 97.78% for a swipe to unlock, 98.51% for a pattern to unlock and 99.25% accuracy for a PIN unlock.

**RELATED WORK**

Overcoming the limitations of large mobile displays through novel interaction techniques has been a focus of many researchers. Boring et al. allowed touch interaction to be more expressive by incorporating the contact size [4], thereby considering the physiological properties of a user’s hand. Kim et al. developed two new selection techniques for objects out of reach on the device’s screen that had to be triggered by different interactions [9]. A comparable approach was also investigated by Löchtfeld et al., which used the back of the device as input for unreachable objects at the display’s top [11]. Wimmer et al. used capacitive sensors in a prototypical device to detect **how** the user grasps the device [16]. Goel et al. incorporated the accelerometer and gyroscope to detect the grip of the user and – by its change – the meaning of the touch [7]. Our approach considers many of these prototypes in that it uses the device’s sensors as well as the contact size of touch points. Instead of changing the meaning of touch during operation, however, it determines the user’s handedness and the number of hands used to adapt the graphical user interface on an otherwise unmodified off-the-shelf mobile device.

Generating knowledge for creating such user interface adaptations has received great attention recently. Bergstrom-Lehtovirta and Oulasvirta created a predictive model that allows inferring the functional area of the thumb on the touch screen based on the index finger’s position on the back of the device [2]. Similarly, Lee and Choi established the user’s thumb functional area through direct input of the user [10]. They further presented an approach to adapt the device’s home screen towards this functional area. It is, however, only recent that predictive adaptations are used for graphical user interfaces in general rather than simply for on-screen keypads. Machine learning has further been used to create user specific models to increase touch accuracy dramatically [15], which allows for selecting even shrunken UI elements with high accuracy using one hand.

The grip of the device has great influence on the performance of touch input on mobile devices. Mohd Noor et al. investigated how the change of grip on the back of the device can be exploited to predict the touch location [12]. In addition, many approaches exist that allow for detecting the user’s grip of the device [13, 8, 5, 16], yet only a few of those were developed with an application that specifically requires this knowledge [5, 12]. Furthermore, all of these approaches relied on additional capacitive sensors around the device. In contrast, our algorithm does not require such sensors, works on an unmodified mobile device, and requires only a minimal training set that needs to be acquired before first use.

Identifying users based on their touch behavior has lately received great attention in the field of usable security [3, 6], reaching accuracies of 99% for detecting different users. The work closest related to the algorithm presented in this paper has been developed by De Luca et al. [6]. By applying DTW on the touch position and contact size they were able to distinguish different users based on their unlocking behavior. Nevertheless they required the user to always use the same finger and handedness. Instead of identifying users we focus on personalized detection of the handedness of the user, to allow for automatic UI adaptation.

**CONCEPT**

Before considering an adaptation of the user interface, it is of utmost importance to identify whether the user operates the device single-handedly or with two hands. An adaptation does not need to be performed when users operate the device with two hands, yet, when operating the device single-handedly, having knowledge of which hand is holding the device is crucial. Otherwise, the user experience will be diminished drastically [1]. Knowing with which hand the user holds the device would avoid having users manually selecting the best adaptation of the user interface.

To do so, our algorithm collects several sensor readings during the unlocking process – accelerometer, device orientation, and touch points. It compares this data to already recorded datasets where the operating hand was known. For the classification we use kNN, a non-parametric classification method. The input consists of \( k \) training examples in the feature space; the output is a class membership. Thus, each new dataset is classified by a majority vote, where the dataset is being assigned to the class closest to \( k \) nearest neighbors of this class.

Naturally, for such a classification a distance function is needed. This function should consider that users take varying times to unlock their device. Thus, we first apply DTW to allow for comparable sensor data from unlocking procedures of different lengths. Originally, DTW was developed for speech recognition and is well-suited for comparing two time sequences of data. It looks for similarities between the sets and calculates the costs to match them. As a result DTW determines a distance – the **warp distance** – that can be used to determine how similar a set is to the reference set. A warp distance of zero indicates that the two sets are identical. Larger distances denote larger differences between the sets. We chose to use this warp distance as the distance function for our \( k \)-nearest neighbor comparison.
We recruited 12 users (6 female) with an average age of 22.5 years. All of our participants were right-handed and owned a smartphone with similar unlocking applications. The average screen-size of the participants’ devices was 4.3” and they all used one form of the unlocking applications (3 swipe-to-unlock, 4 pattern- and 4 PIN unlock). For the two-handed interaction all participant except one would hold the device in their left hand and interact with the right hand. The users were asked to stand during the study. This was done to emulate a mobile user that stops to, e.g., look up directions.

