AdaptiveSound: An Interactive Feedback-Loop System to Improve Sound Recognition for Deaf and Hard of Hearing Users

Hang Do

University of Washington, doh25@cs.washington.edu

Quan Dang

University of Washington, quangary@cs.washington.edu

Jeremy Zhengqi Huang

University of Michigan zjhuang@umich.edu

Dhruv Jain

University of Michigan, profdj@umich.edu

Figure 1: AdaptiveSound allows end-users to provide positive and negative feedback to a sound recognition model's output, adapting the model to a variety of sounds in diverse contexts and environments.

Sound recognition tools have wide-ranging impacts for deaf and hard of hearing (DHH) people from being informed of safety-critical information (*e.g.,* fire alarms, sirens) to more mundane but still useful information (*e.g.,* door knock, microwave beeps). However, prior sound recognition systems use models that are pre-trained on generic sound datasets and do not adapt well to diverse variations of real-world sounds. We introduce AdaptiveSound, a real-time system for portable devices (*e.g.,* smartphones) that allows DHH users to provide corrective feedback to the sound recognition model to adapt the model to diverse acoustic environments. AdaptiveSound is Informed by prior surveys of sound recognition systems, where DHH users strongly desired the ability to provide feedback to a pre-trained sound recognition model to fine-tune it to their environments. Through quantitative experiments and field evaluations with 12 DHH users, we show that AdaptiveSound can achieve a significantly higher accuracy (+14.6%) than prior state-of-the art systems in diverse real-world locations (*e.g.,* homes, parks, streets, and malls) with little end-user effort (about 10 minutes of feedback).

CCS CONCEPTS • Human-centered computing ~ Accessibility ~ Accessibility technologies

Additional Keywords and Phrases: Accessibility, deaf, Deaf, hard of hearing, sound awareness, audio event detection, applied machine learning, reinforcement learning, incremental learning, human-in-the-loop, Human-AI.

ACM Reference Format:
First Author's Name, Initials, and Last Name, Second Author's Name, Initials, and Last Name, and Third Author's Name, Initials, and Last Name. 2018. The Title of the Paper: ACM Conference Proceedings Manuscript Submission Template: This is the subtitle of the paper, this document both explains and embodies the submission format for authors using Word. In Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY. ACM, New York, NY, USA, 10 pages. NOTE: This block will be automatically generated when manuscripts are processed after acceptance.

1 INTRODUCTION

Sound recognition has the potential to help deaf and hard of hearing (DHH) better understand their environment (*e.g.,* through a door knock), perform safety-related tasks (*e.g.,* through a fire-alarm), or feel present (*e.g.,* through a bird-chirp). Past work has built sound recognition systems and apps, a feature also available in commercial iOS [16] and Android [28] smartphones. However, since real-world sounds vary a lot in different environments, it is difficult to capture the full variation in a given dataset, and these solutions–that are trained on generic datasets–are only about 60-80% accurate when deployed in the real-world [18, 19]. This causes DHH users to miss many important sounds of interest. Indeed, in recent studies [18–20], including a large-scale survey with 472 DHH participants [20], DHH people expressed their dissatisfaction with the state-of-the-art sound recognition solutions and desired the ability to provide corrective feedback to the model output. For example, one user commented:

> *"Before remodeling the kitchen, we had a porcelain sink. And the system used to detect [the sound] correctly… We have [a] stainless steel sink now. […] The sound of water hitting it is very different [and is misrecognized by the system] So, I wanted to tell [the system] … this is not a fan sound, but a water running sound, and it should then be able to recognize it."*

To accommodate this need, we introduce *AdaptiveSound*, the first "feedback loop" sound recognition system, that allows users to provide positive and negative feedback to the model's output, greatly increasing the model's accuracy and adaptability to users' contexts and environments. AdaptiveSound uses an incremental reinforcement learning technique that gradually adapts a pre-trained model to new environments with little end-user effort.

When evaluated on sounds from multiple real-world locations (*e.g.,* homes, urban streets, parks, malls), AdaptiveSound archived a significantly higher accuracy (+14.6%, $p<.001$) than the state-of-the-art pre-trained sound recognition approach, after about 10 minutes of end-user feedback[1]. User study of our deployed Android app with 12 DHH people revealed that the system is easy-to-use, can train in real-time, and greatly enhances the users' awareness of sounds in the field.

