EarBuddy: Enabling On-Face Interaction via Wireless Earbuds

Xuhai Xu[1,2], **Haitian Shi**[2,3], **Xin Yi**[2,4+], **Wenjia Liu**[5], **Yukang Yan**[2], **Yuanchun Shi**[2,4], **Alex Mariakakis**[3], **Jennifer Mankoff**[3], **Anind K. Dey**[1]

[1]Information School | DUB Group, University of Washington, Seattle, U.S.A

[2]Department of Computer Science and Technology, Tsinghua University, Beijing, China

[3]Paul G. Allen School of Computer Science & Engineering | DUB Group, University of Washington, Seattle, U.S.A

[4]Key Laboratory of Pervasive Computing, Ministry of Education, Beijing, China

[5]Department of Computer Science and Technology, Beijing University of Posts and Telecommunications, Beijing, China

{xuhaixu, sht19, anind}@uw.edu, {yixin, shiyc}@mail.tsinghua.edu.cn,

{sophie_liu}@bupt.edu.cn, {yyk15}@mails.tsinghua.edu.cn, {atm15, jmankoff}@cs.uw.edu

ABSTRACT

Past research regarding on-body interaction typically requires custom sensors, limiting their scalability and generalizability. We propose EarBuddy, a real-time system that leverages the microphone in commercial wireless earbuds to detect tapping and sliding gestures near the face and ears. We develop a design space to generate 27 valid gestures and conducted a user study (N=16) to select the eight gestures that were optimal for both human preference and microphone detectability. We collected a dataset on those eight gestures (N=20) and trained deep learning models for gesture detection and classification. Our optimized classifier achieved an accuracy of 95.3%. Finally, we conducted a user study (N=12) to evaluate EarBuddy's usability. Our results show that EarBuddy can facilitate novel interaction and that users feel very positively about the system. EarBuddy provides a new eyes-free, socially acceptable input method that is compatible with commercial wireless earbuds and has the potential for scalability and generalizability.

Author Keywords

Wireless earbuds; face and ear interaction; gesture recognition

CCS Concepts

•**Human-centered computing** → **Human computer interaction (HCI);** *Interaction techniques; Ubiquitous and mobile computing systems and tools;*

INTRODUCTION

Past research from the human-computer interaction community has explored the use of surfaces on the body like the palms [65], arms [26], nails [27], and teeth [71] for convenient, subtle, and eyes-free communication [20]. Leveraging these surfaces has typically required custom sensors—fingertip cameras [60], ultrasonic wristbands [77], and capacitive fingernails [27], *etc.* Such custom sensors limit the scalability and generalizability to other applications.

Figure 1: EarBuddy leverages the microphone embedded in wireless earbuds to recognize gestures on the face or around the ears.

Our work takes advantage of the growing popularity of wireless earbuds as ubiquitous sensors for on-body sensing. Apple sold tens of millions of AirPods [16]. Other companies like Samsung [7] and Sony [8] are expected to show comparable trends in uptake of their earbuds. Although wireless earbuds are mainly used for audio output (*i.e.*, playing music and videos), most products also include a microphone for audio input so that people can respond to phone calls. The fact that wireless earbuds rest within a person's ears means that their microphone is conveniently situated near multiple surfaces that are suitable for on-body interaction: the cheek, the temple, and the ear itself. Tapping and sliding fingers across these surfaces generates audio signals that can be captured by an earbud, transmitted to a smartphone via Bluetooth, and then processed on-device to interpret gestures.

This observation gives rise to EarBuddy, a novel eyes-free input system that detects gestures performed along users' faces using wireless earbuds. As shown in Figure 1, users can easily control a music player or react to a notification by EarBuddy. Since EarBuddy augments the capabilities of devices that are already commercially available, our technique can easily be deployed through software updates to the phone to provide new interaction experiences for users.

+ indicates the corresponding author.

CHI '20, April 25–30, 2020, Honolulu, HI, USA.

ACM ISBN 978-1-4503-6708-0/20/04 ...$15.00.
http://dx.doi.org/10.1145/3313831.3376836

We develop a comprehensive design space with 27 gestures along the side of a person's face and ears. Since users cannot realistically remember all 27 gestures and some gestures are not easily detectable by earbud microphones, we conducted a user study (N=16) to narrow our gesture set to eight gestures. We carried out a second user study (N=20) to collect a thorough dataset with those gestures in both a quiet environment and an environment with background noise. We used that data to train a shallow neural network binary classifier to detect gestures and a deep DenseNet [25] to classify gestures. Our best classifier achieved a classification accuracy of 95.3%. Finally, we built a real-time implementation of EarBuddy using those models and conducted a third user study (N=12) to evaluate EarBuddy's usability. Our results show that EarBuddy sped up interactions by 33.9 - 56.2% compared to touchscreen interactions. Users provided positive feedback as well, saying that EarBuddy can be used easily, conveniently, and naturally.

Our contributions of this paper are threefold:

- We propose EarBuddy, a novel eyes-free input technique supported by wireless earbuds without the need for hardware modification, and implement a real-time instantiation of EarBuddy.
- We create a two-dimensional design space for gestures near the face and ears. Our first user study selects the gesture set for EarBuddy that is optimized for user preference and microphone detectability.
- We train a gesture recognition model based on a second data collection study, and evaluate the usability of EarBuddy in a third user study.

RELATED WORK

We provide a general overview of on-body interaction with special attention towards interactions with the face and ears. We then review research on sound-based activity recognition.

On-Body Interaction and Sensing

On-body interaction refers to the use of the human body as an input or output channel [20]. A wide range of human body parts have been leveraged for on-body interaction. Examples include the palm [19, 20, 65], arms [18, 20, 26], fingers [24, 67, 70], nails [27, 63], the face [26, 57, 74], ears [28, 36, 44], and teeth [10, 71], and even clothing that goes beyond skin [48, 56]. Researchers have used various sensing techniques to support these interaction surfaces. For example, Harrison *et al.* [20] used a ceiling-mounted infrared camera to locate a person's arms and hands and a digital light processing projector to shine interfaces onto the user's limbs. FingerPing by Zhang *et al.* [77] identified hand postures using an ultrasonic speaker on the thumb and speakers placed at the thumb and wrist. Through capacitive sensing, Kao *et al.* [27] created printable electrodes that can be placed on a person's fingernails to enable touch gestures on nails. Finally, Weigel *et al.* [67] explored various forms of deformation sensing (*e.g.*, capacitive and strain sensors) for on-skin gestures.

The aforementioned techniques require additional hardware, thus limiting their deployability. In this paper, we strictly rely on the microphone that is built into commercially available wireless earbuds to detect gestures on the face and ears.

Interaction on the Face and Ears

Within the realm of face and ear gestures, Serrano *et al.* [57] examined the overall design space on the face for head-mounted display interaction, with special attention paid towards social acceptability. Their findings suggest that the cheek and forehead are the most practical locations for gesture sensing. However, they did not use their findings to propose a specific gesture set for the face. Lissermann *et al.* [36] offer three categories of interaction with the ear rim: touch (slide, single- and multi-touch), grasp (bend, pull lobe, and cover), and mid-air (hover and swipe). Inspired by the related work and literature on gestures performed with touch screens [11, 38, 72], we propose a two-dimensional design space for touch-based interaction on the face and ears.

To detect face- and ear-based gestures, Masai *et al.* [41] installed photo reflective sensors on glasses to measure cheek deformation during different facial expressions [42]. Yamashita *et al.* [73] used similar sensors on a head-mounted display to detect face-pulling gestures. Kikuchi *et al.* [28] augmented earbuds with photoreflective sensors around the periphery; as users tugged on their ear, the distance between the ear's antihelix and the sensors changed to produce distinguishable signals. Lissermann *et al.* [36] detected gestures behind the ear using an array of capacitive sensors. Wang *et al.* [66] used the capacitive phone screen to capture the contact between the ear and the screen to help blind users to interact with the phone with ear. Tamaki *et al.* [62] mounted a camera and a projector on earbuds to recognize hand gestures and provide visual feedback. Lastly, Metzger *et al.* [44] added a proximity sensor to earbuds to detect in-air gestures near the ear. As with the broader literature concerning on-body interaction, none of this work investigates gesture recognition without the use of additional hardware. To the best of our knowledge, we are the first to detect touch-based gestures on the face and ears using existing commercially available wireless earbuds for interaction.

