Ui-Ear: Towards Pervasive On-face Gesture Recognition Through On-ear Vibration Sensing

Paper ID # 228

***Abstract*—With the convenient design and prolific functionalities, wireless earbuds are fast penetrating in our daily life and taking over the place of traditional wired earphones. The sensing capabilities of wireless earbuds have attracted great interests of researchers on exploring them as a new interface for human computer interactions. However, due to its extremely compact size, the interaction on the body of the earbuds is limited and not convenient. In this paper, we propose *Ui-Ear*, a new on-face gesture recognition system to enrich interaction maneuvers for wireless earbuds. *Ui-Ear* exploits the sensing capability of Inertial Measurement Units (IMUs) pervasively provided on most of the budget and high-end wireless earbuds to extend the interaction to the skin of the face near ears. The accelerometer and gyroscope in IMUs perceive dynamic vibration signals induced by on-face touching and moving, which brings rich maneuverability. To demonstrate the feasibility of the system, we define seven different on-face gestures and design an end-to-end learning approach based on Convolutional Neural Networks (CNNs) for classifying different gestures. To further improve the generalization capability of the system, adversarial learning mechanism is incorporated in the offline training process to suppress the user-specific features while enhancing gesture-related features. We recruit 20 participants and collect a realworld datasets to evaluate the recognition accuracy. The extensive evaluations show that the average recognition accuracy of *Ui-Ear* is over 95% and 82.3% respectively in the user-dependent and user-independent tasks. Moreover, we also show that the pre-trained model (learned from user-independent task) can be fine-tuned with only few training samples of the target user to achieve relatively high recognition accuracy. At last, we implement the personalization and recognition components of *Ui-Ear* on an off-the-shelf Android smartphone to evaluate its system overhead. The results demonstrate *Ui-Ear* can achieve real-time response while only brings trivial energy consumption on smartphones.**

I. Introduction

With the development of chip design and manufacturing, the sizes of the electronic units are being miniaturized rapidly. The wireless chips (e.g., Huawei Kirin A1 [1] and Apple H1 [2]) and different types of sensors are encapsulated to create "true" wireless earbuds, which is one of the most successful inventions in recent years. Due to its convenient operation and rich functionality, wireless earbuds are quickly grabbing the market of traditional earphones and are becoming the first choice for most of the users in their daily use, especially for sports and fitness. As reported by Polaris Market Research, the global market size of true wireless earbuds will reach $14.51 billion by 2028.[1]

The competition on the earbuds market is fierce, the choice of users are normally made according to the sound quality, noise cancellation capability, etc. Besides, the design of interactive interface is an important factor to be reckoned with having a significant impact on user experience. For example, when listening to the music or watching multimedia content, it will be greatly convenient if the users are able to directly interact with their earbuds to start/pause, switch between titles, turn up/down the column without working on their master devices. Different from traditional wired earphones or bulky headphones, true wireless earbuds normally do not feature dedicated interaction hardware, e.g., line controllers, for users to interact with their master devices easily. To provide user-device interactive functionalities, most of the developers of the wireless earbuds coincidentally choose the similar solution, i.e., tapping on the body of the earbuds. Most of the wireless earbuds facilitate Inertial Measurement Units (IMUs) to detect users' tapping once or twice, while Airpods Pro [3] utilizes force sensor which provides higher detection accuracy. However, the tiny surface of the earbuds place a huge limit on the richness of the skill sets of the interactions between users and earbuds.

Utilizing the face area near the ears is an effective way to significantly enlarge the interactive surface so that more on-face gestures can be included in the user-earbuds interaction skill sets. For example, EarBuddy [4] facilitates the microphones on most of the wireless earbuds to record the acoustic signals generated by finger sliding on face in different patterns to distinguish different on-face gestures. However, the microphone is open for all acoustic signals and therefore is susceptible to environment interference. OESense [5] utilizes the inward-facing microphone on the noise-canceling earbuds to capture the sound generated by on-face gesture and transmitted through skin. The inward-facing microphone will not be affected by environment interference, however, it is normally only available on expensive noise-canceling earbuds but not the budget earbuds. At last, capacitive [6] and vision-based sensing [7] solutions are also explored, however, the dependence on the bespoke hardware restrains their application on the commercialized wireless earbuds.

In this paper, we aim to design an on-face gesture recognition system which can be universally applied on most of the wireless earbuds including both budget and high-end ones. To this end, we propose *Ui-Ear*, an on-face finger gesture recognition system by facilitating the sensing capability of IMUs on most of the mainstream wireless earbuds. As shown in Fig. 1, *Ui-Ear* extends the user-earbuds interaction to the face area to significantly enrich the maneuvers of user-device interaction. It is worth noting that, the on-face and on-device

[1]https://www.prnewswire.com/news/polaris-market-research/

gestures are not mutually exclusive. They can work together to form the new skill sets of the user-earbuds interaction.

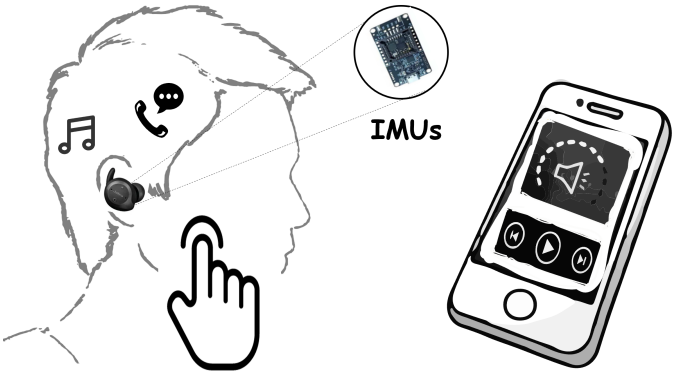

Fig. 1. A typical use case of *Ui-Ear*: the user performs on-face finger gestures to control the music app on smartphones

The contributions of this work are multi-folds:

- We propose, *Ui-Ear*, an on-face finger gesture recognition system to enrich the skill sets of user-earbuds interaction based on sensor reading of IMUs which are available on most of wireless earbuds. To the best of our knowledge, this is the first piece of work facilitating vibration sensing for on-face gesture recognition.
- We propose a Domain Adversarial Learning (DAL) framework for end-to-end feature extraction to deal with the domain generalization issue in the gesture recognition. The DAL framework fosters the features independent of users by suppressing the user-dependent information. The extensive evaluations on realworld datasets show that the DAL framework can effectively improve the recognition accuracy with or without training samples from the target user.
- To further improve the accuracy on target user, we employ the transfer learning to fine-tune the classifier with few training samples from the target domain. The evaluation results demonstrate, the classifier with the feature extractor learned from DAL framework can reduce the number of training samples needed for fine-tuning, therefore, user's training efforts on personalization can be significantly reduced.
- At last, we implement the inference and personalization components of *Ui-Ear* on an off-the-shelf Android smartphone. Experimental tests show that the system overhead of *Ui-Ear* on smartphones is trivial, e.g., it only takes less than 1.3ms to perform the gesture recognition on the resource-constrained device.

