Less Is More: Efficient Back-of-Device Tap Input Detection Using Built-in Smartphone Sensors

Emilio Granell
PRHLT, Universitat Politècnica de València
egranell@dsic.upv.es

Luis A. Leiva
Sciling
name@sciling.com

ABSTRACT

Back-of-device (BoD) interaction using current smartphone sensors (e.g. accelerometer, microphone, or gyroscope) has recently emerged as a promising novel input modality. Researchers have used a different number of features derived from these commodity sensors, however it is unclear what sensors and which features would allow for practical use, since not all sensor measurements have an equal value for detecting BoD interactions reliably and efficiently.

In this paper, we primarily focus on constructing and selecting a subset of features that is a good predictor of BoD tap-based input while ensuring low energy consumption. As a result, we build several classifiers for a variety of use cases (e.g. single or double taps with the dominant or non-dominant hand). We show that a subset of just 5 features provides high discrimination power and results in high recognition accuracy. We also make our software publicly available, so that others can build upon our work.

Author Keywords

BoD interaction; Tap-based input; Sensors; Machine learning; Feature selection

ACM Classification Keywords

H.5.2 User Interfaces: Input devices and strategies; Interaction styles; I.5.4 Pattern Recognition: Applications

INTRODUCTION

Current smartphones have enough capabilities to serve for many tasks. Increases in mobile performance, along with their high availability and range, are expanding further their capabilities and uses. One prominent use case is back-of-device (BoD) interaction, which enables eyes-free indirect input. BoD interaction can address severe occlusion problems on small touchscreens [1], has shown to be useful in increasing privacy by preventing shoulder-surfing attacks [3, 13], and can be used to control screen sharing apps such as multi-player games [2] or even to predict users' intention by the way they hold the device [15].

BoD interactions are largely dependent on sensors. Recently, it has been shown that this input modality can be accomplished on today's mobile devices with their built-in hardware like accelerometers, microphones, or gyroscopes [21, 24]. For this, many different features derived from sensor readings have been proposed, like sound volume, device motion, or frequency analysis. However, it is unclear which features would allow for practical use, since not all sensor measurements have an equal value for detecting BoD interactions reliably and efficiently. And while modern smartphones are incorporating low-power coprocessors for managing the built-in sensors, energy consumption is still a matter of great concern [20, 22, 24]. Our technique is the first to address this concern for BoD tap-based input detection, by analyzing which sensor features are most appropriate and removing thus unnecessary computations that could drain battery life after extended periods of use. Further, our technique is segmentation-free and provides high accuracy, which allows for practical, real-world use cases such as gaming or casual applications; see some examples below.

Today, one-handed input is widely recognized as a desirable property for mobile interaction [6, 9, 13], as the other hand is often occupied with a primary or secondary task [10]. For this reason, we mainly focus on detecting BoD taps with one hand. Some common scenarios of BoD tap-based interaction on smartphones include e.g. scroll a web page, take a picture, stop/resume a video, mute an incoming call, or play one-button games like Badland or FlappyBird. In sum, BoD taps enable an alternative set of interactions to augment (rather than replace) touchscreen input.

This paper presents a detection technique grounded on machine learning principles that enables highly accurate and practical BoD tap-based interaction using current hardware and sensors, without the need to instrument the mobile device. Concretely, this paper offers the following contributions:

- **Feature engineering.** We construct and select a subset of features that is a good predictor of BoD tap-based input, showing that a small subset thereof provides high discrimination power and results in high recognition accuracy.
- **BoD tap modeling.** We build several classifiers for a variety of real-world use cases; e.g. single or double taps with the dominant or non-dominant hand.
- **Release of the software.** We believe that making our software publicly available will be useful for others to enable efficient BoD interaction on current smartphones.

ISS '16, November 06–09, 2016, Niagara Falls, ON, Canada

ISBN 978-1-4503-4248-3/16/11…$15.00
http://dx.doi.org/10.1145/2992154.2992166

RELATED RESEARCH

Overall, sensing the mobile device with additional hardware has been very popular for enabling BoD interactions. For example, using a touchpad [1] or touch-sensitive surface [11], a stethoscope [14], tactile landmarks [2], or an external accelerometer attached to the device [12]. However, today's smartphones already include sophisticated built-in sensors, such as gyroscope, accelerometer, or gravity sensors. Therefore, in the remainder of this section we discuss previous works that focus on BoD interaction using current hardware and sensors.