Sound-Based Activity Recognition

Sound can capture rich information about a person's physical activity and social context, thus leading researchers to use audio signals for activity recognition. For example, Chen *et al.* [12] used acoustic signals on a wooden tabletop to recognize users' finger sliding. These methods have used a range of classification models, ranging from traditional machine learning models like support-vector machines [15] and hidden Markov models [14] to deep learning models like fully connected networks [31] and convolutional neural networks [12, 23, 69]. Models for activity recognition have also leveraged different types of audio features. Lu *et al.* [39], for example, demonstrated that time-based features like zero-crossing rate and low energy frame rate can be used to distinguish speech, music, and ambient sound with a smartphone's microphone. Mel-frequency cepstral coefficients (MFCCs) are a particularly popular choice for audio analysis because of how they distribute spectrogram energy in accordance with human hearing. Stork

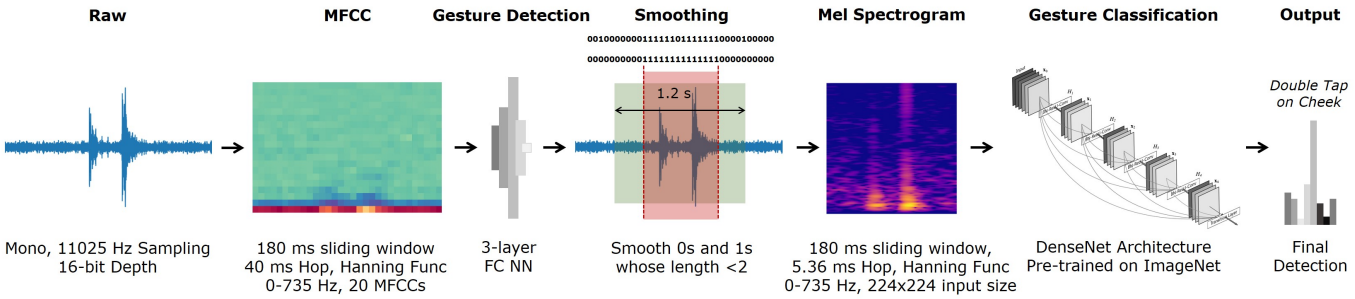


Figure 2: EarBuddy pipeline overview. Audio augmentation and optimizer tuning techniques are used to tune the state-of-the-art vision model DenseNet pre-trained on ImageNet Dataset.

et al. [61] used non-Markovian ensemble voting based on MFCC features to have a robot distinguish 22 human activities within bathrooms and kitchens. Laput *et al.* [32, 33] developed custom hardware to distinguish 38 environmental events using MFCCs and a pre-trained neural network.

Closer to our work, BodyScope [75] and BodyBeat [49] combined time- and frequency-based features to classify sounds recorded by a microphone pressed directly against a person's throat. Both systems recognize events like coughing and chewing but hint at the idea of recording subtle sounds like hums and clicks. EarBuddy builds on this idea, using deep learning to classify gestures on the face and ears.

EARBUDDY DESIGN

EarBuddy allows people to perform tapping and sliding gestures on their face and around their ears to interact with devices. We leverage the fact that touching body parts naturally produces subtle but perceptible sounds that can be captured by wireless earbuds. We introduce both the sound-capturing system and interaction design below.

System Design

EarBuddy recognizes gestures in two steps. First, a gesture detector judges whether a gesture is present. If a gesture is detected, the gesture is recognized by a classifier. Figure 2 illustrates the overall pipeline of the system, which we describe in detail below. For the purposes of this paper, we implement EarBuddy using Samsung's Gear IconX 2018 wireless earbuds [7]. The built-in microphones of these earbuds sample sound through a single channel at 11.025 kHz with 16-bit resolution.

Detection

Gesture detection starts using a 180 ms sliding window with a step size of 40 ms. Twenty MFCCs are extracted from the window at each step and fed into a binary neural network classifier [31, 61]. The classifier outputs a 1 whenever there is audio content belonging to a gesture and a 0 otherwise. Almost all gestures take longer than three single steps ($>$ 120 ms), so the presence of a gesture should lead the classifier to produce multiple 1's in succession; however, temporal shifts in the data and noise can make the classifier's serial output noisy. EarBuddy remedies this issue by using a majority voting scheme where adjacent sequences of consecutive 1's are merged if they are separated by one or two 0's. A gesture is defined to be present whenever there are 3 or more consecutive 1's, corresponding to a minimum gesture duration of 120 ms. Whenever a gesture occurs, EarBuddy takes a 1.2 s segment of raw audio (covers more than 99 % of the gestures) centered on the sequence of 1's and feeds it into the gesture classifier.

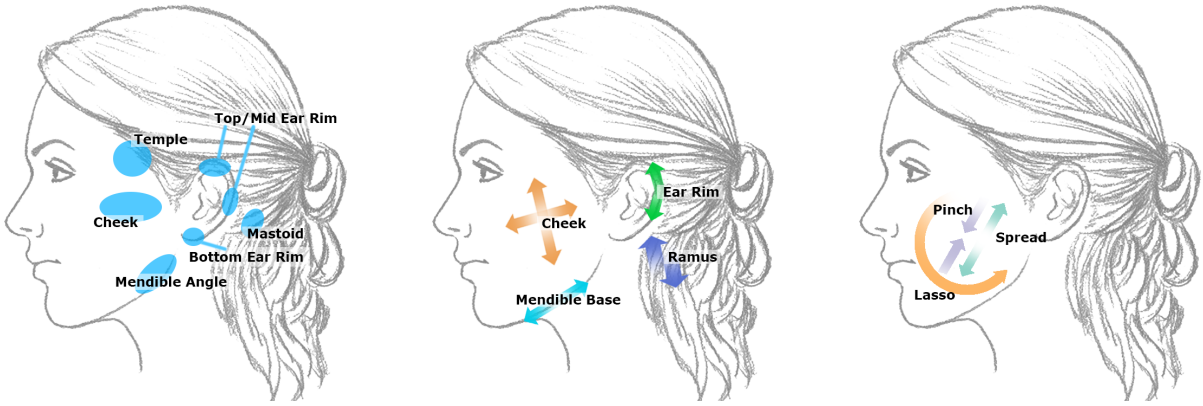
Classification

EarBuddy processes audio data for classification using mel spectrograms, similarly to past work [23, 32]. Mel spectrograms are generated by applying the short-time Fourier transform with a 180 ms window and step size of 1200 / 224 = 5.36 ms, thus yielding a 224-length linear spectrogram that can be converted into a 224-bin mel spectrogram. This process produces a 224$\times$224 input frame for each audio segment that can be fed into a deep-learning classification model.

Deep learning models with large numbers of parameters are very capable of accurately modeling data. However, training a deep model from the scratch on a small dataset can easily lead to overfitting. Transfer learning alleviates this issue by pre-training a model from a large, well-labeled dataset and then conducting additional training with the smaller target dataset [46]. As EarBuddy converts audio signals into mel spectrograms, the 1-D audio signal is transformed into a 2-D image format. We tried transfer learning using pre-trained vision models like VGG16 [58], ResNet [22], and DenseNet [25]. We found that DenseNet, which is pre-trained on ImageNet-12 [53], produced the best accuracy for our data, leveraging the advantages of DenseNet: having a deep dense network but relatively small number of parameters. DenseNet is a network with one convolutional layer, four dense blocks, and intermediate transition layers. We modify this architecture after pre-training by replacing the last fully-connected layer with two fully-connected layers, using a dropout layer [59] a ReLU activation function [45] in between. Modifying the output layer is required because DenseNet has 1000 possible output classes for the ImageNet dataset [25], but EarBuddy requires far fewer output classes (1 for each gesture). We train the modified, pre-trained network on our dataset to produce the final classification model used by EarBuddy.

Real-time System Implementation

We prototype EarBuddy using a ThinkPad T570 laptop with a quad-core CPU processor to perform gesture recognition in real-time. The wireless earbuds transmit the microphone audio to the laptop via Bluetooth in 40 ms chunks. The chunks are accumulated to identify the presence of gestures and perform classification when needed. Despite the fact that our



(a) Tap-based Gestures (b) Simple Slide-based Gestures (c) Complex Slide-based Gestures

Figure 3: Gesture Design of EarBuddy

Table 1: The names and shorthand identifiers for all 27 gestures that we investigated in this work: (T1-) single tap gestures, (T2-) double tap gestures, (S-) simple sliding gestures, and (C-) complex sliding gestures.