In summary, our work contributes: (1) the first feedback-loop sound recognition system for deaf and hard of hearing (DHH) users, (2) findings from performance experiments and field evaluations of our incremental reinforcement learning technique that generalize beyond the scope of our current implementation, and (3) two open-source artifacts: a platform-agnostic python implementation of AdaptiveSound for researchers and developers, and an Android app for end-users.

[1] Note that this is a very low-effort compared to training a fully supervised ML pipeline from scratch, which is extremely data-intensive.

2 RELATED WORK

We provided background on DHH culture and sound awareness needs as well as contextualize our work within prior sound awareness systems and relevant machine learning techniques.

2.1 DHH Culture and Sound Awareness Needs

Designing usable sound awareness technology requires understanding of the diverse sound awareness needs of the DHH community. For many DHH people, the preference for assistive technology is not determined by the degree of hearing loss alone; their cultural identity also plays a role–*that is,* whether they identify as Deaf (capital 'D'), deaf (small 'd'), or hard of hearing [5, 49]. Individuals identifying as Deaf follow an established set of norms, behaviors, and language set in the 'Deaf culture' [6, 25, 32]. In contrast, the terms 'deaf' and 'hard of hearing' indicate people for whom deafness is primarily an audiological experience and who refrain from membership to a particular community [6, 32]. These individuals do not have a distinct cultural identity of their own, and they may choose to interact with either hearing or Deaf people based on their comfort.

Prior surveys have found different preferences for the desired sounds among the cultural groups [4, 8]. For example, hard of hearing people may desire speech sounds more than Deaf people, who may be more accustomed to non-vocal ways of living [4, 8]. While acknowledging these differences, prior work also highlights several synonymous needs within the DHH groups [4, 8, 17, 29]. For example, among the different possible sound characteristics (*e.g.,* volume, pitch, duration), the identify of a sound (*e.g.,* "dog barking" or "door knocking") was the most desired, with all DHH cultural groups ranking safety-related sounds (*e.g.,* siren, fire alarms) higher than sounds identifying human activities (*e.g.,* footsteps, door knocks) and appliance alerts (*e.g.,* kettle whistles, dryer beeps) [4, 8]. This preference is often mediated by the social context (*e.g.,* friends *vs.* strangers) [8, 18]. For example, identifying speech of a friend signaling you is important, but not of a passerby talking on a phone. This points to the importance of end-user customization of sound recognition systems.

2.2 Sound Awareness Systems for DHH Users

Common sound awareness technologies used by DHH people include flashing doorbells and vibratory wake-up alarms. While these technologies can substitute certain auditory information (*e.g.,* doorbells, alarm clock) with visual and haptic feedback, they do not serve as a general alternative to many other environmental sounds.

In an early work, Matthews *et al.* [29] built a desktop prototype that used sound visualizations (*e.g.,* rings, spectrograph) to convey basic sound information (*e.g.,* pitch, source location). Towards identifying specific sounds, Bragg *et al.* [4] used a Gaussian Markov Model (GMM) to classify two sounds (door knock and alarm clock) on a mobile app.

More recent work aimed to classify a greater number of sounds with pre-trained deep-learning models [18, 19, 40]. For example, Sicong *et al.* [40] leveraged convolutional neural networks (CNNs) to build and evaluate a smartphone-based app that sensed and classified nine environmental sounds (*e.g.,* door knock, bell ringing). Jain *et al.* [18] extended this work and built a smarthome sound awareness system that recognizes 19 sounds (*e.g.,* microwave beeps, water running) in the homes of DHH users. The same team later developed a portable sound recognition solution that uses a small pre-trained CNN model to classify 20 sounds locally on a smartwatch [19]. User study of the smartwatch app in three campus locations (an office, a lounge, and a bus stop) indicated the usefulness of the sound recognition technology for DHH users, but participants complained about frequent misrecognition, especially in the outdoor "bus stop" context [19]. In a follow-up survey [20], participants expressed

their strong desire to provide feedback to the sound recognition model to improve its recognition accuracy, with all participants being comfortable with putting in 20 minutes of effort in each context.

We build on the above work by contributing the first end-to-end system that allows DHH users to provide iterative feedback to the sound recognition model, showing a significant accuracy improvement over state-of-the-art pre-trained model systems (+14.6%), after about 10 minutes of end-user effort (+18.1% after about 20 minutes).