ForceTap [8], JerkTilts [16], and TimeTilt [19] investigated motion as input using built-in accelerometers. However, these techniques seem unsuitable for continuous input. TapBack [18] relies on the sound created when tapping on the case of the phone, although recognition rates were found to be insufficient for practical use. BackPat [21] added frequency analysis and was primarily focused on finger differentiation, concluding that users preferred interacting with the index finger. Finally, BackTap [25] used sound volume and device motion for classifying BoD taps on a tablet's corners. This work was extended in BeyondTouch [24] to support more interactions, achieving encouraging results.

Altogether, previous works have relied on an unknown or a relatively arbitrary choice of sensors and features. For example, TimeTilt [19] looks at jerk movements over the Z axis (6 features), BackPat [21] uses an unknown number of features derived from gyroscope and sound frequency analysis, and BeyondTouch [24] uses 241 features sampled at 100 Hz. It is thus unclear how many features and what sensors would actually perform better for detecting BoD taps reliably. Our work is the first systematic examination in this regard.

On the other hand, researchers have been mainly concerned with accurate BoD tap event *segmentation*, and have approached this problem either from manual or ad-hoc perspectives. For example, relying on the user long-tapping on the screen for activation [21], waiting for a stable frame of 300 ms in the acceleration variation [19], or examing all available sensors and looking for a predefined threshold [22, 24]. In contrast, our technique is segmentation-free, based on solid machine learning procedures, so no previous knowledge is required to detect BoD tap-based input. Further, previous works were tested on a single smartphone, which calls into question how they would work across devices; as the position of sensors and thus the sensing values can differ from one device to another. In this regard, we have analyzed 9 different smartphones (Table 4), contributing thus with new knowledge on the feasibility and applicability of BoD tap input detection.

METHOD

Taking into account the sensors already included in most of today's smartphones, and considering the previous work on BoD interactions as well, we analyzed the accelerometer, gyroscope, gravity, and microphone. Altogether, these sensors measure, respectively: device acceleration, device angular speed, influence of the gravity acceleration, and environmental loudness.

Sensor Signals

We considered different signals derived from the sensors mentioned above. First, we computed the triaxial values and magnitude from accelerometer, gyroscope, and gravity sensors. Second, we measured environmental loudness with and without a "no tap" reference, in order to consider the influence of background noise while interacting. As in previous works [24, 25] the sound signal was treated in a decibel scale, by computing the logarithmic ratio between the measured value P and a reference value P_{ref}. Concretely, P_{ref} was set to 1 for the signal without "no tap" reference, whereas for the signal with "no tap" reference P_{ref} was set to the mean value of the first second previous to the tap acquisition. Finally, we computed the fraction of the acceleration attributed to the user's force, by subtracting the gravity acceleration to the measured device acceleration. This signal thus accounts for the actual force applied by the user while tapping. In sum, we eventually considered 18 different sensor signals for analysis, as listed in Table 1.

No.	Signal sources
4	Device acceleration (triaxial + magnitude)
4	Device angular velocity (triaxial + magnitude)
4	Gravity acceleration (triaxial + magnitude)
2	Environmental loudness (with and without "no tap" reference)
4	User acceleration (triaxial + magnitude)

Table 1: We analyzed 18 different signals derived from the smartphone sensors.

Sampling and Labeling

We conducted a preliminary experiment with 4 users to gain insights about the data we would collect afterward. Users were simply told to tap on the back of the device with their index finger within 1 second, followed by 3 seconds to rest. This was repeated up to 10 times. The Experimental Evaluation section provides more details of the procedure.

First of all, we should decide an appropriate sampling rate that would aim for lowering both computational cost and energy consumption. Based on the preliminary experiment, we observed that a BoD tap usually occurs in a time span of around 200 ms of duration (*M*=198, *SD*=11). Therefore, we measured the power consumption [26] of acquiring the sensor signals (Table 1) with the usual sampling rate of 100 Hz and by using the minimum rate that satisfies the Nyquist frequency, which in this case is a sampling rate of 10 Hz. Table 2 shows the results.

Sampling Rate (Hz)	Power Consumption (mW/s)
10	29.3 ± 6.6
100	126.3 ± 14.7

Table 2: Power consumption in the signal acquisition using different sample rates.

