Enhancing Context-Aware Applications Accuracy with Position Discovery

Khaled Alanezi(✉) and Shivakant Mishra

Computer Science Department, University of Colorado, Boulder, CO 80309-0430, USA
khaled.alanezi@colorado.edu, mishras@cs.colorado.edu

Abstract. Detecting user context with high accuracy using smartphone sensors is a difficult task. A key challenge is dealing with the impact of different smartphone positions on sensor values. Users carry their smartphones in different positions such as holding in their hand or keeping inside their pants or jacket pocket, and each of these smartphone positions affects various sensor values in different ways. This paper addresses the issue of poor accuracy in detecting user context due to varying smartphone positions. It describes the design and prototype development of a smartphone position discovery service that accurately detects a smartphone position, and then demonstrates that the accuracy of an existing context aware application is significantly enhanced when run in conjunction with this proposed smartphone position discovery service.

1 Introduction

Modern smartphones embody a large set of sensors that can be utilized to learn a wealth of information about a user's surrounding environment. Researchers view the availability of such sensors as an opportunity for developing context-aware applications that can provide services tailored for each user's context. Context-aware mobile computing is not a new research topic, for example, a survey paper [4] covering advances in this field was published more than a decade ago. Despite the concept being there for a while, a breakthrough for the number of context-aware applications offered in smartphones application markets (e.g., App Store for Apple iOS or Google Play for Android OS) is yet to happen. For the most part, the current context-aware applications do not meet users' high expectations from technology.

A key problem with current context aware applications is that they typically provide low level of accuracy, particularly when used in an environment different from what was conceived at the application development stage. A major reason leading to low accuracy is the wide variety of ways a user may carry his/her smartphone, henceforth referred to as *smartphone position*. Users carry their smartphones in different positions, e.g. in hand, in purse, in pants pocket, in shirt pocket, etc. Sometimes, their smartphones are in covered positions, in purse or pockets,

I. Stojmenovic et al. (Eds.): MOBIQUITOUS 2013, LNICST 131, pp. 640–652, 2014.
DOI: 10.1007/978-3-319-11569-6_50

while uncovered at other times, watching a video or talking on the phone. Sensor values of different sensors naturally vary based on smartphone position, which in turn impacts the accuracy of the context derived from these values.

The key idea explored in this paper is that the accuracy of context discovery can be significantly improved if the applications rely not only on the sensor values, but also on the knowledge of smartphone positions from which these values are collected. In particular, we propose a new classification methodology as a way to utilize smartphone position information to improve context discovery. During the training period, an application trains multiple classifiers, one for each smartphone position. During operational periods, the application first collects not only the sensor values but also learns the smartphone position using a generic smartphone position discovery service. Next, it chooses a position-specific classifier to discover the context.

We describe the design, implementation and evaluation of a generic smartphone position discovery service. This service utilizes sensor values collected from some carefully chosen sensors and detects smartphone position with a very high accuracy. We propose a two-stage classification method and demonstrate that the accuracy of an existing context-aware application is significantly improved when it is integrated with the proposed smartphone position discovery service following this two-stage classification methodology.

2 Related Work

We focus on studies aimed at providing generic solution for the smartphone position problem. The work in [9] anticipated the importance of body-position knowledge even before the popularity of sensor-equipped smartphones. Their analysis utilized wearable accelerometer sensor to differentiate between four body positions. Another early work [7], limited to on-table, in-hand, and inside-pocket positions, augmented smartphones with a 2D accelerometer and demonstrated an accuracy of 87 %. Some preliminary work to distinguish between the in-pocket and out-of-pocket body-positions is provided in [13] for a smartphone based on the microphone sensor. Good accuracy level of 80 % was achieved. However, this study is also limited considering the number of positions covered. A recent project [6] used accelerometer to detect nine body positions with an accuracy of 74.6 %. It identifies 60 relevant features for body position discovery. We believe that this result can be integrated with our work to increase the number of positions covered and enhance the overall accuracy of position discovery.