2.3 Relevant Machine Learning Techniques

Traditional supervised training paradigms, used by common sound recognition systems [16, 18, 19, 28], require a large amount of in-situ data, and are not practical for use in diverse environments. For more modest training data sizes, relevant machine learning approaches include transfer learning [42], a supervised training method that uses limited training examples to fine-tune a model previously trained on large datasets from a different domain (*e.g.,* image classification). Likewise, meta-learning approaches [44] allow models to recognize previously unseen classes with very few labeled training instances. While promising, the above methods require that the full dataset is available beforehand, and do not allow for sequential adaptation on the go.

Our approach leverages incremental learning, a learning approach that continuously improves a model's knowledge through new data without fully retraining the model [12]. We append this new data in a reinforced way, using the user actions as our "data points" to adapt the model through trial and error [38]. This way, the model learns to perceive and interpret its environment over time. This recently emerging incremental reinforcement learning technique has gained popularity for several crucial tasks such as image classification [48] and online learning [45]. However, no work has used it in the context of sound classification–our focus.

Closest to our approach are *ListenLearner* [47] and *ProtoSound* [20], in that they support end-user interaction with a sound recognition model. In both systems, user feedback is integrated at the beginning of the machine learning pipeline (*i.e.,* the model input side) to provide labeled sound data for model training. Whereas our approach integrates user feedback at the end of the machine learning pipeline (*i.e.,* the output side) to provide corrective feedback to the model output. This enables different behaviors: while ListenLearner and ProtoSound train the model on new, unseen sound classes, our approach increases model performance on the existing sound classes (and, in the future, can be combined with either of the two approaches to support new classes).

3 ADAPTIVESOUND: ADAPTING SOUND RECOGNITION THROUGH USER FEEDBACK

AdaptiveSound incorporates continuous yet subtle end-user feedback in the sound recognition pipeline to adapt the model to new contexts and environments. AdaptiveSound leverages the fact that sounds can undergo a lot of modifications in the real-world: they can amplify, overlap with each other, be of varying duration, and reverberate in a room. Hence, the sounds heard in the users' specific sonic environment could vary a lot, and it is impossible to capture the full variation in any dataset used to train the model. Consequently, pre-trained models that are used in common sound recognition systems may not work with high accuracy (as past work [18, 19] also shows) and fine-tuning the model in users' own sonic environments is needed.

To provide data for this fine-tuning, AdaptiveSound leverages end-user's feedback in a low-effort manner. This approach is supported by feedback from DHH people in several prior studies [13, 18–20], where participants felt comfortable with providing some amount of feedback to the sound recognition model's output to improve its recognition accuracy (about 20 minutes of user effort in each context was deemed comfortable). As well, we worked closely with members of the DHH community throughout the design of AdaptiveSound, including one of our authors

who is DHH. A notable challenge identified was how people who are themselves unable to hear sounds provide this feedback? For this, DHH participants in a prior study [14] suggested leveraging visual cues such as seeing a microwave beeping or a door opening to supplement their feedback. Of course, these visual cues may not always be available, but fortunately, AdaptiveSound does not need the feedback *every time*. It benefits whenever the feedback is available, and once a small amount of feedback is provided (*e.g.,* about 5-10 minutes of end-user effort), the system can function independently with greater accuracy, without requiring more feedback.

Below, we describe AdaptiveSound's primary machine learning technique and additional technical features to support real-world use.