The rest of this paper is organized as follows. In Section II, we review the papers related to our work. Section III introduces the system design of *Ui-Ear* and the DAL framework in feature learning. The evaluation and system implementation are in Section IV and Section V respectively. Finally, Section VI concludes the whole paper.

II. RELATED WORK

Functionally, our work is related to smart device interaction expansion. We separate the research on smart device interaction expansion into *device-based interaction* and *skin-based interaction* depending on the surface where the human perform actions.

The *device-based interaction* exploited the smart device itself or the solid medium around the device as the input surface. Simply, some smartphone manufacturers added additional physical buttons to enhance users gaming experience [8]. However, this increased manufacturing costs. Based on acoustic signals and IMUs signals, other researchers utilized the existing internal sensors to perceive surface gestures of smartphones [9]–[11]. Since smaller volume and no screen, wrist-worn devices such as smartwatches and fitness bands have greater demand for interaction expansion. WatchIt [12] added thin resistive bands to wristbands to provide precise continuous over list scrolling control. WatchOut [13] extended input modalities to the watch body e.g., case, bezel, and band, etc. Based on smartwatches' internal IMUs, the SVM classifier can distinguish up to eight distinct gestures. In addition to using the smart device itself as the operation interface, some studies broaden the operation interface to the solid medium around the device. [14]–[16] used the smartphones' microphones or an additional piezo sensor to perceive the keystrokes on the nearby paper keyboard.

Compared with *device-based interaction*, our work is more related to *skin-based interaction*, which token the human's body surface as an interface. Some recent work [17], [18] pointed out that an adult's average body surface area is $1.73m^2$, which was greater than the touch screen of the smartphone by 400 times. When the human's finger tapped on the skin, the impact created various signals, which were propagated by the soft tissues of the skin surface, and eventually can be received by various sensors. Based on the electrical signal, SkinTrack [19] used capacitive sensors to track various gestures on the arm skin. Similarly, Earput [6] used capacitive sensing based on electrodes to provide ear-based interaction. However, these works depended on additional hardware, which cannot be directly applied to commercial devices. TapSkin [20] utilized IMUs and the microphone of a smartwatch to recognize tapping gestures on the back of the hand. However, its gesture family only contained simple tap gestures, and not included elaborate sliding sequences. EarBuddy [4] was the most similar to *Ui-Ear*, which utilized the face area near the ears as an input surface. However, based on acoustic signals, it was susceptible to environmental interference. And as we know, the collection and transmission of acoustic signals usually bring more energy consumption than IMUs.

Technically, our work is most related to *domain adversarial learning*. In the early years, in the field of computer vision [21]–[24] utilized *domain adversarial learning* to solve the unsupervised domain adaptation problem. After that, Sriram et al. [25] proposed an end-to-end speech recognition framework based on a generative adversarial network (GAN).

And CrossGR [26] used a target-adaptive scheme for Wi-Fi based gesture recognition. In summary, the core ideas of these studies were similar, they all mapped the feature of the target domain and the source domain to the same embedding space. Recently, some state-of-the-art works had improved the domain adaptation performance. Zhao et al. [27] combined convolution and recurrent neural networks to extract sleep information from radio frequency signals, and discarded information specific to measurement conditions or subjects. Similarly, EI [28] based on the four wireless signals of mmwave, wifi, ultrasonic and visible light to recognize human daily activities, which can remove the environment and subject specific information contained in the activity. However, all of the above work relied on unseen users' or environmental data during the model training process. Consequently, they cannot be used in real-time inference tasks. Unlike them, EUIGR [29] token the ongoing gesture into consideration, which integrated a complicated convolution and recurrent neural networks to fuse RFID low-level physical characters and extract space-temporal information. Ultimately, it can obtain strong robustness to subjects diversity and reduced environmental dependence. However, different from the above human activity recognition based on wireless signals, our work *Ui-Ear* exploits IMUs of wireless earbuds to perceive skin gestures. And the Domain Adversarial Learning framework we proposed is more lightweight, which can run on the resource-constrained device in real-time.

III. SYSTEM DESIGN

In this section, we will provide an overview of the proposed on-face finger gesture recognition system, *Ui-Ear*, for wireless earbuds based on vibration sensing. The overall system design of *Ui-Ear* is presented in Fig. 2 which consists of three major components. First, in training data collection, we recruit a certain number of volunteers to perform seven different on-face gestures to build a realworld dataset for model training and evaluation. Second, in DAL-based classifier training phase, the on-face gesture classifier is trained based on Domain Adversarial Learning (DAL) framework. At last, in personalization and online inference phase, the learned model is fine-tuned with few training samples from the target user and the personalized classifier is used for online gesture inference.

A. Data Collection

Before the DAL-based classifier training, a realworld dataset is collected. The participants are asked to perform seven different on-face gestures including tapping, slide-up, side-down, slide-left, slide-right, clockwise circle and counter-clockwise-circle as shown in Figure 3. During the data collection, we observe, different participants may perform the same gesture in relatively different ways, which brings significant variance on the data samples we collect. For example, the users may slide on their face in different speed; some users may touch on the face with their nails but some with their pad of the finger. These all introduce significant challenge on learning an on-face gesture classifier with good generalization capability.

We show some examples of the sensor reading when the users performing different on-face gesture in Fig. 4. The first row is sensor-reading of the x-axis of accelerometer when the same subject performing seven different gestures. The waveform of the seven different gestures are clearly distinctive so that they are distinguishable. In the second row, the sensor reading of clockwise-circle (CW) from seven different users is presented. Though they are performing the same gesture, the waveform is variant. Therefore, to obtain a robust on-face gesture recognition system, we have to deal with the inter-users variance carefully.

B. DAL-based Classifier Training

The DAL-based classifier training framework is designed based on Domain Adversarial Learning. The framework consists of three major components which are feature extractor, domain discriminator and gesture classifier. The framework is trained in two steps: 1) training the domain discriminator and 2) training the gesture classifier.