Together with the observed pattern of BoD taps (see an example in Figure 1), we conclude that a sampling rate of 10 Hz is enough to detect a BoD tap while ensuring minimum power consumption. Notice that this sampling rate is one order of magnitude lower than in previous works; e.g. [21, 24]. Eventually, the obtained signals were sampled using a fixed-width sliding window of 300 ms with 50% of overlapping context.

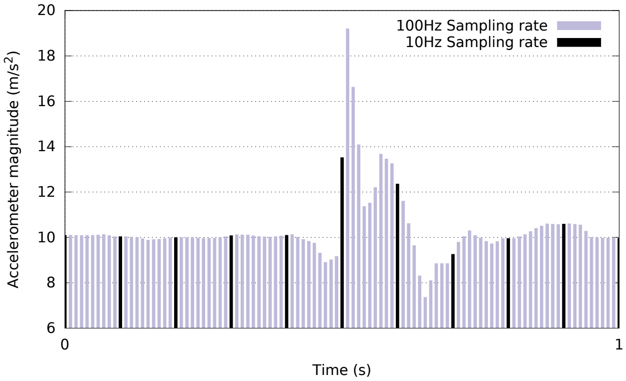

Figure 1: Sampling the accelerometer magnitude signal at 100 and 10 Hz for 1 second. When a BoD tap occurs, the detection pattern is equivalent in both cases.

Initially, during data acquisition, all samples were automatically labeled, at the frame level. That is, the samples included in the 1-second window were automatically labeled as tap frames, and the samples included in the 3-second window were automatically labeled as no-tap frames. Nonetheless, these labels were not accurate enough as not all frames in the 1-second window are actual tap frames. Thus, in order to provide a solid ground truth (i.e., the gold standard), all tap frames were ultimately supervised and manually annotated (10 814 out of 32 130 frames in total).

Feature Computation

Finally, a set of 24 functions was applied to each windowed sample. Eight of them were in the temporal domain, according to the research literature: root mean squared (RMS) and its derivative (RMS'), mean, median (Mdn), variance (σ^2), maximum (Max) and minimum (Min) values, and the Max/Min ratio. The other 16 functions were in the frequency domain, based on a Discrete Fourier Transform (DFT) [17] of 9 bins with a bandwidth of 0.625 Hz in the 0–5 Hz frequency range.[1] Table 3 depicts the functions used to parameterize each signal (both in time and frequency domains) listed in Table 1. In sum, we are considering an overall set of 18 sensors × 24 functions = 432 features for analysis.

Feature	Description
RMS	Root mean squared value
RMS'	Derivative of the RMS
Mean	Mean value
Mdn	Median value
σ^2	Variance
Max	Largest value
Min	Smallest value
Max/Min	Ratio between largest and smallest values
DFT	Discrete Fourier Transform (16 functions)

Table 3: Functions used as features of each signal sample.

Feature Selection

The purpose of feature selection is three-fold [5]: improving the prediction performance of the features, providing faster and more cost-effective features, and providing a better understanding of the underlying process that generated the data. In general, feature selection methods are used to identify and remove unneeded, irrelevant and redundant attributes from data that do not contribute to the accuracy of a predictive model or may in fact decrease it because of overfitting.

Step-wise, recursive strategies seem to be particularly computationally advantageous for feature selection. They come primarily in two flavors: forward selection and backward elimination. In forward selection, features are progressively incorporated into larger and larger subsets, whereas in backward elimination one starts with the set of all features and progressively eliminates the least promising ones. Both methods yield nested subsets of features, although backward elimination is usually preferred because in forward selection a feature added early may become redundant afterward. Therefore, in order to select the best subset of features for BoD tap-based input, we used backward feature elimination.

Recursive Backward Feature Elimination Algorithm

Formally, the algorithm for backward feature elimination is described as follows. Let $\mathcal{S} = \{S_1, \ldots, S_n\}$ be a sequence of candidates for the number of features to retain ($S_1 > S_2, \ldots$). First, the algorithm fits a chosen model to *all* features (S_1) and each feature is ranked according to its importance to the model, using a normalized score that accounts for prediction errors and correlations between features. Then, at each iteration, the top S_i ranked features are retained, the model is refit, and performance is assessed. At the end of the algorithm, a consensus ranking is used to determine the best features to retain.

EXPERIMENTAL EVALUATION

To begin, we acquired sensor data from different users, who tested different conditions of BoD tap-based input on different smartphones. Then, several algorithms for feature selection were tested; among which we chose random forests, an ensemble learning method for classification that is well suited to non-linear modeling and has been proved robust against noise and overfitting [7]. Finally, we compared different classifiers for the most promising feature subset found.