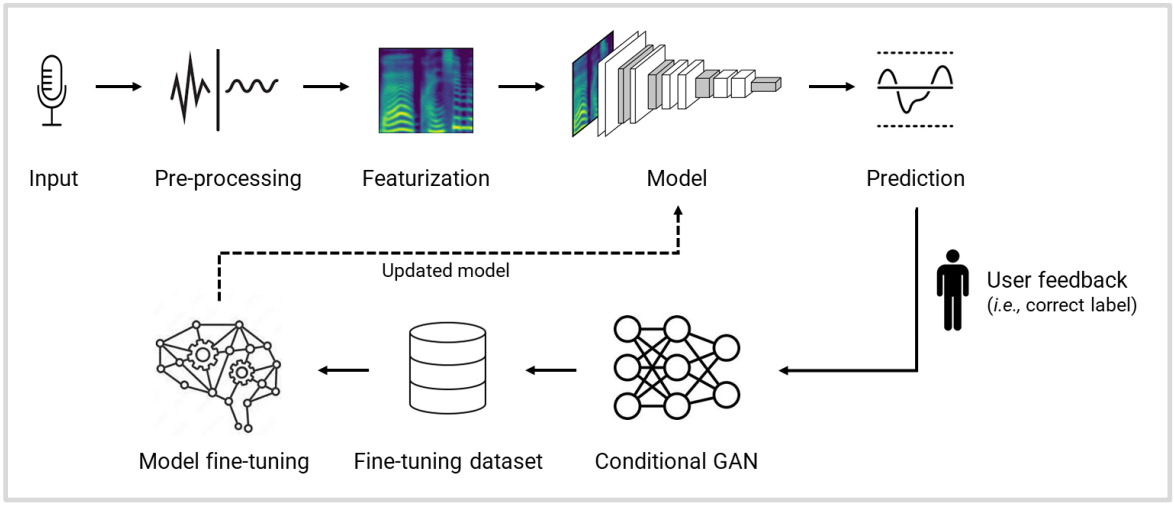

Figure 2: AdaptiveSound's architecture and data flow.

3.1 Primary Technique: Incremental Reinforcement Learning

Common training paradigms such as supervised or transfer learning do not carry the class weights forward and require that the full training data is available for training [9, 42]. Consequently, any new incoming data will need to be appended to the existing data, with the model re-trained on the full dataset. Using these paradigms for sequential learning will waste significant time, memory, and computation and will become impractical if new user data comes in frequently.

Thus, we chose to use *incremental learning* [12], a training approach that continuously improves a model's knowledge through new data without fully retraining the model. In this approach, the model weights from any previous training are "carried forward" and knowledge gained from new data is added to the updated network. By requiring training on only the new data, incremental learning significantly reduces the training time and compute requirement, and is very suitable for deployment on small portable commodity devices such as smartphones, smartwatches or small embedded hardware (*e.g.,* Raspberry Pi).

Figure 2 describes AdaptiveSound's architecture and data flow. The new data comes in from the user in a *reinforced* way; hence, the resultant technique is called *incremental reinforcement learning* [45]. When a model predicts a sound, the user has an option to provide a positive or negative feedback to the output– *that is,* they indicate whether the prediction is correct or not (and in the case of incorrect prediction, can also indicate the correct sound category). The system then uses the correct label provided by the user with the recording of the sound to

incrementally fine-tune the model. Figure 7 describes a possible user interface, a smartphone app, to interact with AdaptiveSound.

Incremental reinforcement learning (IRL) allows a model to continuously adapt to end-user feedback, increasing the system's reliability in users' specific contexts and environments. While this technique is theoretically promising, it also introduces several practical concerns when deployed in the field for our specific use case of sound recognition. Below, we discuss these concerns and our technical strategies to mitigate them.

3.2 Concern 1: Some Sounds Occur More Frequently Than Others

To maintain balance across all classes, incremental reinforcement learning (IRL) requires that the samples of each class are inputted nearly at the same frequency. In the field however, some sounds (*e.g.,* water running, door opening/closing) occur more frequently than others (*e.g.,* fire alarms, sirens), which could overfit the model on the frequent sounds. To avoid this, we use conditional generative adversarial networks (conditional GAN or cGAN) which are commonly used in the field to handle class imbalances [31], although other approaches are also possible such as changing the evaluation metrics or resampling[15, 33]. In short, cGAN essentially creates a dummy data for all the classes already trained in the system when re-training with the latest user data. Compared to other competing approaches, cGAN has the advantage of faster convergence [7, 31].

3.3 Concern 2: User Feedback Can Be Unreliable

Like most machine learning approaches, IRL requires clean training samples to perform effectively. However, user feedback can be unreliable, which in-turn can reduce model reliability. Fortunately, carefully designed user-interfaces can reduce user errors, but what if the user is unsure about their feedback?

Historically, interfaces have been designed to deal with uncertainty of user feedback [22, 37, 46], a notable example being search engines that allow users to select among possible search options if they are unsure of the correct search term. Studies have shown that such interfaces not only help improve accuracy of the resultant system, but also enhance the user experience by allowing the user to have a greater agency in the system's decisions [37, 46].

AdaptiveSound uses a tunable parameter α to incorporate the surety of user feedback. Users can specify how sure they are of their feedback on a scale of 1-5 (5 being the surest), and AdaptiveSound proportionally sets the learning rate for training on the incoming sample.