T1-Temple Single Tap on Temple	T1-Cheek Single Tap on Cheek	T1-Mandible Single Tap on Mandible Angle	T1-Mastoid Single Tap on Mastoid	T1-TopEar Single Tap on Top Ear Rim	T1-MiddleEar Single Tap on Middle Ear Rim	T1-BottomEar Single Tap on Bottom Ear Rim
T2-Temple Double Tap on Temple	T2-Cheek Double Tap on Cheek	T2-Mandible Double Tap on Mandible Angle	T2-Mastoid Double Tap on Mastoid	T2-TopEar Double Tap on Top Ear Rim	T2-MiddleEar Double Tap on Middle Ear Rim	T2-BottomEar Double Tap on Bottom Ear Rim
SBF-Cheek Back-to-Front Slide on Cheek	STB-Cheek Top-to-Bottom Slide on Cheek	STB-Ear Top-to-Bottom Slide on Ear Rim	STB-Mandible Top-to-Bottom Slide on Mandible Base	STB-Ramus Top-to-Bottom Slide on Ramus	C-Pinch Two Fingers Pinch	C-Lasso Lasso on Cheek
SFB-Cheek Front-to-Back Slide on Cheek	SBT-Cheek Bottom-to-Top Slide on Cheek	SBT-Ear Bottom-to-Top Slide on Ear Rim	SBT-Mandible Bottom-to-Top Slide on Mandible Base	SBT-Ramus Bottom-to-Top Slide on Ramus	C-Spread Two Fingers Spread	

laptop does not have a GPU, the average computation time of detection and classification is only 190ms. The average delay between the completion of a gesture and the classification result being returned is around 800 ms.

Interaction Design

People can produce different sounds by touching different areas around their face and ears. This is because the face and ears have unique structures with distinct combinations of materials. For example, the ear rim is primarily composed of cartilage, while the cheek is typically more fleshy. We identified seven areas that can be used for interaction: the temple, the cheek, the mandible angle, the mastoid, and the top/middle/bottom of the ear rim.

Different sounds can also be produced by different touching gestures. For instance, sliding gestures produce a sustained high-frequency sound, whereas a tap produces a broadband impulse. Past work has explored a number of touch-based finger gestures [34, 57, 64] including tap-based gestures (single- and double-tap) and slide-based gestures (straight slide, lasso slide, and pinch-and-spread).

Together, the gesture's position on the face and the action by the fingers are the two dimensions that define our design space. Using all possible pairs of options along those two dimensions that are feasible to perform, we generate 27 gestures (Figure 3). Single- and double-tap gestures can be performed at all 7 locations within our design space (Figure 3a), producing 14 tap-based gestures (T1-/T2-). Simple slide-based gestures, on the other hand, can only be performed on larger areas of the face: the cheek, the ear rim, the ramus, and the mandible base (Figure 3b). At each location, sliding can be either top-to-bottom (STB-) or bottom-to-top (SBT-). Because the cheek is particularly wide, it is also possible to perform back-to-front (SBF-) and front-to-back (SFB-) slides on it. The cheek can also support complex sliding gestures (Figure 3c) like a lasso motion (C-Lasso), a two-finger pinch (C-Pinch), and a spreading gesture (C-Spread).

STUDY 1: GESTURE SELECTION

We wanted to narrow down the gesture set from 27 gestures to a subset that can be naturally performed, quickly remembered, and easily classified. Therefore, we conducted a study to identify a subset of the most preferable gestures.

Participants and Apparatus

We recruited 16 participants (8 male, 8 female, age = 21.3 $\pm$ 0.9) via email and paper flyers. The study was conducted in a quiet room with an ambient noise level around 35-40 dB. As mentioned earlier, we implemented EarBuddy using a pair of Samsung Gear IconX [7] for data collection.

Design and Procedure

Each participant performed all 27 gestures three times using their right hand. The order of the gestures was pre-determined

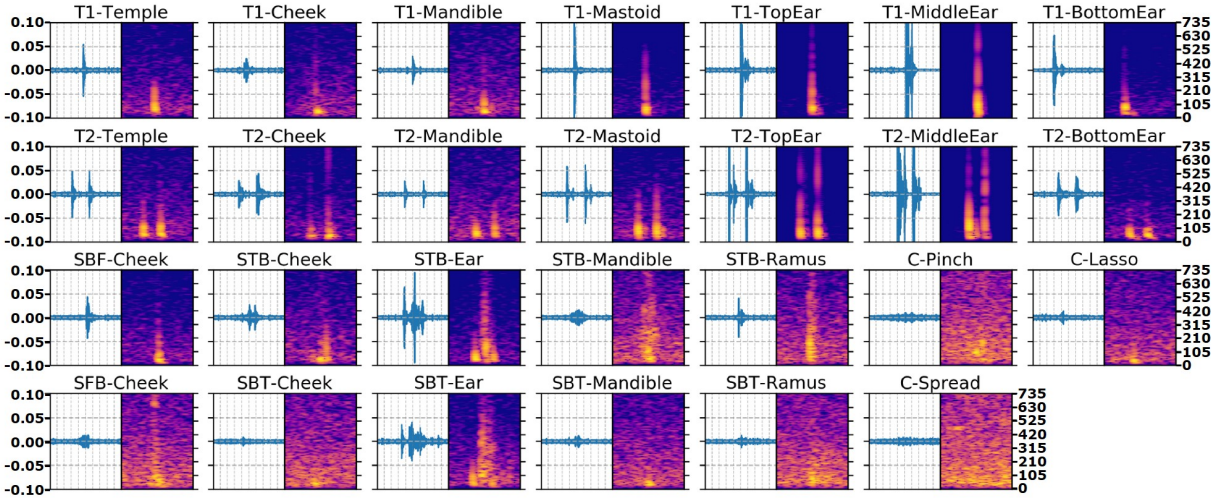


Figure 4: Example plots of all 27 gestures. For each plot, the left side is the waveform of the raw audio and the right side is the mel spectrogram. The X-axis indicates the window size, which is 1.2 s.

to counterbalance ordering effects. For each gesture, the experimenter led the participant through a brief practice phase to ensure they could perform the gesture correctly. The participant then followed instructions provided on a laptop screen to perform gestures at pre-defined times; doing so facilitated gesture segmentation for data analysis. After performing the gesture three times, the participant was asked to rate the gesture according to three criteria along a 7-point Likert scale (1: strongly disagree to 7: strongly agree):

- *Simplicity*: "The gesture is easy to perform precisely."
- *Social acceptability*: "The gesture can be performed without social concern."
- *Fatigue*: "The gesture makes me tired." (Note: Likert scores were reversed for analysis)

Results

Figure 4 shows example signals for all gestures. Figure 5 shows all gestures' ratings, sorted by the sum of the scores. We used the following aspects to select the best gestures:

1. *SNR*. We calculated each sample's signal-to-noise ratio (SNR) and removed the gestures that had an average SNR lower than 5 dB. This removed eight gestures, many of which were sliding-based gestures that either went bottom-to-top or complex sliding gestures: SBT-Cheek, SBT-Ear, SBT-Mandible, STB-Mandible, SBT-Ramus, C-Spread, C-Pinch, C-Lasso.
2. *Signal Similarity*. We used dynamic time warping (DTW) [54] on the raw data to calculate signal similarity between pairs of gestures. We created a 27×27 distance matrix where each entry was the average DTW distance across all possible pairs of the corresponding gestures. We then summed each row to calculate the similarity between each gesture and all others. Gestures with total distances lower than the 25th percentile were removed, since they are most likely to be confused during classification. Doing so removed T1-Temple, T1-Mandible, T1-TopEar, T2-Mandible, T2-BottomEar.

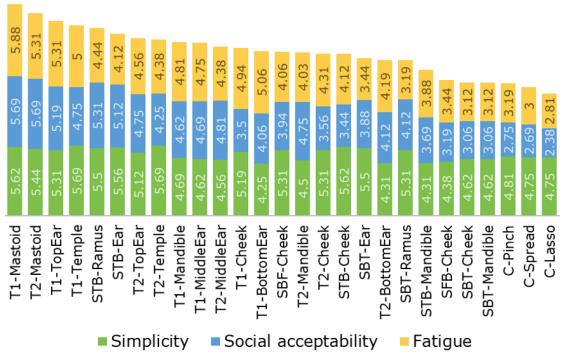


Figure 5: Subjective ratings of all 27 gestures in terms of simplicity, social acceptability, and fatigue (reversed).

3. *Design Consistency*. Prior work has shown that single- and double-click gestures usually appear in a design space together [51], *i.e.*, if an interface supports single-click gesture, it usually supports double-click gesture as well. Therefore, for each single-tap gesture that was eliminated before this point, the corresponding double-tap gesture was removed, and *vice versa*. This eliminated T1-BottomEar, T2-Temple, and T2-TopEar.
4. *Preference*. We used the subjective ratings to decide between the remaining gestures. For each participant, each gesture was ranked from first to last along each of the three criteria. Those rankings were mapped to a score (first = 1, second = 2, *etc.*), and those scores were summed across criteria and participants. We selected the ranked gestures from the top to the bottom, and stopped selection once either of the three criteria had a score below 4. This eliminated SFB-Cheek, STB-Cheek, and SBT-Ear.

The gesture selection procedure resulted in 8 gestures. Our final gesture set had 6 tapping gestures—single- and double-tap on cheek (T1-Cheek and T2-Cheek), mastoid (T1-Mastoid and T2-Mastoid), and middle ear rim (T1-MiddleEar and T2-MiddleEar)—and 2 sliding gestures—top-to-bottom slide on ear rim (STB-Ear) and ramus (STB-Ramus).