The work in [15] suggested the use of a rotation-based approach to recognize four body-positions. The presented solution is based on accelerometer and gyroscope. Achieved accuracy using SVM classification was 91.69 %. We achieve a comparable accuracy while covering a larger set of seven positions. The work in [3] targets body-position localization of wearable sensors used for health monitoring applications. Authors used SVM to achieve a localization accuracy of 87 % when distinguishing between 6 body-positions. The studied positions are typical for health sensors and are not applicable to smartphones. A completely different approach utilizing external multispectral material sensor is proposed in [8]. The solution is based on the idea that smartphone positions are typically correlated with surrounding material with specific features, which the sensor they used is able to detect. We defer from most of above solutions by covering a wide set of positions and by introducing the two-stage classification methodology for context-aware applications to utilize the smartphone position.

3 Position Discovery Service

3.1 Impact of Smartphone Position

Since context derivation starts from raw sensor data collected from smartphone sensors, it is important to understand the impact a smartphone position may have on the sensor values of different sensors. To understand this impact and identify the sensors that can be used to detect smartphone positions, we have conducted an extensive experimental study involving six popular smartphone sensors: accelerometer, microphone, gyroscope, magnetometer, GPS and light sensors. The details of this study are described in a technical report [1], and due to space limitation, we provide only a summary of this study here. The smartphone positions covered in this study and in our position discovery service design are shown in Fig. 1.

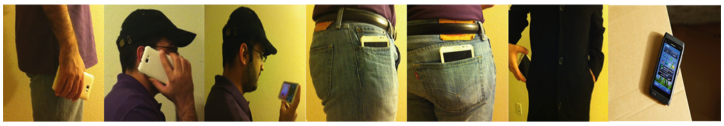


Fig. 1. Inspected positions from left to right, hand holding, talking on phone, watching a video, pants pocket, hip pocket, jacket pocket and on-table.

Table 1 summarizes the results for this study. All sensors, except for the GPS, are affected by the smartphone position. Sensor values of accelerometer, gyroscope and magnetometer are affected by the differences in vibrations at different smartphone positions. So, a context aware application that is based on these sensors values is likely to benefit from the knowledge of actual smartphone position. Light sensor is affected by the blocking of light. Therefore, context-aware applications using this sensor are likely to benefit from the knowledge of whether the phone is covered or uncovered. Finally, the microphone is affected by friction noise and an application utilizing this sensor will likely be interested of whether the phone is in the upper body position where friction is low or in the lower body position where friction is high. Also, this study has shown that a major factor affecting the sensor values besides the smartphone position is the physical context of the user. The sensor values of accelerometer, gyroscope, and magnetometer are affected by the vibrations in case of a walking user. However, these vibrations are missing in case of an idle user. As for running physical context, it can take various paces. A user might run very fast in one situation

Table 1. Summary of smartphone position impact study

Sensor	Affected	Cause	Required position
Accelerometer	Yes	Vibrations	Exact
Gyroscope	Yes	Vibrations	Exact
Microphone	Yes	Friction noise	Upper-body or lower-body
Light	Yes	Blocking of light	Covered or uncovered
Magnetometer	Yes	Vibrations	Exact
GPS	No	-	-

and run relatively slower in another. These different paces make it difficult to find any sensor data patterns from these three sensors that can be attributed to the position since these patterns will be overwhelmed by the effect of a running user. In addition, we looked at the popular position of on-table. Analogous to an idle user carrying a smartphone, the smartphone at this position will not experience any movement patterns. However, a distinguishing factor between the two situations is the orientation of the smartphone. A smartphone placed on a table will typically have the gravity component appearing in its z-axis.

3.2 Position Discovery Service: Design

Based on these observations, we have designed, implemented and evaluated a smartphone position discovery service that provides four types of information: (1) Is the user idle, walking or running? (2) Is the phone covered or uncovered? (3) Is the phone placed in upper body or lower body? (4) What is the actual smartphone position? This service is designed to be configurable, so that an application can choose to receive only one or two or all types of information.