1) Domain Discriminator Training: The domain discriminator is used to distinguish the gestures performed by different users. An overview of the structure and training process of the domain discriminator is shown in Fig. 5. $\{M, y_D\}$ is a training sample with data matrix M and its corresponding domain label y_D. $M = \{M_1, M_2...M_6\}$ consists of six independent time-series from different axes of accelerometer and gyroscope. In order to remove the influence of gravity on acceleration data and speed up the convergence of training, the data matrix M is normalized in each axis before being feed to the input of the deep neural network. As shown in Fig. 5, this step involves training the parameters of feature extractor ($\mathcal{C}$) and domain discriminator ($\mathcal{D}$). The feature extractor is composed of three layers of 1-dimensional convolution. The kernel size of the convolution is 3 and the stride is 1. The number of output channels are 32, 64 and 128 respectively for each layer. Batch normalization operation is added after each convolutional layer to accelerate the convergence of training and Relu is used as the activation function. MaxPooling layers are also applied after each of the convolutional layer to prevent the network from over-fitting and reduce the complexity of the network. At last, a flatten layer finalized the feature extractor and outputs a 2048-dimensional feature vector F. In mathematics, feature extractor can be expressed as,

$$F = \mathcal{C}(M; \theta_{\mathcal{C}}) \tag{1}$$

Where $\mathcal{C}$ is the mapping function and $\theta_{\mathcal{C}}$ is the set of parameters of the feature extractor.

Then the extracted feature vector F is forwarded to the domain discriminator($\mathcal{D}$) which is composed of two fully connected layers. The number of nodes are 512 and K (i.e, the number of users in trainingset) respectively. The output of the final layer is an K-dimensional probabilistic vector and the domain discriminator can be expressed as,

$$D_p = \mathcal{D}(F; \theta_{\mathcal{D}}) \tag{2}$$

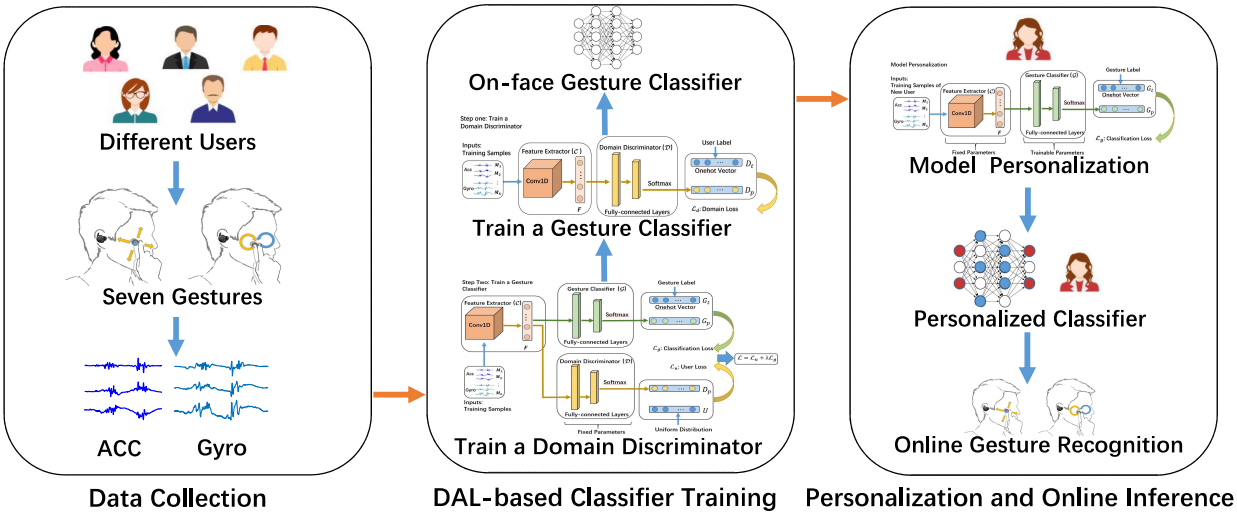

Fig. 2. The system overview of major components of *Ui-Ear*

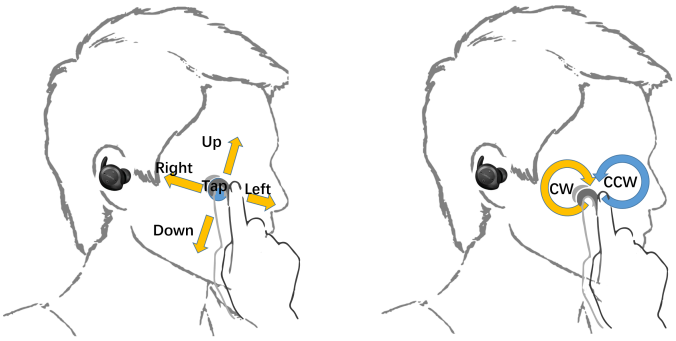

Fig. 3. The seven on-face finger gestures defined in this paper: tap, slide-up. slide down, slide-left, slide-right, clockwise circle (CW) and counter-clockwise circle (CCW)

where D_p is predicted vector with real-valued probabilities as its elements and $\theta_{\mathcal{D}}$ is the set of network parameters of the domain discriminator. Then, then the user label is converted to a onehot encoding vector: D_t. Cross entropy is applied to estimate the loss between the prediction and the user label,

$$\mathcal{L}_d = -\frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K} D_t^{(k)} log D_p^{(k)} \tag{3}$$

where N is number of training samples of a batch (i.e., batchsize). The objective of training on domain discriminator is to minimize the domain loss $\mathcal{L}_d$. Adam optimizer is used for optimization. After the training, the CNN-based feature extractor and domain discriminator can work together to distinguish different users according to users on-face gestures.

2) Gesture Classifier Training: In the second step, we will train an on-face gesture classifier based on Domain Adversarial Learning (DAL) framework. The domain discriminator trained from the first step is used to suppress the user-dependent information. The workflow of DAL-based gesture classifier training is shown in Fig. 6. When training the classifier, the parameters of domain discriminator are fixed and the optimization only updates the parameters of feature extractor and gesture classifier. In this step, we aim to train a feature extractor independent of users (domains). To achieve this, the objective of the optimization in training is to maximize the accuracy of gesture classification while fooling the domain discriminator so that the users cannot be identified. If the discriminator cannot distinguish different users at all, the expected output should be a uniform distribution $U = \{\frac{1}{M}, \frac{1}{M}..., \frac{1}{M}\}$. We define the user loss $\mathcal{L}_u$ as the Mean Squared Error (MSE) between the prediction and the expected output,

$$\mathcal{L}_u = \frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K}(D_p^{(k)} - U^{(k)})^2 \tag{4}$$

As shown in Fig. 6, the output of feature extractor is also feed to a gesture classifier ($\mathcal{G}$) which consists of two fully-connected layers, the number of nodes are 512 and I (I=7, i.e., seven different on-face gestures) respectively. Softmax is applied for the gesture classification. The predicted probability vector of the gesture classifier is:

$$G_p = \mathcal{G}(F; \theta_{\mathcal{G}}) \tag{5}$$

where $\theta_{\mathcal{G}}$ is the set of parameters of the gesture classifier. Cross entropy is used calculate the classification loss between the predicted probability vector and onehot vector converted from the gesture label, which is expressed as,

$$\mathcal{L}_g = -\frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{I} G_t^{(i)} log G_p^{(i)} \tag{6}$$

To minimize the user loss and classification loss simultaneously, a combined loss for the DAL-based framework is defined as,

$$\mathcal{L} = \mathcal{L}_u + \lambda \mathcal{L}_g \tag{7}$$

After training with on-face gesture samples from different users, a gesture classifier with user-independent feature extractor is obtained.