Participants

We recruited 9 participants aged 25–35 (*M*=28, *SD*=2.3) using our University's mailing lists. We intentionally wanted users with rather broad backgrounds and so we recruited participants from e.g. Mechanical Engineering, Computer Science, or Physics. There was no economic compensation for the participants, who just provided us with raw sensor data.

Apparatus

Each participant tested two devices: their own smartphone and another participant's smartphone. This way, we can cover a range of usage scenarios. First, by using each participant's own smartphone, we ensure that the user is familiar with the device itself. Second, by using other participant's

[1] Regarding our frame size, the DFT input was padded with zeros.

smartphone, we can gather insights about first-time usage of a BoD-capable device. Table 4 describes the smartphones used in our experiments. No smartphone was equipped with a protection case. We developed for Android, since it currently dominates the smartphone market.

Manufacturer	Device model	Size (mm)	Weight (g)	Android ver.
Samsung	Galaxy Note 3	151 x 79 x 8	168	4.4.4
	Galaxy S3 mini	122 x 63 x 10	112	5.0.2
	Galaxy S6 edge+	154 x 76 x 7	153	5.1.1
Google/LG	Nexus 4	134 x 69 x 9	139	5.1.1
	Nexus 5	138 x 69 x 9	130	6.0.0
Sony	Xperia Z	139 x 71 x 8	146	4.4.4
OnePlus	One	153 x 76 x 9	162	5.1.1
BQ	Aquaris M5	69 x 143 x 8	144	5.0.2
Motorola	Moto G2	141 x 71 x 11	149	5.0.2

Table 4: Smartphones used in our experiments. The 'Size' column denotes height x width x thickness.

Design

We acquired sensor readings for one-handed BoD tap input with the index finger, considering 2 factors: *Tap* (2 levels: single, double) and *Hand* (2 levels: dominant, non-dominant). This choice of factors and levels is motivated by previous works which suggested that users generally prefer tapping with the index finger and using one hand; c.f. Introduction section.

Both classification accuracy and the kappa statistic were considered the main outcome for analysis. The kappa statistic [23] gives a strong indicator of how a classifier performed across all instances, as compared to merely using accuracy for comparison. Further, the kappa statistic for one model is directly comparable to the kappa statistic for any other model used for the same classification task; akin the effect sizes used in statistical testing.

Procedure

We conducted the study in an open office environment; i.e., there were no fixed partitions or private rooms. Participants were seated during the whole study in the center of the office, surrounded by 15 employees. This way, we could strive for a balance between a carefully controlled setting and a real-world setting, where environmental noise and distractions may happen. In fact, different types of distractions actually happened, as reported by the participants. For instance, some participants found distracting the environmental noise coming from computer fans and people talking or laughing. One participant pointed out that the occasional movement of office chairs prevented him from being fully concentrated in the acquisition study. Finally, another participant mentioned that the mobile phone fell off his hands once. All these eventual distractions therefore reduced cognitive performance, which mimic a range of typical everyday scenarios. In sum, we acquired both positive (taps) and negative samples (ambient data) reflecting everyday smartphone usage.

We used a repeated measures within-subjects design, i.e., all participants tested all combinations (4 conditions in total). We ensured we would collect the same amount of data for each user: each participant had to perform 10 BoD taps with their index finger (both hands) on 2 smartphones, resulting in 40 session logs per participant and smartphone, 720 logs in total, 32 130 annotated frames overall (with tap or no-tap labels). As mentioned in the previous section, all tap-labeled data (10 814 frames) were manually supervised.

We used Latin squares to counterbalance the order of the conditions and mitigate learning effects between trials. This procedure reduces learning effects as well as asymmetrical skill transfer across conditions. We also used Latin squares to decide which device would be used by each user (both their own phone and another participant's phone).

Participants were able to practice and get accustomed to each condition before actually testing it. We developed an Android application that requested participants to enter a BoD tap within 1 second, followed by 3 seconds to prepare for the next BoD tap acquisition. We used a Wizard of Oz procedure, which informed participants only that they advanced to the next BoD tap acquisition.