The challenge in building this service is that it utilizes sensor data from specific sensors (e.g. accelerometer and gyroscope), whose values are dependent on the physical contexts of the user. It is possible that the data from a particular sensor under one smartphone position and user activity is indistinguishable from the data from the same sensor under a different smartphone position and user activity. We address this challenge by detecting user's physical context (idle, walking or running) and utilizing data from multiple sensors. The key idea is that different sensors are affected differently by various user contexts, and we exploit these differences to accurately detect smartphone positions.

To detect whether the smartphone is in covered or uncovered position, the service compares the online captured light intensity data with a predefined threshold. The situation is more complex when it comes to the other finer granularity information. For both the upper-body/lower-body and the exact smartphone position decisions, the service uses machine-learning libraries to compare knowledge obtained from online sensor data with knowledge from labeled training data prepared offline. This classification process involves data from accelerometer or gyroscope, or both sensors based on the preference of the serviced context-aware

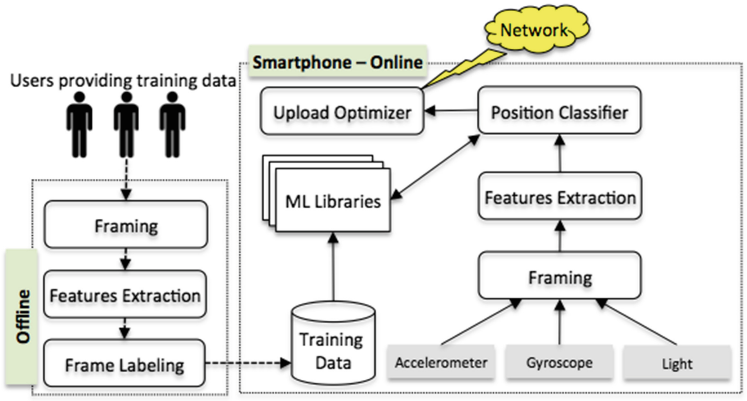


Fig. 2. Design for the smartphone position discovery service

application. Figure 2 illustrates this design. It is worth noting that the complete solution runs on the smartphone. The smartphone position service can be utilized locally by other applications running on the same smartphone or remotely by collaborative sensing applications running on other smartphones.

Offline Components: There are three offline components: Framing, Feature extraction and Frame labeling. The Framing component aims at capturing the repetitive patterns in the raw sensor data by dividing the data stream, from accelerometer and gyroscope, into five-second frames. Our choice of five-second frame size is based on analysis presented in [5] on the effect of frame size on step detection accuracy. Their analysis revealed that a frame size larger than three seconds is sufficient enough to provide good step detection accuracy and favored the five-second frame as it gives more accurate results.

The features extraction component calculates statistical features for each frame. Frame features must be chosen smartly to reveal the different patterns induced by each smartphone position. Our frame features are subset from the features presented in [10]. Based on our observations from smartphone position impact on sensor data [1], we have chosen the mean, variance, and standard deviation over 50 data points (10 data points per second) for each frame to capture the variations in accelerometer and gyroscope data. We have also included two other features related to each axis (average for each axis and average absolute difference of each axis) so as to capture the different orientations a smartphone can take for each position. This is because we noticed that each position has a common orientation that occurs quite frequently. Average of each axis captures the variation in the data due to body motion at the axis level. In addition, it reveals the nature of the orientation the smartphone is experiencing for each body position. Average absolute difference of each axis is the sum of the differences between each axis data point and the mean of that axis divided by the number of data points. We include the average absolute difference to enhance the solution accuracy in capturing the information revealed by axis data points.

A recent study [6] identified an extensive set of features for detecting smartphone positions. These features can be integrated to the position discovery service to expand covered set of positions and further enhance the accuracy. Finally, the frame labeling component labels each frame with the corresponding smartphone position before loading the data to the training database. The labeling process was done manually. We asked our users to capture the data for the different smartphone positions while walking and labeled resultant frame segments with the practiced smartphone position during the experiment. Each frame record carries two labels: upper-body/lower-body position and the exact smartphone position.