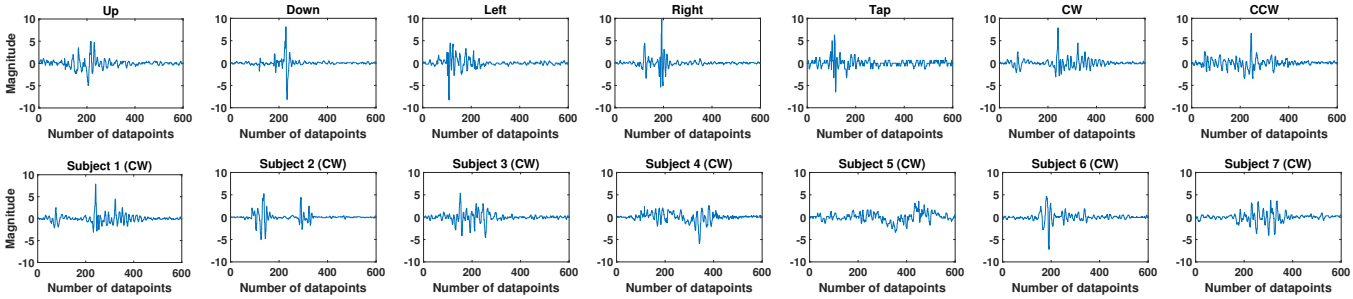

Fig. 4. Comparison of seven different gesture data of the specific user (upper row), and comparison of the same gesture (CW) data of seven different users (lower row).

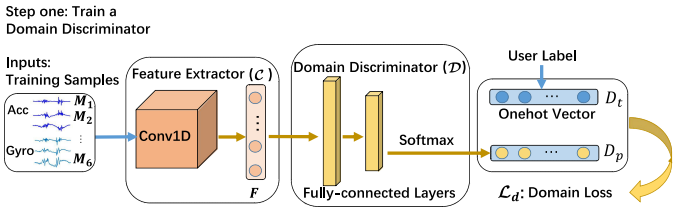

Fig. 5. Step one: train a domain discriminator: a CNN-based discriminator is trained to identify the users according to the on-face gesture.

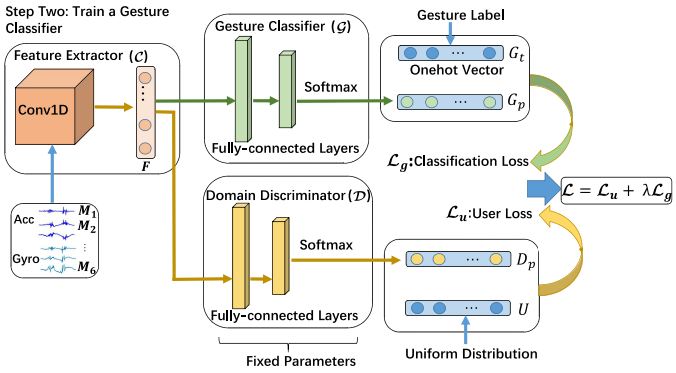

Fig. 6. Step two: train a gesture classifier: a CNN-based gesture classifier is trained with DAL framework to extract user-independent features.

C. Model Personalization and Online Gesture Inference

After the DAL-based classifier training, the gesture classifier ($\mathcal{G}$) with feature extractor($\mathcal{C}$) can be implemented on the personal device of the users for on-face finger gesture recognition. Based on the concept of transfer learning, personalizing deep neural network with few training samples from the new user can significantly improve the recognition accuracy. The CNN-based classifier normally captures low-level features with the first few convolutional layers, i.e., the feature extractor, which is significantly reusable. While the higher level (fully-connected) learns the representation of different classes with the low-level features. Therefore, as shown in Fig. 7, during personalization, the CNN-based model learned from step two is transferred to the target user then the parameters of upper layers are fine-tuned. As the coefficients of the feature extractor is reusable, their values are fixed during personalization to reduce the requirement on the amount of training data and the training efforts from users can be significantly saved.

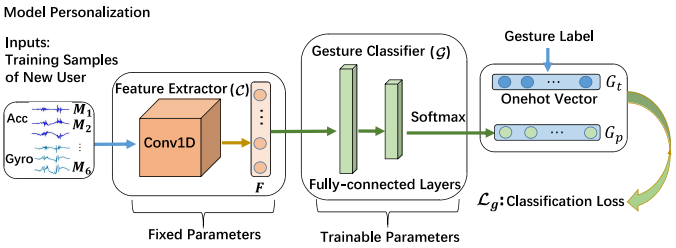

Fig. 7. Model personalization with transfer learning: the values of convolutional layers are fixed and the fully-connected layers are fine-tuned with few samples from the new user.

Cross-entropy is adopted to estimate the loss between the predicted probability vector and the onehot vector converted from gesture label. The objective of personalization is to minimize the loss $\mathcal{L}_g$ in Equation (6) with fixed feature extractor. The personalized model combines a gesture classifier fine-tuned with few training samples from current user and feature extractor trained with large number of training samples from other users. The model personalisation can significantly improve the accuracy of the gesture classification of current user.

IV. Dataset Evaluation

A. Experimental Setup & Competing Methods

We recruit 20 subjects in different heights and weights to participate in two realworld data collection sessions. HI226DK [30] IMU board is used for picking up the vibration induced by on-face gestures. During the data collection, it is attached on the auricle of the ear where the wireless earbuds are normally worn. The sampling rate of the IMU is set to be 200Hz. In each session, the participants sit in chair and perform seven different on-face gestures including tapping, slide-up, slide-down, slide-left, slide-right, clockwise-circle and counter-clockwise-circle. The reading of three-axes accelerometer and three-axes gyroscope is recorded along with the gesture labels and identity labels. One participant contributes 100 samples for each gesture and therefore, 28,000

samples with labels in total are collected from the two sessions. Then, we will evaluate the accuracy of gesture recognition with *Ui-Ear* on the realworld dataset. The evaluation can be vastly categorized into three major tasks which are user-dependent classification, user-independent classification and personalization. The dataset evaluation is conducted on a high performance desktop running Ubuntu 18.04.1. The machine contains two 12-core 2.3GHz Intel Xeon Gold CPU and 512Gb DDR4 RAM. The deep neural networks are coded and trained with Pytorch [31] on two NVIDIA GeForce RTX 3090 GPUs with 24 GB of G6X memory.