STUDY 2: DATA COLLECTION

After finalizing our gesture set, we conducted a second study to collect more instances of those particular gestures and evaluate both the detection and classification accuracy of EarBuddy.

Participants and Apparatus

We invited another 24 participants for this study. All participants used earbuds on a daily basis and were right-hand dominant. Software and hardware errors caused the collected data to be corrupted for six of them. This left us with 18 participants (9 male, 9 female, age = 21.6 $\pm$ 1.3) with valid data. The study was conducted with the same devices and room as the previous study.

Noisy Environment

Handling ambient noise is one of the most salient challenges for sound-based interaction techniques [13, 40]. Therefore, this study was conducted in two sessions: one in a quiet environment (quiet-session) and one in a noisy environment (noisy-session). In the quiet-session, participants sat in the room with minimal noise (38 dB on average). In the noisy-session, standardized noise was generated by a stereo playing a soundtrack at 55 dB [5]. The audio contains standard ambient office noise such as talking, laughing, walking, and typing. The soundtrack was started at a random timestamp for each session to avoid systematic biases.

Design and Procedure

We conducted a within-subject study with a 2×8 factorial design, with *Session* and *Gesture* being the factors. The order of the two sessions and eight gestures was counterbalanced to reduce ordering effects.

Participants were only required to perform the gestures on the right side of their face. They first went through a 5-minute practice phase to familiarize themselves with the eight gestures. During the data collection, participants were asked to perform each gesture 10 times in 5 rounds in both sessions, thus generating 100 examples of each gesture per participant (10 examples/round $\times$ 5 rounds/session $\times$ 2 sessions). To validate the detection accuracy of EarBuddy, participants were instructed to perform gestures in sync with a countdown timer presented on a laptop screen. The timer counted down for 2 seconds, and then participants had another 2 seconds to complete the gesture. Audio was recorded during those 4 seconds to capture audio both with and without gestures. A 1-minute break was placed between each round, during which participants were asked to remove the earbuds and then put them back into their ears to allow for different earbud positioning. The study lasted about 45 minutes.

Annotation

Three researchers examined all of the data to annotate the start- and end-times of each gesture. They removed samples obscured by noise due to some software issue (the audio channel crashed, leading to large noise in the audio sample) and hardware issue during data collection (occasionally the built-in noise cancellation function was activated). 11,147 (77.4%) gesture samples remained in our dataset after filtering.

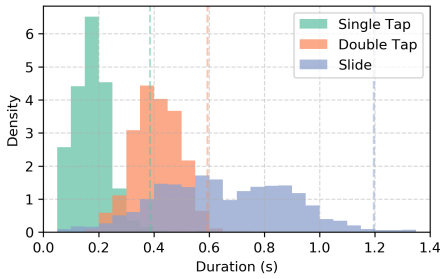


Figure 6: The distribution of three gesture types' duration. The vertical dashed lines indicate the 99th percentile of the duration of that type.

Figure 6 illustrates how long it took for participants to perform the single-tap, double-tap, and slide gestures. Slide gestures took the longest amount of time, with the 99th percentile being 1.2 s. EarBuddy uses this duration as the length of raw audio input for gesture classification. Each gesture is segmented by clipping a 1.2 s-long window of audio data centered at the middle of its annotated range to produce the dataset we use to evaluate gesture detection and classification.

GESTURE DETECTION AND CLASSIFICATION

To test the feasibility of EarBuddy, we trained two models using the data that was collected in this study, one to segment the audio (*i.e.*, gesture detection) and one to recognize the gesture in the segment (*i.e.*, gesture classification).

Gesture Detection

We simulated real-time input by manually applying a 180 ms sliding window across the data with 40 ms steps, the same rate as our implementation of EarBuddy. If more than 50% of the sliding window overlapped with the audio data related to a manually annotated gesture, the window was considered to be a positive gesture detection example; otherwise, the window was negative. This procedure led to 120k positive samples and 252k negative samples for training and testing.

As described earlier, we converted each sliding window to a vector of 20 MFCCs which was used as input for the gesture classifier. A three-layer fully connected neural network binary classifier [55] was trained on the data. The hidden layers had 100, 300, and 50 nodes from input to output, with intermediate dropout layers. Using an 80-20 train-test split on all of the samples produced an overall weighted accuracy of 92.6% (precision: 91.7%, recall: 85.3%). After the classification results were smoothed using the majority-vote algorithm described earlier, 98.2% of the gestures were successfully detected. Among the remaining 1.8% of gestures that were missed, 0.4% were from the silent environment and 1.4% were from the noisy environment, showing that noise complicated gesture detection.

Gesture Classification

The manually annotated gestures were used to assess the optimal performance of EarBuddy's gesture classification performance.

Data Augmentation

Because our dataset was relatively small compared to what is normally desirable for deep-learning, we augmented our dataset by producing similar variations of the collected examples. We did so using the following methods:

- Mixing Augmentation [32]: Noise from two common scenarios—office noise [5] and street noise [9]—were mixed with the raw audio data before they were converted to mel spectrograms.
- Frequency Mask [47]: f consecutive mel frequency channels $[f_0, f_0+f)$ were replaced by their average, where f was chosen from a uniform distribution from 0 to the maximum mel frequency channel v, and f_0 was chosen from $[0, v-f)$.
- Time Mask [47]: t consecutive time steps $[t_0, t_0+t)$ were replaced by their average, where t was chosen from a uniform distribution from 0 to the maximum time τ, and t_0 was chosen from $[0, \tau-t)$.
- Horizontal Flip [22]: The mel spectrogram was flipped horizontally.

Each of the four augmentation methods was independently applied on the raw dataset with a probability of 50% during each epoch of training.

Learning Optimization

Past literature has suggested that stochastic gradient descent (SGD) [50] has better generalization than adaptive optimizers, such as Adam [29, 68]. Therefore, we employed SGD as the optimizer for the training, with the momentum parameter at 0.9 [52] to accelerate convergence and the weight decay regularization parameter at 0.0001 [30] to prevent overfitting. These parameters are commonly adopted for SGD [35]. We combined the linear graduate warm-up method [17] and the cosine-annealing technique [37] to update the learning rate. The learning rate started at 0.01, then climbed up to 0.1 in 20 epochs, then decayed in a cosine curve in the next 400 epochs. Such a learning rate schedule has the advantage of fast (large learning rate at the beginning) and robust (small rate at the end) convergence.

Population Results

We trained two additional models as our baselines:

1. Twenty MFCCs were extracted in 40 ms steps, similarly to what is done for EarBuddy. The mean and standard deviation of the MFCCs were calculated to summarize each gesture as a feature vector with 40 values. Those features were used to train a random forest classifier.
2. A VGG16 Net [58] was trained from scratch on the mel spectrogram images.

We mixed all users' data together and randomly separated them into an 80-20 train-test split. Table 2 provides the classification performance of the baseline models and variations of models. Pre-training with DenseNet, data augmentation, and learning optimization each significantly improved EarBuddy's performance. The final model with all techniques achieved an overall classification accuracy of 95.3% and an F1 score as 0.954 on the test set.

Table 2: Test results of different models and enhancing techniques. Precision, recall, F1, and accuracy values are weighted across gestures.

Model	Prec	Rec	F1	Acc
Random Forest on Means and Std of 20 MFCCs over the window	0.607	0.631	0.620	0.602
VGG16 from scratch with Adam	0.637	0.645	0.640	0.629
Pre-trained VGG16 with Adam	0.769	0.755	0.762	0.761
Pre-trained ResNet with Adam	0.810	0.793	0.802	0.785
Pre-trained DenseNet with Adam	0.807	0.803	0.805	0.809
Pre-trained DenseNet with Adam + Data Augmentation	0.872	0.872	0.872	0.872
Pre-trained DenseNet with SGD + Data Augmentation	0.929	0.893	0.916	0.914
Pre-trained DenseNet with SGD + Data Augmentation + Schedule	**0.956**	**0.951**	**0.954**	**0.953**

Note that these results included data from both the quiet and noisy environments. When we trained our best model configuration using data from each environment separately, EarBuddy had overall classification accuracies of 93.8% and 92.5%, respectively. The decrease in accuracy from the quiet to the noisy environment was expected due to the increased noise in the latter. We also expected a slight drop in accuracy when the data was separated into two halves because there was less data to train each model.