Data Preprocessing and Analysis

We normalize the feature values (zero mean and unit variance) so that values that fall in greater numeric ranges do not dominate those in smaller numeric ranges. We consider user-independent tests, aiming at producing a subset of features that is a good predictor of BoD tap-based input for general use. We use 80% of the data for training (feature selection) and the remaining 20% for testing different classifiers. We balance the training data of positive (tap) and negative (no-tap) samples to avoid negatively skewing the classifiers.

To get performance estimates that incorporate the variation attributed to feature selection, we use the Area Under the ROC Curve (AUC) as the measure to optimize. The Receiver Operating Characteristic (ROC) curve is a plot of the true positive rate (TPR, also called specificity or recall) against the false positive rate (FPR, also called sensivity or fallout) of a classifier. In Section 4.6 we conduct a follow-up analysis in this regard.

By optimizing for the AUC (with regularization) we prevent overfitting and also exclude noisy features that do not contribute to the accurate discrimination of our classes (tap/no-tap). Next, we perform a 10-fold cross-validation using stratified sampling, to create balanced splits of the data that preserve the overall class distributions. We then choose the most promising feature subset and evaluate its performance against several classifiers on the testing partition.

Results

We report the results of detecting either single or double BoD taps. Actually, a double tap is detected when two consecutive taps are separated by a short amount of time; concretely, less than 300 ms according to our observations (*M*=287.52, *SD*=94.10). We first split the results by hand (dominant and no-dominant) and then analyze the different classifiers over the full testing partition (considering the data from both hands).

All experiments were performed in a user-independent setting, since user-independent tests allow us to generalize the results to potentially any user. In contrast, user-dependent tests are harder to deploy for real-world use, since each user would require a dedicated model, with a dedicated number of training and testing samples.

As observed in Figure 2, a small number of features is able to achieve competitive results. For example, using 5–10 features it is possible to achieve 98% of accuracy, which is an excellent result for real-world use. Furthermore, the Kappa statistics for subsets of 5 features and above suggest a high practical importance of the results.

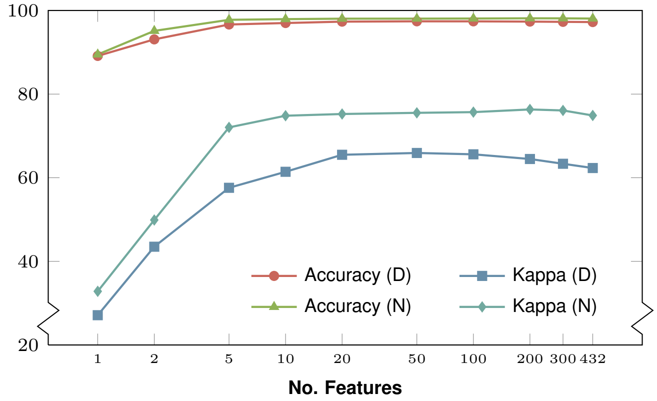

Figure 2: Results of BoD tap input detection, either single or double taps with the dominant (D) and non-dominant (N) hand.

Interestingly, slightly better results were observed for the BoD taps performed with the non-dominant hand. This is explained by the fact that users proceed a bit more carefully in comparison to their dominant hand, a well-known observation that has been reported in the research literature [4].

Surprisingly, we observed that the top-5 features were the same in all experiments; see Table 5. This is an interesting result, since a single general model can be implemented to detect BoD taps with either hand. More interestingly, the top-10 features are all related to the Fourier transform, which proves advantageous since they are really fast to compute. Also interestingly, we can conclude that the values of the higher frequency bins of the DFT from user acceleration are the features with most discriminative power overall.

In light of these observations, we decided to use the top-5 feature subset and build several classifiers. The choice of these classifiers was informed by previous works and their popularity in machine learning experiments. Concretely, we tested Partial Least Squares (PLS), Linear Discriminant Analysis (LDA), Generalized Linear Models (GLM), Generalized Boosted Models (GBM), and Support Vector Machines (SVM). We processed the data for both the dominant and non-dominant hand, aiming for general-purpose classifiers. Table 6 shows the classification results.

By way of baseline, in these experiments we include a PLS classifier that takes into account all the 432 features (PLS0). We also include SVMb, which is the same classifier with the same feature set used in BeyondTouch [24], by way of comparison with the most recent approach to detecting BoD taps.