In this section, we implement and compare the classification accuracy of five different machine learning models for on-face gesture recognition in the three evaluation tasks. The five machine learning models are:

- **DAL-CNN** is the convolutional neural network trained with Domain Adversarial Learning Framework which is machine learning model used in *Ui-Ear*.
- **CNN** is the traditional convolutional neural network without adversarial learning. CNN is also an end-to-end approach in which the data matrix obtained from IMU is directly used as the input of the network for model learning and inference.
- **XGBoost [32], Randomforest, SVM**: we also implement three traditional machine learning models. Different from the CNN-based approaches, these methods are not end-to-end. The training and inference of these models are based on the 100-dimensional feature vectors extracted from IMU time-series. The detailed description and implementation of the feature extraction can be found in AccelWord [33].

B. User-dependent Classification

We first evaluate and compare the recognition accuracy of different machine learning models on the user-dependent classification task. The "user-dependent" refers to that the users' identities in the training and inference stages are the same, i.e., all the 20 subjects are included. During the evaluation, the collected dataset from the first session is used as training samples and the learned machine learning models are applied on the dataset collected from the second session for gesture recognition. We gradually change the number of samples per gesture of each subject from 1 to 100 and calculate the classification accuracy of different gesture recognition methods. To reduce the randomness of the results, the average accuracy from 30 independent trials is reported in 8. From the results we can observe, with the growth of number of training samples, the classification accuracy of gesture recognition methods increases sharply at the beginning, and then the improvement becomes insignificant after the number of training samples reach 30 for each gesture of each subject. Moreover, by comparing the accuracy across the competing methods, the end-to-end approaches, **DAL-CNN** and **CNN** achieves significantly higher accuracy than the traditional feature-based methods and **DAL-CNN** is at least 5% higher than traditional **CNN** when number of training samples are 30. At last, we also notice that, the accuracy **SVM** is far from other methods, therefore, we omit the results of **SVM** in the following context.

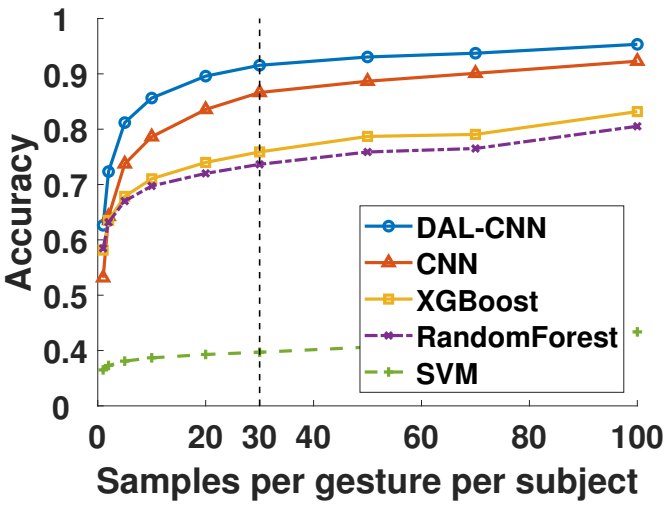

Fig. 8. The recognition accuracy against different number of training samples with different classification methods (all 20 subjects are included in the training dataset)

We then show the specific accuracy of each subject using the four different machine learning models in Fig. 9. The number of training samples are 100. By carefully comparing the recognition accuracy of different methods for each subject, we can observe, our proposed **DAL-CNN** achieves the highest accuracy on gesture recognition for most of the subjects and its accuracy is close to the best one for the rest two subjects (No.11 and No.17). Therefore, we can claim, for the user-dependent evaluation task, **DAL-CNN** is both the most accurate and reliable method.

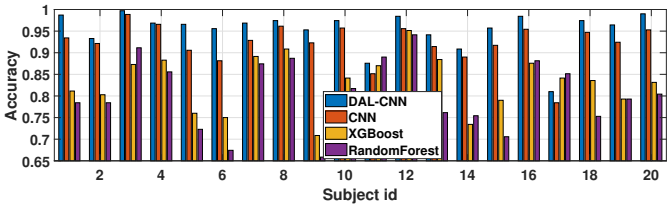

Fig. 9. The recognition accuracy of different subjects

C. User-independent Classification

In the second task, we will evaluate the accuracy of different machine learning models for user-independent classification. In the user-independent task, the generalization capability of the machine learning models are evaluated. Specifically, for each round of evaluation K_1 subjects are randomly selected and all the samples of these subjects are used to train the machine learning models. Then the models are applied on all the samples of the rest subjects to show the accuracy of the user-independent classification. During the evaluation, K_1 = {2, 5, 10, 15, 19} subjects are randomly selected to train the machine learning model and the classification accuracy on

the rest of the subjects is calculated. To reduce the impact of randomness, we repeat the random selection for 20 times and the average of the classification accuracy is presented in Fig. 10. We omit the results of **SVM** and **Randomforest** as their accuracy are far less than their competing methods. From the results, we can observe, by including more subjects in trainingset, the classification accuracy of all three methods is improved significantly as more variance of the on-face gestures is learned. Moreover, comparing the results between different methods, **DAL-CNN** always achieves the highest recognition accuracy and the improvement over the second best method is over 3% when 19 subjects are used as training.

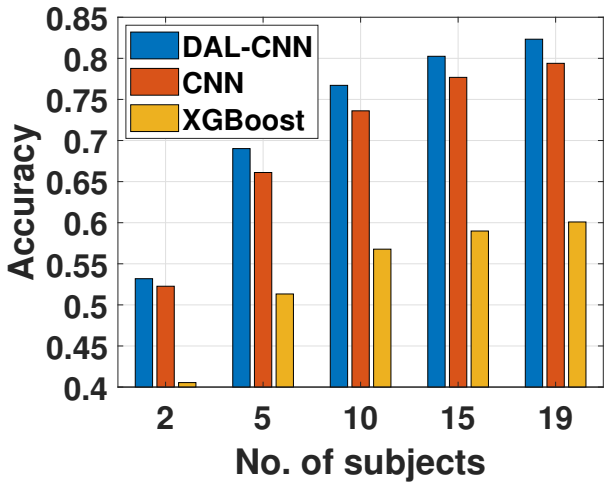

Fig. 10. The user-independent classification accuracy with respect to different number of subjects used in training

We also show the classification accuracy of each subject in the leave-one-out cross-validation in Fig. 11 to demonstrate the stability of the gesture recognition methods. By comparing the accuracy of different methods for each subject, we can observe, **DAL-CNN** achieves the highest or close to highest classification accuracy for each subject and it is up to 10% higher than the second best method (subject No. 16). Again, for the user-independent evaluation task, **DAL-CNN** is both the most accurate and reliable method.