Figure 7 presents the confusion matrix for the eight gestures based on the best model in Table 2. The three double-tap gestures had the highest accuracy (97.3%), followed by the three single-tap gestures (94.4%) and the two sliding gestures (93.1%). The STB-Ramus had the lowest accuracy (91.7%), which may be explained by the relatively lower signal SNR (see Figure 4). That error rate (8.3%) is about two times

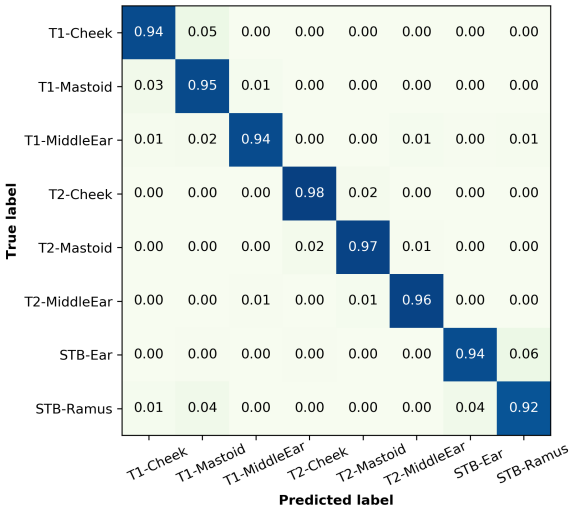


Figure 7: Confusion matrix of the best model on test set. The overall weighted precision, recall, F1 score, and accuracy are 0.956, 0.951, 0.954 and 0.953, respectively.

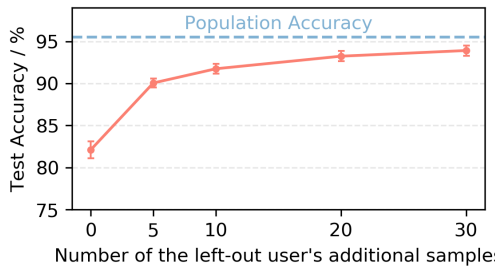


Figure 8: Results with the leave-one-user-out data plus the ignored user's additional samples. Error bar is the standard error. The population accuracy is when all users' data is merged for training.

higher than the average error rate (4.7%). For this reason, we eliminated it and only evaluated the remaining seven gestures in the real-time system in our final evaluation study.

Leave-One-User-Out Results

The audio signal for the same gesture can appear different across users for a couple of reasons: (1) users may perform gestures in unique ways, or (2) users' unique body structure can produce sounds in slightly different ways. To investigate the feasibility of a model that could recognize gestures by new users, we trained our best model configuration using leave-one-user-out cross-validation. Doing so produced an overall accuracy 82.1%, a 13.2% drop from the model that was trained within users.

In a real-world situation, it is realistic to ask a new user to perform each gesture a few times before using the system (*e.g.*, during a tutorial). The system can utilize these samples to apply additional training on a pre-trained model. We tested this approach by saving our leave-one-user-out model and further training it on a small number of examples of each gesture from the ignored user. Figure 8 shows how the inclusion of a small amount of data from the new user can improve model performance. With just five gestures, the performance improved to 90.1%. The performance approached the population test accuracy with additional samples, reaching 93.9% with 30 gestures.

STUDY 3: USABILITY EVALUATION

Our final user study evaluated a real-time implementation of EarBuddy on its performance and usability.

Participants and Apparatus

Twelve participants (8 male, 4 female, age = 21.4 $\pm$ 0.8) from Study 2 were invited back to evaluate the system. The same earbuds and room were used for this study, with the software issue in Study 2 fixed. To test the robustness of our system, the same office audio [5] was employed to simulate a noisy office. We employed an Android phone as the interface where all the gesture results would appear. The phone communicated with the laptop via TCP. The laptop was also used to instruct participants on when to perform which tasks.

Design

We compared our input technique with two baselines in three common application tasks. We conducted a a 3×3 factorial within-subject study with *Task* and *Setup* being the factors.

Figure 9: UI design of the three tasks for evaluation. The two physical buttons on the left edge are used for volume adjustment in music application. And the button on the right edge is used for muting a call.

Table 3: The design of the mapping of EarBuddy gestures and on-screen touch operations for the three applications examined in the user study.

Task	Operation	Earbud Gesture	Touch Gesture
Music	Play/Pause	STB-Ear	Virtual Button Click
Music	Vol Up	T1-Cheek	Physical Button Click
Music	Vol Down	T2-Cheek	Physical Button Click
Music	Next	T1-Mastoid	Virtual Button Click
Music	Previous	T2-Mastoid	Virtual Button Click
Call	Answer	T1-MiddleEar	Virtual Button Click
Call	Reject	T2-MiddleEar	Virtual Button Click
Call	Mute	STB-Ear	Physical Button Click
Notification	Read	STB-Ear	- (Read)
Notification	Open	T1-MiddleEar	Notification Click
Notification	Delete	T2-MiddleEar	Notification Slide

Setups

As our system has the advantage to provide eyes-free interaction, all setups were designed in such a way that the phone screen was locked at the beginning so interactions were not visually available initially. Three setups were involved in the study—one based on EarBuddy and the other two based on touchscreen input:

- **EarBuddy:** The smartphone was placed on the table, and participants used the seven gestures to complete the task.
- **Table:** The smartphone was also put on the table, but this time, participants had to interact with the phone by touchscreen. This required participants to pick up and unlock the phone and then finish the task.
- **Pocket:** Participants were asked to wear a jacket and place the smartphone in the right pocket. They need to remove the phone from the pocket and then complete the task.

Tasks

We designed three common applications for our study, each of which required a different set of actions to complete operations:

- **Music Player:** Participants controlled music with five actions: play/pause, volume up, volume down, next song, and previous song.
- **Phone Call:** When a phone call came in, participants could either answer, reject, or mute the call.
- **Notifications:** Participants consumed a notification by either hearing it in the EarBuddy setup or by picking up the phone and reading it in the other two setups. They could either open the notification for more details or delete it.

Figure 9 shows the interfaces for the three tasks. Table 3 shows the mapping between gestures and smartphone operations, which we pre-determined using pilot testing.

Details

We used a Latin square to assign the ordering of tasks and interfaces. Within each task, the order of operations were randomized and each operation appeared three times. We logged the completion time of every operation and three types of errors: (1) user errors, where participants performed the wrong gesture or clicked on a wrong button; (2) segmentation errors, where participants performed a gesture but EarBuddy failed to recognize it (false negative) or EarBuddy mistakenly detected a gesture when none was performed (false positive); and (3) recognition errors, where EarBuddy did not correctly recognize a gesture that participants performed. For touch interaction, the detection and recognition errors were assumed to be zero. Note that if a user did not complete an operation in 20 seconds, the operation would be skipped.

After completing the three tasks in each setup, participants completed a 7-point Likert scale NASA-TLX questionnaire [21] to assess the perceived workload of the task and the effectiveness of the gestures.

Procedure

Participants first familiarized themselves with the three setups. The experimenter then introduced the three applications to the participants. As EarBuddy provides a new interface that users have never experienced before, we included a 3-minute practice phase for each combination of setup and task to allow participants to familiarize themselves with the gesture mappings. Participants followed the instructions on a laptop screen to complete the required tasks for each setup. Participants were asked to complete the task as soon as possible after a beep from the laptop so that each action could be timed. There was a one-minute break between each task. After each setup, participants filled out the NASA-TLX questionnaire for the setup. The study took about 40 minutes.

Results

Participants were able to easily remember the mapping between EarBuddy gestures and setup actions since nobody performed an incorrect gesture. Meanwhile, our system had a low segmentation error rate (4.1% of gestures were missed) and a low recognition error rate (6.3% of gestures were incorrectly classified).

Figure 10 top shows the average time participants took to complete each of the three setups. As the data violated normality and homoscedasticity assumption, we used analysis of variance on a generalized linear mixed model (GLMM) with Gamma family link function [43]. The results indicate a significant effect on *Setup* ($\chi^2(2) = 73.0, p < 0.001$), but neither on *Task* ($\chi^2(2) = 2.1, p = 0.34$) nor the interaction between *Setup* and *Task* ($\chi^2(4) = 3.2, p = 0.52$). Three post-hoc paired-samples z-tests on *Setup*, corrected with Holm's sequential Bonferroni procedure, indicate that the setups were all significantly different ($p < 0.001$). Participants completed the *EarBuddy* setup 33.9% faster than the *Table* setup, and 56.2% faster than *Pocket* setup.