Online Components: There are seven online components: Training data, Machine learning libraries, Sensor values from accelerometer, gyroscope and/or light sensor, Framing, Feature extraction, Position classifier and Upload optimizer. The training data is the output from the offline components. Sensor data from ten users performing the same experiment for different smartphone positions was collected offline. After performing the (offline) framing and feature extraction processes, the resultant frame records constitute the knowledge database to be utilized for automatic discovery of smartphone position. Once ready, the training data is placed on the external memory card of the smartphone to be utilized by the smartphone position service.

We utilized Java language machine-learning libraries provided by the WEKA tool for Android [11]. The correctness of used classifiers was tested by performing a test experiment with the same training and test data on a desktop by normal WEKA and on Android device by WEKA for Android. Same results were obtained for the two experiments.

We chose three sensors (accelerometer, gyroscope and light sensor) for the service to operate on. Accelerometer is used for detecting physical context, accelerometer and/or gyroscope are used for detecting the actual phone position, and light sensor is used for detecting whether the phone is covered or uncovered. The use of two sensors for detecting actual phone position is subject to a tradeoff between energy consumption and smartphone position detection accuracy. The framing and features extraction components have the same functionalities as in the offline case. The only addition is the capture of average light intensity per frame, which is not required for training the classifier. The position classifier component receives the gathered online frame data and uses it in three ways. First, it compares the standard deviation of accelerometer magnitude with predefined thresholds to determine idle/walking/running contexts. Second, it consults the machine learning classifier to detect smartphone position information. Third, it compares the light intensity average of the frame with a predefined threshold to determine covered/uncovered position. In the case of idle or running contexts, the position service provides the latest smartphone position discovered under walking context along with a timestamp and leave it to the consuming context-aware application to use this cached smartphone position based on its accuracy preferences. Our goal here is to exploit the fact that, in some situations the user might change their physical context but maintain the same smartphone position.

Finally, the upload optimizer is utilized only in case the position is required to be relayed over the network as part of a collaborative sensing solution. We developed this component because we envision the smartphone position discovery service to be an important part of collaborative sensing applications. The upload optimizer logic is based on optimization techniques discussed in [14]. The optimizer implements three alternative techniques for upload optimization: (1) Upload whenever a position change occurs; (2) Upload when a position change persists for some period of time; and (3) Upload the position with the highest number of occurrences within a window of given size. While the first technique is simple and provides most accurate results, it is subject to noise due to momentary smartphone position changes. The second technique eliminates this noise and reports only more permanent position changes. Finally, the third technique is suitable when there are frequent smartphone position changes. This technique tries to report the most commonly occurring smartphone position.

Use Case Scenario: Assume a context-aware application that is interested in finding the exact position. Figure 3 demonstrates the flow of execution of the position discovery service to provide this information. In the beginning, the service uses the variance of the accelerometer for the captured window to detect the physical context. If the smartphone is idle, the service will detect if it is placed on table or any other position. In case the smartphone is idle and not on table, the exact position is difficult to provide and a cached recent value along with the activity is returned instead. Also, a running context means that position can't be detected and a cached value is returned. If the user is walking, the service will utilize the online features extracted from the accelerometer and gyroscope to provide the exact position.

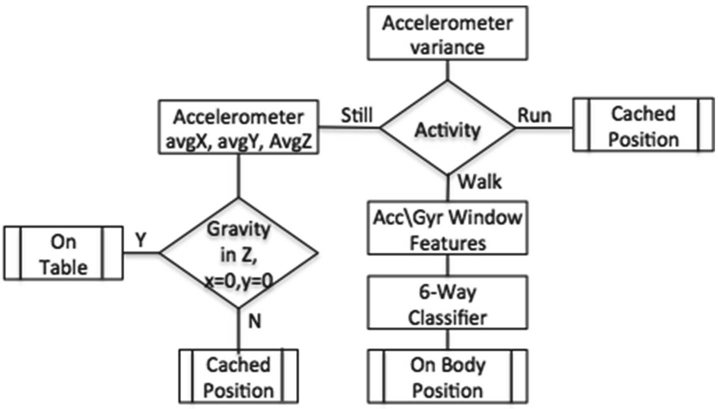


Fig. 3. Position service execution flow for a request for an exact position.