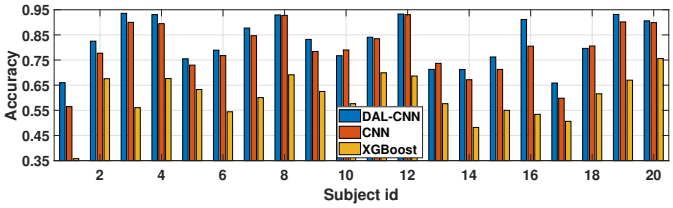

Fig. 11. The user-independent classification accuracy of each subject(leave-one-out cross-validation)

D. Personalization

To further improve the classification accuracy, personalizing the machine learning model with few training samples from the new users normally works well. In this section, we compare the performance of different personalization approaches. Leave-one-out cross-validation is applied for evaluation. Specifically, the model trained from 19 subjects are used as pre-trained model and the rest one subject provides new training samples for personalization. Four personalization approaches and one baseline method are implemented:

- **DAL-CNN** and **CNN**: the CNN-based personalization fixes the parameters of the feature extractor obtained from pre-trained model and fine-tuning the parameters of the fully-connected layers.
- **DAL-CNN-SVM and CNN-SVM** apply the CNN-based feature extractor from the pre-trained **DAL-CNN** and **CNN** models to replace traditional feature extraction method. The SVM models are then personalized by the features extracted from the new samples.
- **XGBoost** is the best traditional method trained with only the new samples from the target user. It is a baseline method without transferred knowledge.

During personalization, we gradually increase the number of training samples per gesture from the target users from 1 to 100 and calculate the gesture recognition accuracy. The training samples are randomly selected and the average of the recognition accuracy over 30 independent trials is presented in Fig. 12.

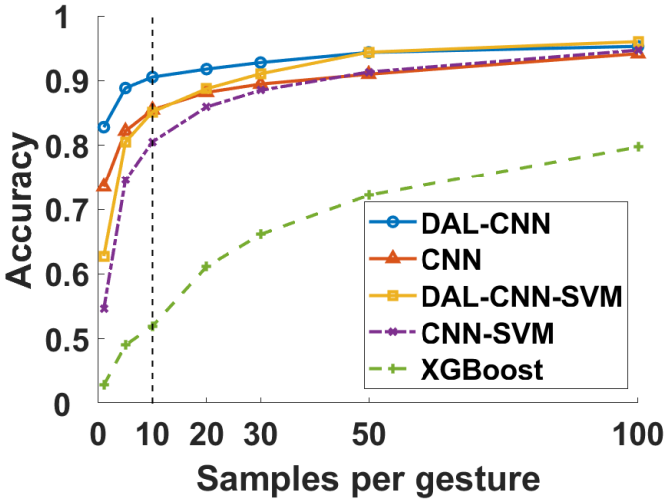

Fig. 12. The recognition accuracy against different number of samples per gesture for personalization.

Overall, with the growth of number of training samples, the accuracy of all methods increases immediately, then when the number of training samples per gesture is over 20, the recognition accuracy tends to be stable. This indicates the personalization does not require massive training samples to produce reasonable accuracy. Moreover, **DAL-CNN** achieves the highest classification accuracy and the gap is significant when the number of training samples is low. For example, when only one sample per gesture is used for personalization, the accuracy of **DAL-CNN** has reached over 82% and it

is about 10% higher than the second best approach. Then, when the number of training samples per gesture are 10, the recognition accuracy of **DAL-CNN** achieves over 91% while that of second best method is about 85%. By comparing the accuracy of the methods with or without DAL (**DAL-CNN** v.s. **CNN** and **DAL-CNN-SVM** v.s. **DAL-CNN**), we can observe, the DAL framework can significantly improve the recognition accuracy. At last, the four methods exploiting the transferred knowledge from pre-trained CNN model demonstrate superior performance on classification accuracy over the baseline. For example, the accuracy of **DAL-CNN** is almost 40% higher than **XGBoost**.

Then we show the per-subject accuracy after personalization in Fig. 13. The number of training samples per gesture for personalization is 10. The accuracy reported are obtained from averaging the results from 30 independent trials. **XGBoost** is not included in the figure as its accuracy is too far from its competing methods. First of all, the average accuracy of the four methods is 90.55% (**DAL-CNN**), 85.49% (**CNN**), 85.15% (**DAL-CNN-SVM**) and 80.53% (**CNN-SVM**) respectively. The DAL framework brings around 5% improvement on recognition accuracy for both CNN-based and SVM-based classifiers when 10 samples per gesture is used for personalization. Then by comparing the accuracy for each subject, we find, **DAL-CNN** shows good reliability in gesture recognition; it achieves the highest accuracy for all 20 subjects.

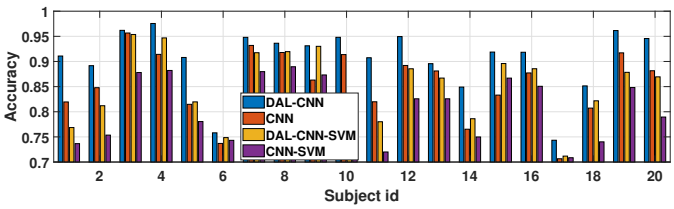

Fig. 13. The recognition accuracy of personalization for different subjects (10 samples per gesture of each subject are provided for fine-tuning)

E. User Identification

When training the domain discriminator, we find the IMU signals generated by on-face gestures from different users are distinguishable (see Fig. 4 in Section III-A). Basing on the inter-subject variance of the gestures, user authentication layer can be added as the security protection for the wireless earbuds. Therefore, at last, we investigate the accuracy of user identification with on-face gestures. The gesture samples collected from one session is used for training and those from the other session are for inference. The network structure is the same as the domain discriminator with feature extractor in Section III-B. In Table I, the top-1 accuracy of user identification with different gestures are presented. From the results, we can observe, the accuracy of user identification with some of the gestures (Tap, Slide-right, CW and CCW) is close or even over 90%.