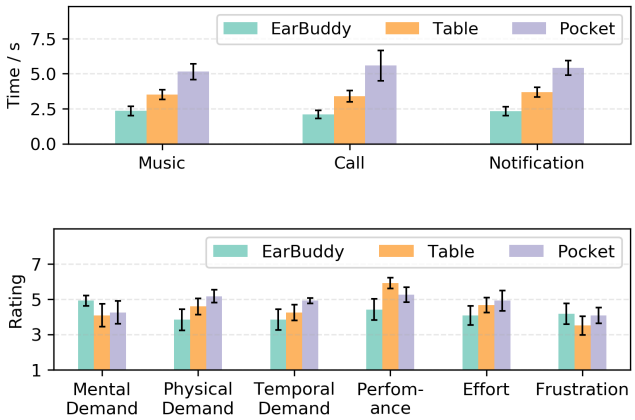


Figure 10: Results of the evaluation study. Top) Time to complete the tasks. Bottom) Subjective ratings of the three setups

Participants' subjective feedback of EarBuddy was also positive, as presented in the bottom of Figure 10. A generalized linear mixed-effects model analysis of variance (with ordinal family link function) on each question indicates a significant effect on *Setup* for physical demand, ($\chi^2(2) = 7.7, p < 0.05$), performance ($\chi^2(2) = 5.7, p < 0.05$), and effort ($\chi^2(2) = 5.8, p < 0.05$). For these three questions, three post-hoc paired-samples Wilcox tests with Bonferroni procedure correction indicate that *EarBuddy* has lower physical demand ($V = 2, p < 0.05$) and requires less effort ($V = 5, p < 0.05$) than *Pocket*, and that *EarBuddy* has better performance than *Table* ($V = 6.5, p < 0.05$).

DISCUSSION

We discuss insights on gesture design, how EarBuddy can be generalized to new users, potential hardware generalizability and applications, as well as limitations and future work.

Gesture Design for Face and Ear Interaction

We discovered a few insights from our first study when users explored the entire design space. Users generally preferred tapping gestures over sliding gestures. Tapping gestures have similar average simplicity ratings compared to sliding gestures (both 5.0), but better social acceptability (4.6 vs. 3.9) and fatigue ratings (4.8 vs. 3.7). Moreover, simple sliding gestures were preferred over complex sliding gestures as the latter were viewed to be socially inappropriate (2.6) and fatiguing (3.0). Users also preferred top-to-bottom and back-to-front sliding over the reverse directions. The top-to-bottom/back-to-front gestures had higher ratings in all three attributes (simplicity: 5.3 vs. 4.8, social acceptance: 4.3 vs. 3.6, fatigue: 4.1 vs. 3.3). This may be due to the fact that moving the finger forward and downward works with gravity rather than against it, returning the arm to a more natural position than the reverse.

As for the signal quality, tapping on facial skin generated louder sounds compared to sliding. Gestures on the ears also produced louder sounds than gestures behind the ears, followed by gestures on the cheek, temple, and mandible. This trend is mainly due to the distance between the microphone and the gesture surface. Putting these facts together, tapping gestures on the ear rim produced the strongest signal. Both

user preferences and signal quality should be considered by researchers and designers in the future.

Improving Performance for New Users

Individuals have unique ways of performing different gestures. When performing a double-tap, for example, some users tap harder the first time than the second, while others do the opposite. Two users may also tap at slightly different positions on the cheek when performing a tapping gesture. Because of these differences, the average accuracy after leave-one-user-out training (82.1%) was lower than the accuracy after training across all users (95.3%). However, as shown in Figure 8, using just five examples per gesture from the new user raises the accuracy to 90.1%. This illustrates that introducing a warm-up phase for a new user can efficiently improve the model's performance, which can be delivered in a clever way to avoid burdening new users. *E.g.,* training examples can be collected while the user walks through a tutorial on which gestures are supported by a given interface.

Generalizability on Hardware

Our studies of EarBuddy used a single pair of in-ear Samsung Gear IconX earbuds. However, commercial earbuds have various form factors that could lead to different acoustic responses. For example, some earbuds are kept in place by clips that wrap around the ear (*e.g.*, Powerbeats Pro [6]), whereas others have a microphone that sticks out like a headset (*e.g.*, Bose SoundSport Wireless [2]). Increasing the distance between the microphone and the different gesture surfaces can weaken the audio signal intensity (SNR). However, having a microphone that is effectively in the air can better detect near-audible sounds that are transmitted through the air. Earbuds that have the microphone embedded within their main housing may do a better job of distinguishing infrasound because of their close contact with the skin.

EarBuddy is compatible with other hardware that has a fixed microphone location around the face/ear as long as the captured audio has a sufficient SNR; for the 8 gestures we chose in Study 2, the average SNR is 10.3 dB. Examples of devices that could potentially be used with EarBuddy include: Bose headphones [2] have microphones in their main housing; the HTC Vive [4] has built-in microphones at the bottom center of the headset; and the HoloLens 2 [3] has two microphone arrays near the nose pad. Although further investigation is needed, the microphone position of these devices are close to the face and ears, thus promising for use in detecting gestures.

Potential Applications

EarBuddy can provide an eyes-free, socially acceptable input method. Users can interact with devices in a more subtle way, *e.g.*, during a meeting, in a library, and in an office. It is suitable for quick reactions such as issuing commands and handling notifications, as illustrated by the application examples in our evaluation study. Moreover, EarBuddy can serve as a convenient input method when a user is using the device in a hands-free mode, such as when watching videos, cooking, *etc.* However, EarBuddy is not suitable for repeated, continuous interactions, *e.g.*, text entry and interface scrolling. It also offers potential use cases in AR/VR. Rather than needing additional input widgets on the headsets, controllers or 3D finger tracking, EarBuddy can be embedded in a headset without additional hardware modification.

Limitations and Future Work

There are some important limitations of our work. First, the hardware we used only allowed the microphone on one side to be activated at a time, likely for better battery life. This prevented us from evaluating gestures on the left and right side of the face simultaneously. There is some work that deals with the problem by introducing a second smart device (*e.g.*, [76]). In addition, we eliminated a number of data (22.6%) from the Study 2. This might be caused by built-in noise cancellation functions. We will investigate these issues in future work.

Second, we only included noise from an office when simulating a noisy environment during data collection and evaluation, but there are other common noise types. One particularly intriguing source of noise was random face touches (*e.g.*, scratching one's face), which could have generated explicit false positives for gesture detection. Generalization with this noise remains as an open issue. We believe that personalized models work better due to differences in noise, physiology and gestures, but a one-fits-all model could also achieve good performance if trained on a large population. We plan to investigate this question by collecting data from more users to enhance model robustness.

Regarding future work, EarBuddy currently only leverages the microphone sensor on wireless earbuds. Commercial wireless earbuds also usually contain other sensors such as an IMU, which may provide additional information that can enhance recognition performance. Moreover, earbuds that rely on bone conduction technology (*e.g.*, AfterShokz Aeropex [1]) provide a unique opportunity for facial gestures. We hope to include these additional data sources in future iterations of EarBuddy.

CONCLUSION

We propose EarBuddy, a novel input system using commercially available wireless earbuds to measure the sound generated by contact between the finger and skin on the face and ears. EarBuddy allows users to interact with any device by simply tapping or sliding on the face and ears. We developed a design space with 27 gestures and conducted a user study with 16 participants to select a subset of gestures optimized for user preference, social acceptability, and microphone detectability. We then conducted a study with 20 users to collect data for the eight gestures in both quiet and noisy environments. Machine learning models were trained for gesture detection and classification, the latter of which was able to identify gestures with 95.3% accuracy. We embedded the models into a real-time system to conduct another usability evaluation study with 12 users. The results indicate that EarBuddy accelerated input tasks by 33.9–56.2%. Users also preferred EarBuddy over touchscreen alternatives since EarBuddy allowed them to interact with devices easily, conveniently, and naturally. Our work demonstrates how earbud-based sensing can be used to enable novel interaction techniques, and we hope to see other researchers leveraging earbuds and other commercial wearables to support novel forms of interaction.

Acknowledgement
This work is supported by Grant 90DPGE0003-01 from the National Institute on Disability, Independent Living and Rehabilitation Research, NSF IIS 1836813, the National Key Research and Development Plan under Grant No. 2016YFB1001200, the Natural Science Foundation of China under Grant No. 61572276, 61672314 and 61902208, China Postdoctoral Science Foundation under Grant No. 2019M660647, and also by Beijing Key Lab of Networked Multimedia. We thank Xin Liu for the advise on deep learning.

REFERENCES

[1] 2019a. Aftershokz Aeropex. (2019). https://aftershokz:com/collections/wireless/products/aeropex.

[2] 2019b. Bose SoundSport Wireless. (2019). https://www:bose:com/en_us/products/headphones/earphones/soundsport-wireless:html.

[3] 2019c. HoloLens2. (2019). https://www.microsoft.com/en-us/hololens.

[4] 2019d. HTC Vive. (2019). https://www.vive.com/us/.

[5] 2019e. Office Noise. (2019). https://www:youtube:com/watch?v=D7ZZp8XuUTE.

[6] 2019f. PowerBeats Pro. (2019). https://www:beatsbydre:com/earphones/powerbeats-pro.