4 Implementation and Evaluation

We have implemented the proposed smartphone position discovery service on Samsung Galaxy Note device running Android version 4.0.3 (Ice Cream Sandwich). The device has a Dual-core 1.4 GHz ARM Cortex-A9 processor and 1 GB of RAM and is equipped with the accelerometer, gyroscope and light sensors required for the service. We collected data from ten different participants to train the smartphone position discovery classifier. Before conducting the experiment, an approval was obtained from the Institutional Review Board at the University of Colorado, Boulder. We asked each participant to carry the smartphone in the six smartphone positions. The experiment setup was kept as natural as possible. Participants were free to move at their own pace and place their smartphones at any orientation they liked. Next, we evaluate the accuracy of the service based on this collected training data.

Physical Contexts: To detect the physical context of a user, we calculate the standard deviation of the accelerometer magnitude and compare it to a predefined threshold. By observing the data we have chosen the threshold values of $0.5\,\mathrm{m/s^2}$ and $5\,\mathrm{m/s^2}$ to detect idle and running contexts respectively. These thresholds achieved near-perfect accuracy in our experiments.

Covered vs. Uncovered: We used a threshold of three luminous flux to detect a dimmed environment that results often from covered positions. This approach works well except for corner cases (e.g. complete absence of any light source in a room).

Upper-Body vs. Lower-Body: Smartphone positions covered in this research can be divided into two groups: upper-body group, including hand holding, talking on phone, watching a video, and jacket pocket; and lower-body group, including pants pocket and hip pocket. To detect the group that a smartphone is in, we trained the classifier with accelerometer data from 10 users and carried a 10-folds cross-validation test. The results of the classification process with different classifiers are shown in Table 2. The achieved accuracy using accelerometer is fairly high for the simple logistic regression and J48 classifiers. Therefore, we conclude that the accelerometer is the best candidate to perform this classification task and exclude gyroscope from our analysis.

Table 2. Upper-body vs. lower-body accuracy using accelerometer features

	Prediction accuracy (%)		
	Naive Bayes	Simple logistic regression	J48
Upper-body	89.2	91.2	92.8
Lower-body	77.2	90.5	83.9
Total accuracy	84.6	90.9	89.4

Exact Smartphone Position: Both accelerometer and gyroscope have shown sensitivity to smartphone positions. Our goal here is to compare between the two sensors. In the beginning, we conducted a test experiment with the six smartphone positions and collected the data for both the accelerometer and gyroscope. Then, to evaluate a single sensor, we kept the data for that sensor and deleted the data for the other sensor. By doing so, we ensure fair comparison since the three results we show next are basically for the same experiment, but, with different sensors included. Here we also used data from 10 users and performed a 10-folds cross-validation test. The classifiers employed are NB: Naive Bayes, MLP: Multilayer Perceptron, LR: logistic Regression, and J48. Table 3 illustrates the results of smartphone position classification using only accelerometer. We note that the J48 decision tree classifier achieves good accuracy of 88.5 % with the accelerometer as the only input. On the other hand, the Naive Bayes classifier had the lowest accuracy of 66.7 %. We also note that the source of confusion varies from one classifier to another for the same experiment. For example, in the multilayer perceptron experiment, the jacket-pocket position produced the lowest accuracy. On the other hand, with the Logistic Regression in use, the pants pocket position was the hardest position to classify.

Table 3. Accuracy using accelerometer

	Prediction accuracy (%)			
	NB	MLP	LR	J48
Handholding	75.9	83.8	94.5	97.8
Watching a video	91.8	93.2	96.0	97.0
Talking on phone	79.4	89.7	91.7	91.3
Pants pocket	65.4	67.9	58.0	78.2
Hip pocket	57.6	78.8	71.6	90.7
Jacket pocket	27.6	67.1	73.2	75.3
Total accuracy	66.7	80	80	88.5