TABLE I
THE TOP-1 ACCURACY OF USER AUTHENTICATION WITH DIFFERENT GESTURES

Gesture	Tap	Up	Down	Left	Right	CW	CCW	Average
Accuracy(%)	89.34	81.98	81.75	74.5	90.5	89.29	89.51	85.27

V. SYSTEM IMPLEMENTATION & RESOURCE CONSUMPTION

In this section, we implement the personalization and inference phases of *Ui-Ear* on a off-the-shelf smartphone to evaluate the system overhead introduced by the gesture recognition.

A. Positive Events Detection

As the IMU sensor picks up all types of physical activities of users, human movement like walking, head motion and pose transition may trigger the on-face recognition falsely and cause inaccurate gesture recognition results. Therefore, we need an on-face gesture detection layer to filter out the irrelevant human motions. Before an on-face gesture is found, a 200ms sliding window with 50ms step size is applied on the IMU time-series. For each 200ms time-series, a positive event detector is used to determine if this belongs to the on-face gestures. The detector consists of two fully-connected layers with softmax as the classification function and the number of nodes for each fully-connected layers are 64 and 2. We collect a number of different daily activities as the negative events to train the detector along with the on-face gestures. The activities include sitting, standing, lying down, sit-to-stand and vice versa, head moving and walking. According to our evaluation, the lightweight detector achieves an overall accuracy of 97.21%. Then the false positive rate of the detection is 2.3% and false negative rate of the detection is 3.25%.

B. System Overhead

We implement *Ui-Ear* on an Android smartphone and evaluate the system overhead of key components running on user's personal devices including the positive event detection, gesture reference and personalization. Considering the different application scenario, the personalization module either runs in-situ on a smartphone or is offloaded to a PC in local area network according to availability of local PC. Remote cloud is not a choice in our paper for the privacy concern. The smartphone we use for implementation is Xiaomi Mi 11 Ultra with Snapdragon 888 platform including one 2.84GHz Cortex X1, three 2.4GHz Cortex A78 and four 1.8GHz Cortex A55 processors. The running memory of the model we use is 12GB and its battery capacity is 73.8KJ (5000mah). The local PC is running Window 10 operating system with Intel Core i7-10700F processor and a Nvidia RTX-2060 GPU. The WiFi chip of the PC is AX200 160MHz.

1) Model Personalization: We implement model personalization on both smartphone and local PC. Deeplearning4j [34] and Pytorch [35] frameworks are adopted for fine-tuning on

smartphone and local PC respectively. For implementation on smartphone, we estimate the running time and energy consumption of personalization with different number of training samples per gesture. The energy consumption is calculated as multiplication of *Current* and *Voltage* and running time T, i.e., $Energy = Current \times Voltage \times T$. The current and voltage can be obtained by Android APIs. The resource consumption of *Ui-Ear* is presented in Table II. From the results we can observe, with the growth of number of training samples for fine-tuning, the resource consumption of personalization implemented on smartphones increases rapidly. Though the time and energy consumption is non-trivial, the personalization is an once-off operation and will not be repeated frequently. For example, when personalizating the model with 10 samples per gesture, the running time is 13.87s and the energy consumption is 29.9J. The energy consumption only accounts for 0.04% of the battery capacity.

TABLE II
THE RESOURCE CONSUMPTION OF PERSONALIZATION WITH DIFFERENT NUMBERS OF SAMPLES PER GESTURE (IMPLEMENTATION ON SMARTPHONE)

Samples Per Gesture	1	2	5	10	20	30	50	100
Running Time (s)	6.02	6.93	9.39	13.87	22.2	31.70	49.80	94.90
Energy Consumption (J)	11.4	15.1	20.0	29.9	60.1	84.5	228.7	478.6

For implementation on local PC, the energy budget of the PC can be regarded as unlimited as it is powered by external electric supply. However, offloading and downloading the deep neural networks and training samples introduces time-delay and transmission cost on the smartphone. Therefore, we estimate the resource consumption of the smartphone in the offloading approach and show the results in Table III. By comparing the resources consumption in Table II and Table III, we can observe, the offload approach is able to save the resource consumption of the smartphone significantly. For example, when the number of training samples per gesture is 10, the smartphone-only approach consumes 15.0 and 7.3 times more energy and running time than offloading the personalization to a local PC.

TABLE III
THE RESOURCE CONSUMPTION OF SMARTPHONE WHEN OFFLOADING PERSONALIZATION TO LOCAL PC

Samples Per Gesture	1	2	5	10	20	30	50	100
Time Delay (s)	1.58	1.61	1.64	1.90	2.09	2.17	2.95	4.47
Transmission Cost (J)	0.97	1.1	1.5	2.0	2.7	3.3	4.5	7.7

2) Gesture Recognition on Smartphone: At last, we implement the whole system of *Ui-Ear* on smartphone and evaluate the resource consumption of its key components including the positive event detection, on-face gesture inference and personalization on smartphone/ PC. Different from model personalization, the positive event detection and on-face gesture inference can be executed frequently whenever the gesture recognition is triggered. Therefore, they are the key impact factor on the system efficiency and user experience when *Ui-Ear* is used in practical scenarios. From the results shown in Table IV, we can observe, each online gesture recognition, including event detection and gesture inference, only takes less than 1.26ms to execute and consumes less than 2.37mJ energy. Considering the targeted daily usage of the smartphone is 10-12 hours, the energy consumption of each online gesture recognition with *Ui-Ear* only accounts for 0.0019% to 0.0023% of the energy budget per minute. Therefore, the online components of *Ui-Ear* only introduces trivial system overhead and can run in-situ on smartphones with unnoticeable time delay.

TABLE IV
SYSTEM OVERHEAD OF *Ui-Ear*

Components	Positive Event Detection	On-face Gesture Inference	Personalization on smartphone/offload
Computation Time	0.067ms	1.1871ms	13.87s/398ms
Energy Consumption	0.079mJ	2.29mJ	29.9J/2.0J

VI. CONCLUSION

In this article, we propose, *Ui-Ear*, an on-face gesture recognition system to enrich the maneuverability of the interaction between human and wireless earbuds. The *Ui-Ear* facilities vibration sensing capability enabled by the IMU sensors embedded in both budget and high-end wireless earbuds to distinguish the on-face gestures. To improve the robustness and accuracy of gesture recognition, we propose a DAL-based framework in training the feature extractor of the gesture classifier. The extensive evaluation on the realworld dataset shows that the feature extractor trained from DAL-based framework can significantly improve the accuracy of the on-face gesture classifier in the user-dependent, user-independent and personalization tasks. At last, *Ui-Ear* is implemented on an off-the-shelf smartphone and its resource-consumption is evaluated. The processing delay and energy consumption shows that *Ui-Ear* can run in-situ on smartphones with trivial system overhead.