To demonstrate this intuitively, say the user is extremely sure of their label feedback and enters the value of 5. In this case, the learning rate is set as high and the model moves faster towards this data point, giving a higher priority to this new data. Similarly, when the user is not sure, the learning rate is set to be low, and the model moves slowly towards the newly imputed data.

3.4 Concern 3: Memory and Compute Requirements

IRL stores and trains on the incoming user data over time, which could cause memory to bloat on smaller edge devices (*e.g.,* smartphones, embedded hardware). Fortunately, since we use an incremental training approach, the user data can be deleted once trained on, saving memory. However, repeated training on a single user data every time can be computationally intensive. To mitigate this, we use a middle ground approach: training on small batches of incoming data (*e.g.,* 10-20 new samples) instead of a single new data. This keeps the memory constant and helps maintain a manageable compute. As seen from our field trials, AdaptiveSound can train on portable commodity

devices (*e.g.,* smartphones) in nearly real-time (less than a second to a couple of seconds). User privacy is also preserved, since the samples are stored locally and discarded after training.

3.5 Implementation

3.5.1 Noise-filtering

To filter background noises, AdaptiveSound uses a dynamic thresholding scheme. The sound sensing processing pipeline triggers when the loudness level sensed from the microphone (dBFS) is 1.4 standard deviation higher than the mean of the past minute. We acknowledge that this strategy may leave out persistent background noises that could be important to detect (*e.g.,* AC noises)—and future work should look into this—but we ignore these sounds for computational efficiency. DHH participants in past work [4, 17] too did not prefer knowing about these sounds.

3.5.2 Sampling

For all sounds that pass our noise threshold, AdaptiveSound samples them into 3-second windows. Selecting an optimal sampling window is challenging. If the window is too small, long-term variations in the sounds are not captured. Conversely, if the window is too long, detecting boundaries between consecutive sound events becomes difficult. Fortunately, based on our internal experiments, we were able to select a 3-second window based on the average duration of all sound events in our dataset described below.

3.5.3 Feature Extraction

For each three-second sound, we compute the short-time Fourier Transforms using a 25ms sliding window and 10ms step size for the frequency range from 20Hz to 8000Hz, which yields a 96-length spectrogram. We then convert our linear spectrogram into a 64-bin log-scaled Mel spectrogram and generate a 300 X 64 input frame for every three seconds of audio. These log-mel spectrograms are low-dimensional representations of the input data, which can be used to determine the kinds of sounds that may be occurring, but sensitive information such as spoken content cannot be recovered. To these spectrograms, we apply Cepstral Mean and Variance Normalization (CMVN) [41] before inputting into our model.

3.5.4 Model Architecture

AdaptiveSound uses a *MobileNetV2* architecture [39], a lightweight CNN for mobile devices, measuring about 8MB. We train the model using a cross entropy loss function with an Adam optimizer [23]. The model outputs the classification confidence associated with different sound classes, which we pass through a sigmoid activation function, and display the most probable sound class (or the top-1 prediction).

3.5.5 Deployment Platform

Our pipeline is implemented using Tensorflow-Lite and is conducive to deployment on any smartphone device (*e.g.,* an Android phone) or a portable embedded system (*e.g.,* Raspberry Pi). For our user study, we implemented a user-facing Android application, but our python pipeline code is open-sourced (github.com/AccessibilityLab/AdaptiveSound) for researchers and practitioners to deploy on a device of their choice (and can be extended to support custom applications).

4 EXPERIMENTS

Before evaluating with users, we performed quantitative experiments with AdaptiveSound on real-world sound datasets and compared its performance with two baseline state-of-the-art sound recognition approaches.

4.1 Experimental Setup

4.1.1 Training set

We trained the AdaptiveSound model on the training set compiled from six online sound effect libraries—*Freesound* [10], *Network Sound* [34], *TUT* [30], *UPC* [43], *BBC*, [3], and *TAU* [1]. Each of these libraries provides a collection of high-quality, pre-labeled sound effects. We used sound effects since prior work has shown that models trained on high-quality sound effects adapt well to field use [27]. From these libraries, we downloaded clips for 22 sound categories that were preferred by DHH people in past work [18, 19] (see Table 1). All clips were converted to a single format (16KHz, 16-bit, mono) and silences greater than one second were removed, resulting in 29.2 hours of training data.