[7] 2019g. Samsung Gear IconX. (2019). https://www:samsung:com/us/support/owners/product/geariconx-2018.

[8] 2019h. Sony WF-1000XM3. (2019). https://www:sony:com/electronics/truly-wireless/wf-1000xm3.

[9] 2019i. Street Noise. (2019). https://www:youtube:com/watch?v=8s5H76F3SIs&t=10517s.

[10] Daniel Ashbrook, Carlos Tejada, Dhwanit Mehta, Anthony Jiminez, Goudam Muralitharam, Sangeeta Gajendra, and Ross Tallents. 2016. Bitey: An Exploration of Tooth Click Gestures for Hands-free User Interface Control. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '16)*. ACM, New York, NY, USA, 158–169. DOI:http://dx.doi.org/10.1145/2935334.2935389

[11] Andrew Bragdon, Eugene Nelson, Yang Li, and Ken Hinckley. 2011. Experimental Analysis of Touch-screen Gesture Designs in Mobile Environments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 403–412. DOI: http://dx.doi.org/10.1145/1978942.1979000

[12] Mingshi Chen, Panlong Yang, Jie Xiong, Maotian Zhang, Youngki Lee, Chaocan Xiang, and Chang Tian. 2019. Your Table Can Be an Input Panel: Acoustic-based Device-Free Interaction Recognition. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 1, Article 3 (March 2019), 21 pages. DOI: http://dx.doi.org/10.1145/3314390

[13] Alain Dufaux, Laurent Besacier, Michael Ansorge, and Fausto Pellandini. 2000. Automatic sound detection and recognition for noisy environment. In *2000 10th European Signal Processing Conference*. IEEE, 1–4.

[14] Antti J Eronen, Vesa T Peltonen, Juha T Tuomi, Anssi P Klapuri, Seppo Fagerlund, Timo Sorsa, Gaëtan Lorho, and Jyri Huopaniemi. 2005. Audio-based context recognition. *IEEE Transactions on Audio, Speech, and Language Processing* 14, 1 (2005), 321–329.

[15] Pasquale Foggia, Nicolai Petkov, Alessia Saggese, Nicola Strisciuglio, and Mario Vento. 2015. Reliable detection of audio events in highly noisy environments. *Pattern Recognition Letters* 65 (2015), 22–28.

[16] Emily Gillespie. 2018. Analyst Says AirPods Sales Will Go Through the Roof Over the Next Few Years, Report Says. (Dec 2018).

[17] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. 2017. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677* (2017).

[18] Sean Gustafson, Christian Holz, and Patrick Baudisch. 2011. Imaginary Phone: Learning Imaginary Interfaces by Transferring Spatial Memory from a Familiar Device. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. ACM, New York, NY, USA, 283–292. DOI: http://dx.doi.org/10.1145/2047196.2047233

[19] Sean G. Gustafson, Bernhard Rabe, and Patrick M. Baudisch. 2013. Understanding Palm-based Imaginary Interfaces: The Role of Visual and Tactile Cues when Browsing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 889–898. DOI: http://dx.doi.org/10.1145/2470654.2466114

[20] Chris Harrison, Shilpa Ramamurthy, and Scott E. Hudson. 2012. On-body Interaction: Armed and Dangerous. In *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction (TEI '12)*. ACM, New York, NY, USA, 69–76. DOI:http://dx.doi.org/10.1145/2148131.2148148

[21] Sandra G Hart and Lowell E Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. In *Advances in Psychology*. Vol. 52. Elsevier, 139–183.

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 770–778.

[23] Shawn Hershey, Sourish Chaudhuri, Daniel PW Ellis, Jort F Gemmeke, Aren Jansen, R Channing Moore, Manoj Plakal, Devin Platt, Rif A Saurous, Bryan Seybold, and others. 2017. CNN architectures for large-scale audio classification. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 131–135.

[24] Da-Yuan Huang, Liwei Chan, Shuo Yang, Fan Wang, Rong-Hao Liang, De-Nian Yang, Yi-Ping Hung, and Bing-Yu Chen. 2016. DigitSpace: Designing Thumb-to-Fingers Touch Interfaces for One-Handed and Eyes-Free Interactions. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 1526–1537. DOI:http://dx.doi.org/10.1145/2858036.2858483

[25] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4700–4708.

[26] Yasha Iravantchi, Yang Zhang, Evi Bernitsas, Mayank Goel, and Chris Harrison. 2019. Interferi: Gesture Sensing Using On-Body Acoustic Interferometry. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, New York, NY, USA, Article 276, 13 pages. DOI: http://dx.doi.org/10.1145/3290605.3300506

[27] Hsin-Liu (Cindy) Kao, Artem Dementyev, Joseph A. Paradiso, and Chris Schmandt. 2015. NailO: Fingernails As an Input Surface. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 3015–3018. DOI: http://dx.doi.org/10.1145/2702123.2702572

[28] Takashi Kikuchi, Yuta Sugiura, Katsutoshi Masai, Maki Sugimoto, and Bruce H. Thomas. 2017. EarTouch: Turning the Ear into an Input Surface. In *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '17)*. ACM, New York, NY, USA, Article 27, 6 pages. DOI: http://dx.doi.org/10.1145/3098279.3098538

[29] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*. 1097–1105.

[31] Nicholas D. Lane, Petko Georgiev, and Lorena Qendro. 2015. DeepEar: Robust Smartphone Audio Sensing in Unconstrained Acoustic Environments Using Deep Learning. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*. ACM, New York, NY, USA, 283–294. DOI: http://dx.doi.org/10.1145/2750858.2804262

[32] Gierad Laput, Karan Ahuja, Mayank Goel, and Chris Harrison. 2018. Ubicoustics: Plug-and-Play Acoustic Activity Recognition. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology (UIST '18)*. ACM, New York, NY, USA, 213–224. DOI: http://dx.doi.org/10.1145/3242587.3242609

[33] Gierad Laput, Yang Zhang, and Chris Harrison. 2017. Synthetic Sensors: Towards General-Purpose Sensing. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 3986–3999. DOI: http://dx.doi.org/10.1145/3025453.3025773

[34] Hyunchul Lim, Jungmin Chung, Changhoon Oh, SoHyun Park, Joonhwan Lee, and Bongwon Suh. 2018. Touch+Finger: Extending Touch-based User Interface Capabilities with "Idle" Finger Gestures in the Air. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology (UIST '18)*. ACM, New York, NY, USA, 335–346. DOI: http://dx.doi.org/10.1145/3242587.3242651

[35] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision*. 2980–2988.

[36] Roman Lissermann, Jochen Huber, Aristotelis Hadjakos, and Max Mühlhäuser. 2013. EarPut: Augmenting Behind-the-ear Devices for Ear-based Interaction. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems (CHI EA '13)*. ACM, New York, NY, USA, 1323–1328. DOI: http://dx.doi.org/10.1145/2468356.2468592

[37] Ilya Loshchilov and Frank Hutter. 2016. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983* (2016).

[38] Hao Lü and Yang Li. 2011. Gesture Avatar: A Technique for Operating Mobile User Interfaces Using Gestures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 207–216. DOI: http://dx.doi.org/10.1145/1978942.1978972

[39] Hong Lu, Wei Pan, Nicholas D. Lane, Tanzeem Choudhury, and Andrew T. Campbell. 2009. SoundSense: Scalable Sound Sensing for People-centric Applications on Mobile Phones. In *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services (MobiSys '09)*. ACM, New York, NY, USA, 165–178. DOI: http://dx.doi.org/10.1145/1555816.1555834

[40] Héctor A. Cordourier Maruri, Paulo Lopez-Meyer, Jonathan Huang, Willem Marco Beltman, Lama Nachman, and Hong Lu. 2018. V-Speech: Noise-Robust Speech Capturing Glasses Using Vibration Sensors. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 4, Article 180 (Dec. 2018), 23 pages. DOI: http://dx.doi.org/10.1145/3287058

[41] Katsutoshi Masai, Yuta Sugiura, Masa Ogata, Kai Kunze, Masahiko Inami, and Maki Sugimoto. 2016. Facial Expression Recognition in Daily Life by Embedded Photo Reflective Sensors on Smart Eyewear. In *Proceedings of the 21st International Conference on*

Intelligent User Interfaces (IUI '16). ACM, New York, NY, USA, 317–326. DOI: http://dx.doi.org/10.1145/2856767.2856770

[42] Katsutoshi Masai, Yuta Sugiura, and Maki Sugimoto. 2018. FaceRubbing: Input Technique by Rubbing Face Using Optical Sensors on Smart Eyewear for Facial Expression Recognition. In *Proceedings of the 9th Augmented Human International Conference (AH '18)*. ACM, New York, NY, USA, Article 23, 5 pages. DOI: http://dx.doi.org/10.1145/3174910.3174924

[43] Charles E McCulloch and John M Neuhaus. 2005. Generalized linear mixed models. *Encyclopedia of Biostatistics* 4 (2005).