Rank	Feature	Signal	Importance
1	$\Re(\text{DFT}_{4.375})$	| User acc. |	14.95
2	$\Re(\text{DFT}_{5})$	| User acc. |	13.82
3	$\Re(\text{DFT}_{3.75})$	| User acc. |	12.56
4	$\Im(\text{DFT}_{3.125})$	| User acc. |	12.09
5	$\Im(\text{DFT}_{2.5})$	| User acc. |	9.95
6	$\Im(\text{DFT}_{2.5})$	User acc. Z axis	9.75
7	$\Im(\text{DFT}_{3.75})$	| User acc. |	9.48
8	$\Im(\text{DFT}_{2.5})$	| Acc. |	8.15
9	$\Im(\text{DFT}_{4.375})$	| User acc. |	8.12
10	$\Im(\text{DFT}_{3.125})$	Gyro X axis	7.85

Table 5: Rank of the top-10 most discriminative features. $\Re(\text{DFT}_i)$ and $\Im(\text{DFT}_i)$ represent the real and imaginary parts of the bin centered in the ith frequency (in Hz), $|\cdot|$ denotes the magnitude of the triaxial signals, and "acc." stands for "acceleration."

Classifier	Accuracy (%)	95% Conf. Interval
PLS0	95.12	[93.87, 96.19]
PLS	96.75	[95.69, 97.61]
GLM	97.10	[96.09, 97.91]
LDA	97.10	[96.09, 97.91]
SVM	97.53	[96.58, 98.27]
SVMb	97.10	[96.09, 97.91]
GBM	97.46	[96.50, 98.21]

Table 6: Classification accuracy for BoD tap input with either hand, using the set of top-5 features. PLS0 indicates a PLS classifier trained with all the 432 features. SVMb denotes the BeyondTouch classifier.

As can be observed, using all features proves disadvantageous in many senses. Not only recognition accuracy is significantly degraded, but also a classifier that uses all features will result in less discriminative power (Table 7) and will require more computational resources than e.g. a GLM that uses just the top-5 discriminative features. This is an important result because selecting the right features can mean a difference between low performance with long computation times and high performance with short computation times.

Also an interesting result is the fact that BeyondTouch could perform a bit better if our set of discriminant features were used. More important, we should note that BeyondTouch uses 241 features sampled at 100 Hz and the equivalent SVM we tested uses 5 features sampled at 10 Hz.

ROC Analysis

One of the harder parts of implementing BoD taps is not about detecting the tap itself, but rather about not raising false positives. These false positives could happen when the user is doing any other activity with the mobile device that may shake it. For this reason, we decided to conduct an additional study on this matter.

The ROC curve (true positive rate *vs.* false positive rate) has many interesting properties:

1. It shows the tradeoff between sensitivity and specificity: any increase in sensitivity will be accompanied by a decrease in specificity.
2. The area under the curve is a measure of classification accuracy.
3. The closer the curve follows the left-hand border and then the top border of the ROC space, the more accurate the classifier.
4. The closer the curve comes to the 45-degree diagonal of the ROC space, the less accurate the classifier.

Classifier	PLS0	PLS	GLM	LDA	SVM	SVMb	GBM
AUC	0.500	0.704	0.782	0.831	0.805	0.768	0.833

Table 7: Testing the discriminative power of different classifiers with the Area Under the ROC Curve (AUC). PLS0 indicates a PLS classifier trained with all the 432 features. SVMb denotes the BeyondTouch classifier.

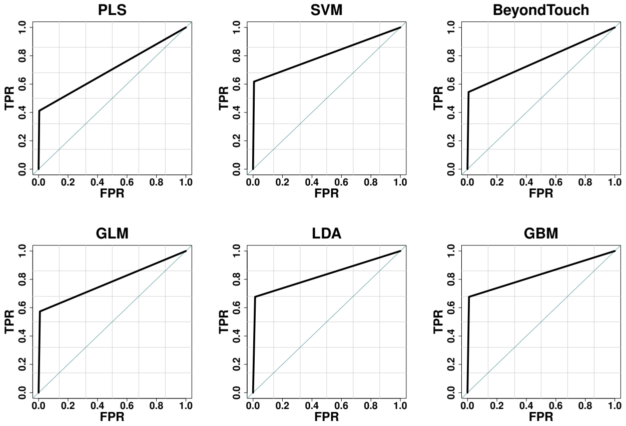

Figure 3: ROC curves of the different classifiers. The PLS0 curve is the 45-degree diagonal, so it is omitted.