**DATA ANALYSIS**

For each application, we randomly selected 5 of the 50 trials per condition to serve as ground truth clusters. We apply a kNN ($k = 3$) to select the corresponding clusters for the new data sets that we want to identify. Our distance function for kNN is a weighted sum of the warp distances of the different sensor recordings. For each sensor readings dimension we calculate the warp distance $\theta$ to the corresponding sensor dimension of the ground truth. Overall we had 9 dimensions $\lambda_1, \ldots, \lambda_9$ (Touch x, Touch y, Touch size, accelerometer x, y, z, device orientation yaw, pitch, roll). For each of the sensors dimensions we determined the weights $\kappa_1, \ldots, \kappa_9$ experimentally. Overall the distance function $\delta$ for two data sets $s_1$ and $s_2$ can be formalized as follows:

$$
\delta(s_1, s_2) = \frac{9}{i=1} \kappa_i \theta(s_1(\lambda_i), s_2(\lambda_i))
$$

We implemented both the DTW and kNN in Java to make it directly portable to an Android unlock application. Furthermore we did not alter or filter the sensor data but instead used the raw sensor data of the device. In the following, we present the weights $\kappa_1, \ldots, \kappa_9$ for each application that we found to perform best. For our accuracy analysis, we classify accurately identified hand posture as correct, classification that are wrong but would lead to the normal user interface (two-handed) as neutral, and classifications that would lead to a wrong user interface adaptation as wrong.

**Swipe-to-Unlock**

For the swipe-to-unlock application we found the touch positions to be highly decisive for determining whether the user is operating the device with the left or right hand. One can see in Figure 3 that users would drag the icon to the right when operating the device with the right hand (and to the left when interacting with the left hand, respectively). Furthermore, right-handed and two-handed interaction can be distinguished by the device’s movements and orientation during the unlocking operation.

**Pattern Unlock**

In the pattern unlock application neither the touch positions nor the contact size were of high importance. Yet, we found that the device’s rotation was highly decisive, which makes sense
since users will have to move the device due to its size to reach the corners of the unlock pattern. Our determined weights reflect this (see Table 2). Here, we achieved an accuracy of 98.51% correctly classified postures (266/270 trials), 1 neutral classification (left-handed was identified as two-handed) and 3 wrong classifications. The two patterns did not lead to different results: the first pattern led to one neutral and one wrong classification and the other to two wrong classifications.

As with the pattern unlock, the two different PINs had no wrong classification and the other two wrong classifications. (here: contact size) had a noticeable impact on classification. Furthermore, the device orientation and acceleration were the most important factors. The determined weights (see Table 3) led to an overall accuracy of 99.25% correctly detected postures (268/270 trials). The other two were classified as wrong. As with the pattern unlock, the two different PINs had no impact on the result of classification (we found one wrong classification per PIN).

### Table 2. The weights for the pattern unlock application classification.

<table>
<thead>
<tr>
<th>$K_1$</th>
<th>$K_2$</th>
<th>$K_3$</th>
<th>$K_4$</th>
<th>$K_5$</th>
<th>$K_6$</th>
<th>$K_7$</th>
<th>$K_8$</th>
<th>$K_9$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>1</td>
<td>1</td>
<td>2.5</td>
<td>2.5</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

### Table 3. The weights for the PIN unlock application classification.

<table>
<thead>
<tr>
<th>$K_1$</th>
<th>$K_2$</th>
<th>$K_3$</th>
<th>$K_4$</th>
<th>$K_5$</th>
<th>$K_6$</th>
<th>$K_7$</th>
<th>$K_8$</th>
<th>$K_9$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0.5</td>
<td>2</td>
<td>1.5</td>
<td>1.5</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

### Cross-User Evaluation

For the PIN unlock, we expected the touch points to be the significant factor for different conditions. However, this proved not to be correct. Instead, we found that the touch pressure (here: contact size) had a noticeable impact on classification. Furthermore, the device orientation and acceleration were the most important factors. The determined weights (see Table 3) led to an overall accuracy of 99.25% correctly detected postures (268/270 trials). The other two were classified as wrong.

As with the pattern unlock, the two different PINs had no impact on the result of classification (we found one wrong classification per PIN).

### DISCUSSION & LIMITATIONS

Across all three applications, we achieved an accuracy of 98.51%. From the non-correct classified trials only 6 trials (0.88%) were wrongly classified. Although these might lead to a worse user experience, we would argue that the benefit that could be reached through our algorithm is higher than the possible bad experience the user could face.

Besides the above described kNN classification approach, we also investigated the use of support vector machines. This did not result in better detection rates. We would recommend using kNN which comes at the advantage of being computationally rather inexpensive and thus can easily be integrated into existing mobile devices. We also acknowledge that the data we gathered was collected while standing and we did not cope for potential extra motion that might occur.

The cross-user evaluation will have benefited from the fact that we only had right-handed participants. The results definitively would be worse for a mix of left- and right-handed users. Additionally the hand-size and finger-size between participants was relatively similar with the exception of three female participants that had significantly smaller hands and where responsible for most misinterpreted trials.

### CONCLUSION & FUTURE WORK

In this paper, we presented an algorithm that allows detecting which hand operates the device from a user’s unlocking behavior. Our algorithm is based on an inexpensive kNN comparison of the smartphone’s internal sensor readings during the unlocking process. Our distance function for kNN is a weighted sum of the warp distances of the different sensor recordings. During the unlock-process we are able to identify whether users use one hand (and in this case which one) or both hands with an accuracy of 98.51%.

For future work we plan to investigate the effects of movement on our detection rate in more depth, and thus conduct further data collections while walking. We are aware that our algorithm focused on a static approach (here: determining the operating hand during the unlock procedure). Thus, we plan on extending our detection to continuously determine the user’s interacting hand during usage – as this might change over time without locking and unlocking the device.
REFERENCES