4.1.2 Evaluation set (fine-tuning and test set)

Since commonly used 'synthetic' sound classification benchmarks (*e.g., ESC-50* [35] and *UrbanSound8k* [38]) do not mimic the real-world conditions (*e.g.,* background noise, overlapping sounds), we created a 'naturalistic' evaluation dataset by compiling datasets of real-life sound recordings from two prior HCI works [18, 19]. This dataset contains samples for the same 22 sounds as our training set, but recorded by hearing researchers in a total of 21 real-world locations (*e.g.,* homes, university labs, lounges, parks, and urban streets). These recordings were converted to the same format as the training set (16KHz, mono) and the resultant dataset, spanning 4.5 hours, was split into two parts—the fine-tuning set and the test set—with 25% and 75% split respectively.

All sounds (*N*=22)	Microwave, Hazard alarm, Baby crying, Alarm clock, Cutlery, Water running, Door knock, Cat Meow, Dishwasher, Car horn, Phone ringing, Washer/dryer, Bird chirp, Vehicle, Door open/close Doorbell, Dog bark, Kettle whistle, Siren, Cough, Snore, Speech
Home context (*N*=18)	Microwave, Hazard alarm, Baby crying, Alarm clock, Cutlery, Water running, Door knock, Cat Meow, Dishwasher, Phone ringing, Washer/dryer, Door open/close, Doorbell, Dog bark, Kettle whistle, Cough, Snore, Speech
Office context (*N*=10)	Microwave, Hazard alarm, Cutlery, Water running, Door knock, Phone ringing, Door open/close, Kettle whistle, Cough, Speech
Outdoor context (*N*=12)	Hazard alarm, Water running, Cat meow, Car horn, Phone ringing, Bird chirp, Vehicle, Door open/close, Dog bark, Siren, Cough, Speech

Table 1: List of real-world sounds in our dataset. Note that some sounds (*e.g.,* dog bark, water) were recorded in multiple contexts.

4.1.3 Sequential Learning Task

We implemented a simple incremental training pipeline to test performance. First, we pre-trained the model on our training set. Then, we iteratively fed in three-second sound samples from the fine-tuning set and, depending on whether the model predicted the sound sample accurately, we incrementally trained with the correct label of the sample. Finally, we evaluated the performance of the fine-tuned model on our test set.

4.1.4 Baselines

We compared AdaptiveSound's performance with two state-of-the-art sound recognition systems: a generically "fully-supervised" pre-trained model system, *SoundWatch* [19], and a meta-learning-based system, *ProtoSound* [20]. The latter uses a prototypical network based few-shot learning algorithm to fine-tune the model on limited input samples. We trained the SoundWatch model on the training set. For ProtoSound, we pre-trained the model on the training set and fine-tuned it on five samples of each sound category from the fine-tuning set.

4.2 Results

4.2.1 Overall Accuracy

Figure 3 shows the change in accuracy of AdaptiveSound with increased incremental feedback. At about mid-point in the graph, which is about 100 feedback steps, the average accuracy across all sound categories was 93.8% (*SD*=4.9%), which is significantly better than ProtoSound (*avg*=85.9%, *SD*=5.8%) and SoundWatch (*avg*=79.2%, *SD*=8.1%); pairwise t-test for AdaptiveSound *vs.* ProtoSound yielded t_{21}=7.6, p<.001 and AdaptiveSound *vs.* SoundWatch yielded t_{21}=12.3, p<.001. As shown from field evaluation of AdaptiveSound mobile app with DHH users (Section 5 below), it took about 10 minutes for the users to give 100 correctional feedbacks to the model (*avg*=9.8 minutes, *range*=6.1-12.9 minutes), suggesting that the system is able to show significantly improved performance over the state-of-the-art with little end-user effort.

Of course, the accuracy increases as more feedback is provided to the system (Figure 3), but the rate of increase decreases over time, indicating that a few initial feedback steps are enough to make the model usable in a context. Indeed, as seen in the figure, the change of accuracy was very low (about 0.5% per 20 feedback steps) at about 200 feedback steps.

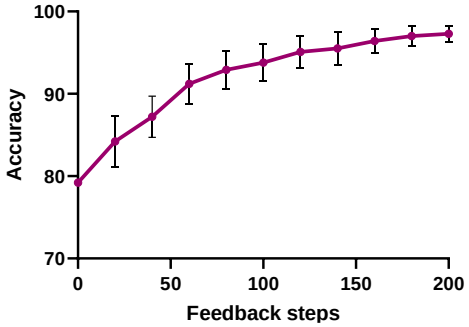

Figure 3: Change in accuracy of AdaptiveSound with number of feedback steps.