A perfect ROC curve will be shaped with a sharp bend. As observed in Figure 3, the performance achieved by most of the tested classifiers is near perfect separation, with GBM and LDA achieving the best results. PLS0 achieved the worst results (its ROC curve is the 45-degree diagonal line). We should mention that the angle-shape elbow appears in logistic regression, i.e., when both the ground truth output and the prediction are binary (tap, no-tap).

Taken together, our results suggest that GLM, LDA, SVM, and GBM are the most competitive classifiers overall, given their accuracy (Table 6) and discriminative power (Table 7 and Figure 3). We hope this evaluation, together with the rest of the paper, will inform researchers and developers interested in incorporating robust BoD tap-based interactions to their prototypes or applications.

SOFTWARE IMPLEMENTATION

We have developed a background service that abstracts the logic for BoD tap input detection. The service implements a GLM-based classifier and does not provide a user interface, this way developers simply have to instantiate the service in their own apps. Moreover, the service runs in a single background thread. This allows it to detect BoD tap input without affecting the user interface's responsiveness.

In order to avoid interfering with unintended motion data, the service does not read the device sensors unless some application requests using the service. This also helps to reduce energy consumption to a great extent, since BoD tap detection is performed only when an app explicitly requires it.

Our implementation provides single and double BoD taps detection off-the-shelf. Therefore, the service can be used to handle basic interactions such as controlling a music player or selecting text onscreen. The service can also be used to complement traditional interactions such as muting an incoming call or playing one-button games. At the moment, the software is available for the Android operating system and can be downloaded from https://btap.tech.

LIMITATIONS

Apart from the "usual suspects" in lab-based studies (number of participants, age, etc.), we should mention a number of limitations worth discussing about this work. We believe that understanding these will be useful to researchers and developers interested in detecting BoD tap-based interactions reliably.

Firstly, we decided that our technique should detect only two location-unaware BoD taps (namely, single and double BoD taps anywhere on the smartphone's case). This decision was informed by previous works; see Related Research section. Eventually, considering more tap locations (1) requires the user positioning his hand more precisely [15] and, most important, (2) decreases accuracy to a great extent [24]. We believe that this would deter real-world use, however it is an interesting research avenue to consider in future work; see next section.

Secondly, we have not tested the use of protection cases, which are relatively common among smartphone users. It is expected that different case materials and thicknesses would have an impact on recognition accuracy. However, this is essentially the subject of a different paper.

Lastly, we should mention that the frequency range we considered was aimed at minimizing energy consumption (see Table 2) and thus it precludes the use of more advanced BoD interactions. For example, any tap can be decomposed into much shorter events (e.g., onset of contact, application of force, release of contact) that unfortunately are hard to detect within a sampling rate of 10 Hz.

FUTURE WORK

At present, we plan to explore how our BoD taps detection technique could be applied in other interaction domains. Namely, *off-screen* interactions such as patting with the thumb on the side of the device [21] and *around-device* interactions such as detecting slide gestures on a nearby surface [24].

On the other hand, we only considered sound frequencies in the subaudible range of 0–5 Hz, therefore the microphone sensor seems irrelevant in our experiments, discriminative wise. Nevertheless, richer interactions can be considered if we incorporate features derived from this sensor. For example, the above-mentioned around-device sliding gestures for scroll, zoom, etc. Although it is important to notice that the microphone, despite being an excellent sensor to detect BoD taps alone, is very sensible to ambient noise and thus it often raises many false positives. Therefore, it must be intelligently combined with other features to ensure a practical use.

CONCLUSION

We have proposed an efficient and reliable technique for detecting BoD tap-based input on current smartphones using commodity sensors. The value of our technique lies in the fact that we use low-cost yet highly discriminative features. We have made our software public so that others can build upon our work. The software can be easily incorporated into production-ready applications, as developers simply have to instantiate a background activity that abstracts the logic for BoD tap input detection.

Device manufacturers can benefit from this work by selecting an appropriate BoD tap detection feature subset that is performance-friendly and relatively easy to implement. Overall, BoD interaction can be used either as a means of direct input or as a complementary input aid. Ultimately, our work provides designers, researchers, and practitioners with new understanding on robust and efficient BoD tap input detection.

ACKNOWLEDGMENTS

Work partially supported by SmartWays (MINECO, RTC-2014-1466-4) and CoMUN-HaT (MINECO/FEDER, TIN2015-70924-C2-1-R).