REFERENCES

[1] "Kirin a1," https://www.huaweicentral.com/kirin-a1-the-worlds-first-bluetooth-5-1-and-bluetooth-low-energy-5-1-wearable-chip/, 2020.

[2] "Apple h1," https://9to5mac.com/2019/03/20/new-apple-airpods-now-available-h1-chip-wireless-charging-case-hands-free-hey-siri/, 2019.

[3] "Airpods pro," https://www.apple.com/uk/airpods-pro/, 2019.

[4] X. Xu, H. Shi, X. Yi, W. Liu, Y. Yan, Y. Shi, A. Mariakakis, J. Mankoff, and A. K. Dey, *EarBuddy: Enabling On-Face Interaction via Wireless Earbuds*. New York, NY, USA: Association for Computing Machinery, 2020, p. 1–14.

[5] D. Ma, A. Ferlini, and C. Mascolo, "Oesense: Employing occlusion effect for in-ear human sensing," in *Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 175–187.

[6] R. Lissermann, J. Huber, A. Hadjakos, S. Nanayakkara, and M. Mühlhäuser, "Earput: Augmenting ear-worn devices for ear-based interaction," in *Proceedings of the 26th Australian Computer-Human Interaction Conference on Designing Futures: The Future of Design*, ser. OzCHI '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 300–307.

[7] E. Tamaki, T. Miyak, and J. Rekimoto, "Brainyhand: A wearable computing device without hmd and it's interaction techniques," in *Proceedings of the International Conference on Advanced Visual Interfaces*, ser. AVI '10. New York, NY, USA: Association for Computing Machinery, 2010, p. 387–388.

[8] "Rog phone," https://rog.asus.com/us/phones-group/, 2021.

[9] K. Sun, T. Zhao, W. Wang, and L. Xie, "Vskin: Sensing touch gestures on surfaces of mobile devices using acoustic signals," in *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 591–605.

[10] S. S. A. Shimon, S. Morrison-Smith, N. John, G. Fahimi, and J. Ruiz, "Exploring user-defined back-of-device gestures for mobile devices," in *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services*, ser. MobileHCI '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 227–232.

[11] Z. Ling, J. Luo, Y. Liu, M. Yang, K. Wu, and X. Fu, "Sectap: Secure back of device input system for mobile devices," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, 2018, pp. 1520–1528.

[12] N. Shalev, I. Keidar, Y. Moatti, and Y. Weinsberg, "Watchit: Who watches your it guy?" in *Proceedings of the 8th ACM CCS International Workshop on Managing Insider Security Threats*, ser. MIST '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 93–96.

[13] G. Presti, D. Ahmetovic, M. Ducci, C. Bernareggi, L. Ludovico, A. Baratè, F. Avanzini, and S. Mascetti, "Watchout: Obstacle sonification for people with visual impairment or blindness," in *The 21st International ACM SIGACCESS Conference on Computers and Accessibility*, ser. ASSETS '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 402–413.

[14] J. Wang, K. Zhao, X. Zhang, and C. Peng, "Ubiquitous keyboard for small mobile devices: Harnessing multipath fading for fine-grained keystroke localization," in *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 14–27.

[15] J. Liu, Y. Wang, G. Kar, Y. Chen, J. Yang, and M. Gruteser, "Snooping keystrokes with mm-level audio ranging on a single phone," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 142–154.

[16] J. Liu, Y. Chen, M. Gruteser, and Y. Wang, "Vibsense: Sensing touches on ubiquitous surfaces through vibration," in *2017 14th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, 2017, pp. 1–9.

[17] S. Singh and A. Kumar, "Review of skinput technology: Input through skin," in *2018 International Conference on Sustainable Energy, Electronics, and Computing Systems (SEEMS)*, 2018, pp. 1–5.

[18] C. Harrison, D. Tan, and D. Morris, "Skinput: Appropriating the skin as an interactive canvas," *Commun. ACM*, vol. 54, no. 8, p. 111–118, Aug. 2011.

[19] Y. Zhang, J. Zhou, G. Laput, and C. Harrison, *SkinTrack: Using the Body as an Electrical Waveguide for Continuous Finger Tracking on the Skin*. New York, NY, USA: Association for Computing Machinery, 2016, p. 1491–1503.

[20] C. Zhang, A. Bedri, G. Reyes, B. Bercik, O. T. Inan, T. E. Starner, and G. D. Abowd, "Tapskin: Recognizing on-skin input for smartwatches," in *Proceedings of the 2016 ACM International Conference on Interactive Surfaces and Spaces*, ser. ISS '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 13–22.

[21] M. L. Wouter M. Kouw, "An introduction to domain adaptation and transfer learning." https://arxiv.org/abs/1812.11806.

[22] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML'15. JMLR.org, 2015, p. 1180–1189.

[23] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, no. 1, p. 2096–2030, Jan. 2016.

[24] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2962–2971.

[25] A. Sriram, H. Jun, Y. Gaur, and S. Satheesh, "Robust speech recognition using generative adversarial networks," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 5639–5643.

[26] X. Li, L. Chang, F. Song, J. Wang, X. Chen, Z. Tang, and Z. Wang, "Crossgr: Accurate and low-cost cross-target gesture recognition using wi-fi," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 5, no. 1, Mar. 2021.

[27] M. Zhao, S. Yue, D. Katabi, T. S. Jaakkola, and M. T. Bianchi, "Learning sleep stages from radio signals: A conditional adversarial architecture," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ser. ICML'17. JMLR.org, 2017, p. 4100–4109.

[28] W. Jiang, C. Miao, F. Ma, S. Yao, Y. Wang, Y. Yuan, H. Xue, C. Song, X. Ma, D. Koutsonikolas, W. Xu, and L. Su, "Towards environment independent device free human activity recognition," in *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 289–304.

[29] Y. Yu, D. Wang, R. Zhao, and Q. Zhang, "Rfid based real-time recognition of ongoing gesture with adversarial learning," in *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*, ser. SenSys '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 298–310.

[30] "Hi226dk," https://9to5mac.com/2019/03/20/new-apple-airpods-now-available-h1-chip-wireless-charging-case-hands-free-hey-siri/, 2019.

[31] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, and T. K. et al., "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, 2019, pp. 8024–8035.

[32] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 785–794.

[33] L. Zhang, P. H. Pathak, M. Wu, Y. Zhao, and P. Mohapatra, "Accelword: Energy efficient hotword detection through accelerometer," in *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 301–315.

[34] "Deeplearning4j," https://deeplearning4j.konduit.ai/android/setup, 2020.

[35] "Pytorch mobile," https://pytorch.org/mobile/android/, 2019.