Table 4. Accuracy using gyroscope

	Prediction accuracy (%)			
	NB	MLP	LR	J48
Handholding	72.2	78.4	85.4	63.6
Watching a video	64.7	81.8	84.8	84.2
Talking on phone	48.4	64.8	72.3	75.6
Pants pocket	60.6	82.0	75.5	55.2
Hip pocket	52.4	72.6	64.1	52.3
Jacket pocket	46.7	62.0	52.9	47.6
Total accuracy	57.6	74.0	72.7	62.9

Next, we evaluate the service when gyroscope is in use. Table 4 illustrates the results of smartphone position classification using only gyroscope. We note that all classifiers achieved lower total accuracy when compared to the use of accelerometer. This shows that the gyroscope is less sensitive to smartphone positions than accelerometer. Nevertheless, the accuracy achieved by the gyroscope is still at acceptable levels making the sensor worth considering for some cases. For example, some positions achieved accuracy level of above 80 % for some classifiers. However, the overall accuracy remains less than the values achieved when the accelerometer is in use.

Now, we consider the situation where we use both accelerometer and gyroscope to detect the smartphone position. Table 5 provides the position discovery results when features from both sensors are used in the classification. As expected, mostly all the classifiers achieved a gain in accuracy when compared to the previous two single-sensor configurations. We also note that three out of the four classifiers achieved very high accuracy levels (above 80 %). However, this improved accuracy comes at the cost of increased energy demands. When only accelerometer is employed the service drained 10 % from the battery in 10 h whereas, using both accelerometer and gyroscope drained more than 50 % within the same period of time. Finally, notice that none of the classifiers consistently produced best results in all three cases. However, the Naive Bayes classifiers always produced worst results due to its assumption of feature independence, which doesn't hold for our set of features.

Group Training vs. Custom Training: A smartphone is typically a personal device owned by a single user. Therefore the idea of each user (custom) training his/her position discovery classifier is worthwhile. We experimented with this idea, where a user trained his smartphone by performing the above-mentioned classifier training experiments. Next day, we collected sensor data from the same user and ran our smartphone position discovery service using the custom-trained classifier from the previous day. Table 6 shows the accuracy of smartphone position detection when both accelerometer and gyroscope data were used. We can see that the total accuracy for each classification algorithm has improved dramatically (compare the results with Table 5). One point to note is that the user wore similar clothing on both days in this experiment. We expect that the detection accuracy may be slightly lower for different style of clothing. One way to address this is to train the classifier with different clothing styles. The idea of training a classifier on smartphone by the user before application use has been used in [12]. However, the authors tried to keep the training period as minimum as possible as they believed that users might refrain from using applications requiring training beforehand. We share the same concern and believe that the position service can be installed with multiple users training data, which has shown acceptable accuracy levels, and the user is then given a choice for custom training.

Table 5. Accuracy using accelerometer and gyroscope

	Prediction accuracy (%)			
	NB	MLP	LR	J48
Handholding	76.6	86.4	98.9	92.6
Watching a video	92.4	91.3	98.0	95.8
Talking on phone	76.9	91.9	96.8	93.1
Pants pocket	73.2	85.8	78.8	75.9
Hip pocket	66.2	93.5	88.3	83.5
Jacket pocket	51.4	80.8	75.6	68.6
Total accuracy	73.0	88.6	89.3	84.9

Table 6. Accuracy using custom-training-data

	Prediction accuracy (%)			
	NB	MLP	LR	J48
Handholding	100	100	100	66.67
Watching a video	92.30	100	100	92.31
Talking on phone	100	100	100	100
Pants pocket	100	94.44	100	83.33
Hip Pocket	100	100	93.33	100
Jacket pocket	100	100	100	100
Total accuracy	98.71	99.07	98.88	90.38

On-Table Position: Due to its special nature, the on table position is handled separately. We directly use the window features of average x-axis, average y-axis, and average z-axis to detect this position. The z-axis value will be nearly equal to the gravity pull value of 9.8 m/s^2 whereas the x-axis and y-axis will have the value of near zero. This approach detected this position with nearly perfect accuracy. However, we can think of rare situations that can confuse the approach such as placing the smartphone on a stand.