REFERENCES

1. Baudisch, P., and Chu, G. Back-of-device interaction allows creating very small touch devices. In *Proc. CHI* (2009).
2. Corsten, C., Cherek, C., Karrer, T., and Borchers, J. HaptiCase: Back-of-device tactile landmarks for eyes-free absolute indirect touch. In *Proc. CHI* (2015).
3. De Luca, A., von Zezschwitz, E., Nguyen, N. D. H., Maurer, M.-E., Rubegni, E., Scipioni, M. P., and Langheinrich, M. Back-of-device authentication on smartphones. In *Proc. CHI* (2013).
4. Gignac, G. E., and Vernon, P. A. Reaction time and the dominant and non-dominant hands: An extension of Hick's law. *Personality and Individual Differences 36*, 3 (2004).
5. Guyon, I., and Elisseeff, A. An introduction to variable and feature selection. *J. Mach. Learn. Res. 3* (2003).
6. Hassan, N., Rahman, M., Irani, P., and Graham, P. Chucking: A one-handed document sharing technique. In *Proc. INTERACT*, vol. 5727 of *LNCS*. 2009.
7. Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning*, 2nd ed. Springer-Verlag, 2009.
8. Heo, S., and Lee, G. ForceTap: extending the input vocabulary of mobile touch screens by adding tap gestures. In *Proc. MobileHCI* (2011).
9. Hinckley, K., and Song, H. Sensor synesthesia: touch in motion, and motion in touch. In *Proc. CHI* (2011).
10. Hirotaka, N. Reassessing current cell phone designs: using thumb input effectively. In *Proc. CHI EA* (2003).
11. Holman, D., Hollatz, A., Banerjee, A., and Vertegaal, R. Unifone: Designing for auxiliary finger input in one-handed mobile interactions. In *Proc. TEI* (2013).
12. Hudson, S. E., Harrison, C., Harrison, B. L., and LaMarca, A. Whack gestures: inexact and inattentive interaction with mobile devices. In *Proc. TEI* (2010).
13. Leiva, L. A., and Catalá, A. BoD Taps: An improved back-of-device authentication technique on smartphones. In *Proc. MobileHCI* (2014).
14. Lopes, P., Jota, R., and Jorge, J. A. Augmenting touch interaction through acoustic sensing. In *Proc. ITS* (2011).
15. Mohd Noor, M. F. Exploring back-of-device interaction. In *Adj. Proc. UIST* (2013).
16. Oakley, I., and Park, J. A motion-based marking menu system. In *Proc. CHI EA* (2007).
17. Proakis, J. G., and Manolakis, D. K. *Digital Signal Processing*, 4th ed. Prentice Hall, 2006.
18. Robinson, S., Rajput, N., Jones, M., Jain, A., Sahay, S., and Nanavati, A. TapBack: Towards richer mobile interfaces in impoverished contexts. In *Proc. CHI* (2011).
19. Roudaut, A., Baglioni, M., and Lecolinet, E. Timetilt: using sensor-based gestures to travel through multiple applications on a mobile device. In *Proc. INTERACT*, vol. 5726 of *LNCS*. 2009.
20. Saeedi, R., Schimert, B., and Ghasemzadeh, H. Cost-sensitive feature selection for on-body sensor localization. In *Adj. Proc. UbiComp* (2014).
21. Seipp, K., and Devlin, K. BackPat: One-handed off-screen patting gestures. In *Proc. MobileHCI* (2014).
22. Serrano, M., Lecolinet, E., and Guiard, Y. Bezel-Tap gestures: Quick activation of commands from sleep mode on tablets. In *Proc. CHI* (2013).
23. Siegel, S., and Castellan, N. J. *Nonparametric Statistics for the Behavioural Sciences*, 2nd ed. McGraw-Hill, 1988.
24. Zhang, C., Guo, A., Zhang, D., Southern, C., Arriaga, R., and Abowd, G. BeyondTouch: Extending the input language with built-in sensors on commodity smartphones. In *Proc. IUI* (2015).
25. Zhang, C., Parnami, A., Southern, C., Thomaz, E., Reyes, G., Arriaga, R., and Abowd, G. D. BackTap: Robust four-point tapping on the back of an off-the-shelf smartphone. In *Adj. Proc. UIST* (2013).
26. Zhang, L., Tiwana, B., Qian, Z., Wang, Z., Dick, R. P., Mao, Z. M., and Yang, L. Accurate online power estimation and automatic battery behavior based power model generation for smartphones. In *Proc. CODES+ISSS* (2010).