[44] Christian Metzger, Matt Anderson, and Thad Starner. 2004. Freedigiter: A contact-free device for gesture control. In *Eighth International Symposium on Wearable Computers*, Vol. 1. IEEE, 18–21.

[45] Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning*. 807–814.

[46] Sinno Jialin Pan and Qiang Yang. 2009. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* 22, 10 (2009), 1345–1359.

[47] Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779* (2019).

[48] Patrick Parzer, Adwait Sharma, Anita Vogl, Jürgen Steimle, Alex Olwal, and Michael Haller. 2017. SmartSleeve: Real-time Sensing of Surface and Deformation Gestures on Flexible, Interactive Textiles, Using a Hybrid Gesture Detection Pipeline. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA, 565–577. DOI: http://dx.doi.org/10.1145/3126594.3126652

[49] Tauhidur Rahman, Alexander Travis Adams, Mi Zhang, Erin Cherry, Bobby Zhou, Huaishu Peng, and Tanzeem Choudhury. 2014. BodyBeat: a mobile system for sensing non-speech body sounds.. In *MobiSys*, Vol. 14. Citeseer, 2–13.

[50] Herbert Robbins and Sutton Monro. 1951. A stochastic approximation method. *The Annals of Mathematical Statistics* (1951), 400–407.

[51] Sami Ronkainen, Jonna Häkkilä, Saana Kaleva, Ashley Colley, and Jukka Linjama. 2007. Tap input as an embedded interaction method for mobile devices. In *Proceedings of the 1st international conference on Tangible and embedded interaction*. ACM, 263–270.

[52] David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, and others. 1988. Learning representations by back-propagating errors. *Cognitive Modeling* 5, 3 (1988), 1.

[53] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, and others. 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision* 115, 3 (2015), 211–252.

[54] Stan Salvador and Philip Chan. 2007. Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis* 11, 5 (2007), 561–580.

[55] Jürgen Schmidhuber. 2015. Deep learning in neural networks: An overview. *Neural networks* 61 (2015), 85–117.

[56] Stefan Schneegass and Alexandra Voit. 2016. GestureSleeve: Using Touch Sensitive Fabrics for Gestural Input on the Forearm for Controlling Smartwatches. In *Proceedings of the 2016 ACM International Symposium on Wearable Computers (ISWC '16)*. ACM, New York, NY, USA, 108–115. DOI: http://dx.doi.org/10.1145/2971763.2971797

[57] Marcos Serrano, Barrett M. Ens, and Pourang P. Irani. 2014. Exploring the Use of Hand-to-face Input for Interacting with Head-worn Displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 3181–3190. DOI: http://dx.doi.org/10.1145/2556288.2556984

[58] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[59] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[60] Lee Stearns, Uran Oh, Leah Findlater, and Jon E. Froehlich. 2018. TouchCam: Realtime Recognition of Location-Specific On-Body Gestures to Support Users with Visual Impairments. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 4, Article 164 (Jan. 2018), 23 pages. DOI:http://dx.doi.org/10.1145/3161416

[61] Johannes A Stork, Luciano Spinello, Jens Silva, and Kai O Arras. 2012. Audio-based human activity recognition using non-markovian ensemble voting. In *2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication*. IEEE, 509–514.

[62] Emi Tamaki, Takashi Miyak, and Jun Rekimoto. 2010. BrainyHand: A Wearable Computing Device Without HMD and It's Interaction Techniques. In *Proceedings of the International Conference on Advanced Visual Interfaces (AVI '10)*. ACM, New York, NY, USA, 387–388. DOI: http://dx.doi.org/10.1145/1842993.1843070

[63] Katia Vega and Hugo Fuks. 2013. Beauty Tech Nails: Interactive Technology at Your Fingertips. In *Proceedings of the 8th International Conference on Tangible, Embedded and Embodied Interaction (TEI '14)*. ACM, New York, NY, USA, 61–64. DOI: http://dx.doi.org/10.1145/2540930.2540961

[64] Craig Villamor, Dan Willis, and Luke Wroblewski. 2010. Touch gesture reference guide. *Touch Gesture Reference Guide* (2010).

[65] Cheng-Yao Wang, Min-Chieh Hsiu, Po-Tsung Chiu, Chiao-Hui Chang, Liwei Chan, Bing-Yu Chen, and Mike Y. Chen. 2015. PalmGesture: Using Palms As Gesture Interfaces for Eyes-free Input. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '15)*. ACM, New York, NY, USA, 217–226. DOI:http://dx.doi.org/10.1145/2785830.2785885

[66] Ruolin Wang, Chun Yu, Xing-Dong Yang, Weijie He, and Yuanchun Shi. 2019. EarTouch: Facilitating Smartphone Use for Visually Impaired People in Mobile and Public Scenarios. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, New York, NY, USA, Article 24, 13 pages. DOI:http://dx.doi.org/10.1145/3290605.3300254

[67] Martin Weigel, Aditya Shekhar Nittala, Alex Olwal, and Jürgen Steimle. 2017. SkinMarks: Enabling Interactions on Body Landmarks Using Conformal Skin Electronics. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 3095–3105. DOI: http://dx.doi.org/10.1145/3025453.3025704

[68] Ashia C Wilson, Rebecca Roelofs, Mitchell Stern, Nati Srebro, and Benjamin Recht. 2017. The marginal value of adaptive gradient methods in machine learning. In *Advances in Neural Information Processing Systems*. 4148–4158.

[69] Xuhai Xu, Ahmed Hassan Awadallah, Susan T. Dumais, Farheen Omar, Bogdan Popp, Robert Routhwaite, and Farnaz Jahanbakhsh. 2020. Understanding User Behavior For Document Recommendation. In *The World Wide Web Conference (WWW '20)*. Association for Computing Machinery, New York, NY, USA, 7. DOI: http://dx.doi.org/10.1145/3366423.3380071

[70] Xuhai Xu, Alexandru Dancu, Pattie Maes, and Suranga Nanayakkara. 2018. Hand Range Interface: Information Always at Hand with a Body-centric Mid-air Input Surface. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '18)*. ACM, New York, NY, USA, Article 5, 12 pages. DOI: http://dx.doi.org/10.1145/3229434.3229449

[71] Xuhai Xu, Chun Yu, Anind K. Dey, and Jennifer Mankoff. 2019. Clench Interface: Novel Biting Input Techniques. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, New York, NY, USA, Article 275, 12 pages. DOI: http://dx.doi.org/10.1145/3290605.3300505

[72] Xuhai Xu, Chun Yu, Yuntao Wang, and Yuanchun Shi. 2020. Recognizing Unintentional Touch on Interactive Tabletop. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 1 (March 2020), 27. DOI: http://dx.doi.org/10.1145/3381011

[73] Koki Yamashita, Takashi Kikuchi, Katsutoshi Masai, Maki Sugimoto, Bruce H. Thomas, and Yuta Sugiura. 2017. CheekInput: Turning Your Cheek into an Input Surface by Embedded Optical Sensors on a Head-mounted Display. In *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology (VRST '17)*. ACM, New York, NY, USA, Article 19, 8 pages. DOI:http://dx.doi.org/10.1145/3139131.3139146

[74] Yukang Yan, Chun Yu, Wengrui Zheng, Ruining Tang, Xuhai Xu, and Yuanchun Shi. 2020. FrownOnError: Interrupting Responses from Smart Speakers by Facial Expressions. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 14. DOI: http://dx.doi.org/10.1145/3313831.3376810

[75] Koji Yatani and Khai N. Truong. 2012. BodyScope: A Wearable Acoustic Sensor for Activity Recognition. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp '12)*. ACM, New York, NY, USA, 341–350. DOI: http://dx.doi.org/10.1145/2370216.2370269

[76] Yingtian Shi Minxing Xie Yukang Yan, Chun Yu. 2019. PrivateTalk: Activating Voice Input with Hand-On-Mouth Gesture Detected by Bluetooth Earphones. In *Proceedings of the 32st Annual ACM Symposium on User Interface Software and Technology (UIST '19)*. ACM, New York, NY, USA, 581–593. DOI: http://dx.doi.org/10.1145/3332165.3347950

[77] Cheng Zhang, Qiuyue Xue, Anandghan Waghmare, Ruichen Meng, Sumeet Jain, Yizeng Han, Xinyu Li, Kenneth Cunefare, Thomas Ploetz, Thad Starner, Omer Inan, and Gregory D. Abowd. 2018. FingerPing: Recognizing Fine-grained Hand Poses Using Active Acoustic On-body Sensing. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 437, 10 pages. DOI: http://dx.doi.org/10.1145/3173574.3174011