5 Two Stage Classification Method

Some context-aware applications are simply blocked by disadvantageous positions (e.g. camera-based application waiting for uncovered position). Other context-aware applications can operate in different smartphone positions, but with severe accuracy degradation when experiencing a smartphone position other than the one trained for. To address this issue, we propose a two-stages classification method. First, the offline training for the classifier is done with different smartphone positions to generate a separate classifier trained for each position. Second, with the presence of the smartphone position discovery service, the application first determines the current smartphone position and then chooses the classifier corresponding to that specific position during the classification process. To demonstrate the effectiveness of this approach, we have implemented a fall classification application that was proposed in [2]. This application detects the type of fall from four different fall categories namely forward trips, backward slips , left lateral falls, and right lateral falls. The output of this application can be used by experts in the field of elderly care to develop fall prevention mechanisms and to assist first responders in providing more customized emergency procedures. The experimental work in [2] placed the smartphone to the backside of a belt and users were asked to wear this belt and simulate the different categories of falls. First, we start on reflecting how severe the situation gets when arbitrary positions are introduced to the scene? We collected training and test data from arbitrary positions for the four types of falls and report classification accuracy in Table 7. The overall accuracy of this experiment is 72.22 %. Note that [2] reported an accuracy of 98.7 % when single position is used. Second, we collected multiple training files each corresponding to a smartphone position and containing the four types of falls. Afterwards, the user was asked to simulate the required four types of falls and the classifier was pointed to the training file corresponding to the smartphone position, *assuming the smartphone position is known in advance*. The confusion matrix of the second experiment is shown in Table 8. The results from these two experiments reflect a significant accuracy improvement from 72.22 % to 94.8 %.

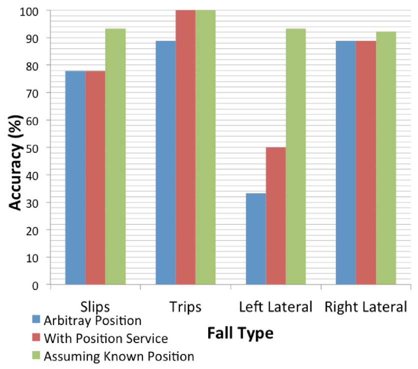
Admittedly, the assumption of a complete knowledge of smartphone position is not a valid one. Therefore, we integrated the fall classification application with the online position discovery service and followed the above-mentioned two-stage classification method. Figure 4(a) and (b) show the "per-fall" and "overall" accuracies for this case. We also included the results from Tables 7 and 8 to make it easier to grasp the effect of introducing the smartphone position discovery service. Notice from Fig. 4(a) that the accuracies of trips and left lateral falls detection have been improved. For the other two types of falls, introducing

Table 7. Fall classification accuracy with arbitrary position.

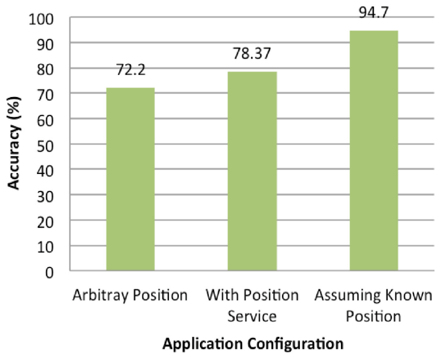
		Prediction Accuracy (%)			
		Slips	Trips	Left	Right
Fall Type	Slips	**77.8**	0	0	22.2
	Trips	11.1	**88.9**	0	0
	Left	0	0	**33.3**	66.7
	Right	0	0	11.1	**88.9**

Table 8. Fall classification accuracy assuming known position.

		Prediction Accuracy (%)			
		Slips	Trips	Left	Right
Fall Type	Slips	**93.3**	0	6.7	0
	Trips	0	**100**	0	0
	Left	6.7	0	**93.3**	0
	Right	7.1	0	0	**92.2**



(a) Per-Fall Accuracy



(b) Overall Accuracy

Fig. 4. Fall detection accuracy.

the position service to the scene didn't improve the results but didn't negatively impact them. Figure 4(b) reflects the overall accuracy improvement. The improvement in the case of "known-position" proves the fact that by introducing the position service, context-aware applications will achieve better results. We also saw an improvement for the case of with-the-position service. However, the improvement was not as significant as in the optimal situation. We noticed that our position service provided the correct position in most situations, but it was the fall classification that is difficult to achieve. We attribute this difficulty to arbitrary after fall behaviors such as standing immediately after fall or remaining stationary.

6 Conclusion

This paper presents a solution that runs solely on the smartphone to address the negative impact of different smartphone positions on context-aware applications accuracy. The proposed solution can act as a service provider to context-aware applications running on the same smartphone by providing them with smartphone position information. The service can answer the following four questions.

(1) Is the user idle, walking or running: (2) Is the smartphone covered or uncovered? (3) Is the smartphone attached to upper-body or lower-body? (4) What is the actual position of the smartphone? We evaluated the service by integrating it with an existing context-aware application for fall classification. Results show significant accuracy improvement proving the utility of the service.

References

1. Alanezi, K., Mishra, S.: Impact of smartphone position on sensor values and context discovery. Technical report, Department of Computer Science, University of Colorado (2013). http://ucblibraries.colorado.edu/repository
2. Albert, M., Kording, K., Herrmann, M., Jayaraman, A.: Fall classification by machine learning using mobile phones. PloS one **7**(5), e36556 (2012)
3. Amini, N., Sarrafzadeh, M., Vahdatpour, A., Xu, W.: Accelerometer-based on-body sensor localization for health and medical monitoring applications. In: PerCom (2011)
4. Chen, G., Kotz, D., et al.: A survey of context-aware mobile computing research. Technical report, TR2000-381, Department of CS, Dartmouth College (2000)
5. Chon, Y., Talipov, E., Cha, H.: Autonomous management of everyday places for a personalized location provider. IEEE Trans. SMCC **42**(4), 518–531 (2012)
6. Fujinami, K., Kouchi, S.: Recognizing a mobile phone's storing position as a context of a device and a user. In: Zheng, K., Li, M., Jiang, H. (eds.) MobiQuitous 2012. LNICST, vol. 120, pp. 76–88. Springer, Heidelberg (2013)
7. Gellersen, H.W., Schmidt, A., Beigl, M.: Multi-sensor context-awareness in mobile devices and smart artifacts. Mob. Netw. Appl. **7**(5), 341–351 (2002)
8. Harrison, C., Hudson, S.E.: Lightweight material detection for placement-aware mobile computing. In: UIST (2008)
9. Kunze, K.S., Lukowicz, P., Junker, H., Tröster, G.: Where am i: recognizing on-body positions of wearable sensors. In: Strang, T., Linnhoff-Popien, C. (eds.) LoCA 2005. LNCS, vol. 3479, pp. 264–275. Springer, Heidelberg (2005)
10. Kwapisz, J.R., Weiss, G.M., Moore, S.A.: Activity recognition using cell phone accelerometers. ACM SIGKDD Explor. Newsl. **12**(2), 74–82 (2011)
11. Marsan, R.J.: Weka for android. http://rjmarsan.com/research/wekaforandroid/
12. Miluzzo, E., Cornelius, C.T., Ramaswamy, A., Choudhury, T., Liu, Z., Campbell, A.T.: Darwin phones: the evolution of sensing and inference on mobile phones. In: MobiSys (2010)
13. Miluzzo, E., Papandrea, M., Lane, N.D., Lu, H., Campbell, A.T.: Pocket, bag, hand, etc.-automatically detecting phone context through discovery. In: PhoneSense (2010)
14. Musolesi, M., Piraccini, M., Fodor, K., Corradi, A., Campbell, A.T.: Supporting energy-efficient uploading strategies for continuous sensing applications on mobile phones. In: Floréen, P., Krüger, A., Spasojevic, M. (eds.) Pervasive 2010. LNCS, vol. 6030, pp. 355–372. Springer, Heidelberg (2010)
15. Shi, Y., Shi, Y., Liu, J.: A rotation based method for detecting on-body positions of mobile devices. In: UbiComp (2011)