4.2.2 Context-Specific Accuracy

Our test set contains samples from three contexts: homes (kitchen, bedroom, and living room), offices (university labs and lounge), and outdoors (parking lots, parks, and streets). As sound quality may vary across contexts, we also calculated the context-specific accuracy of AdaptiveSound. Figure 4 shows our results. As expected, the accuracy was higher for quieter environments (homes and office, *avg*=97.8%, *SD*=1.9%) than for noisier environments (outdoors, *avg*=85.7%, *SD*=5.7%). For the outdoor context, we compared the accuracy with the two baselines, and found that the accuracy difference between AdaptiveSound and other approaches was even more

pronounced for this context (+11.5% over ProtoSound and +23.3% over SoundWatch) than the average accuracy difference noted above (+7.9% over ProtoSound and +14.6% over SoundWatch). This increased performance for outdoor contexts was perhaps because sounds can overlap or attenuate much more in environments of variable noises, and fine-tuning in those environments, as done by AdaptiveSound, will greatly help the model.

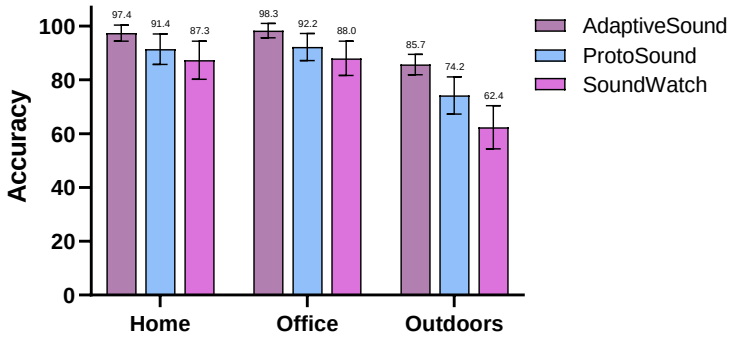

Figure 4: Context-specific accuracies of AdaptiveSound and the two baseline approaches.

4.2.3 Class-Specific Accuracy

Finally, we compared AdaptiveSound's accuracy on different sound classes. Results are shown in Figure 5. Of the 22 sound classes, the accuracy was highest for microwave (99.2%), followed by dog bark (98.9%) and hazard alarm (98.8%). The worst performing classes were alarm clock (87.3%), water running (87.8%), and dishwasher (88.0%). On investigating further, the low-performing classes sounded similar to other classes in our dataset and were understandably confused (*e.g.,* "alarm clocks" were often confused with "phone ringing"; "water running" sounds were confused with "dishwasher" sounds). Figure 6 visualizes the low-dimensional t-SNE embeddings for samples from the five mid-range sound classes.

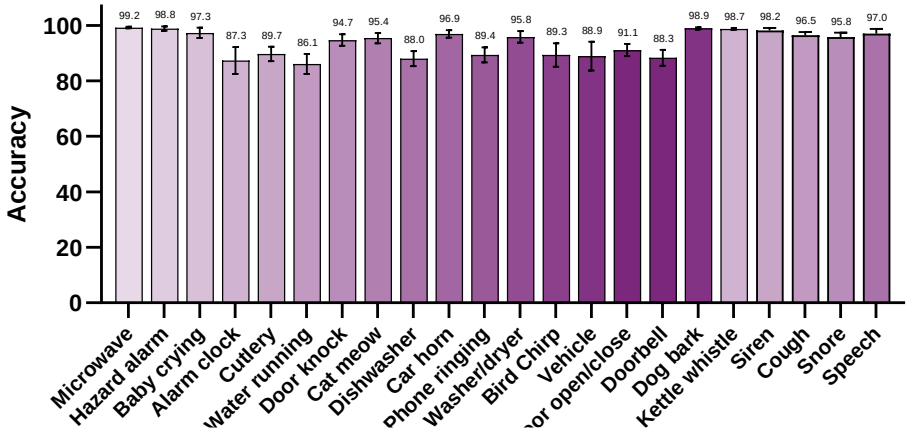

Figure 5: Class specific accuracy of AdaptiveSound.

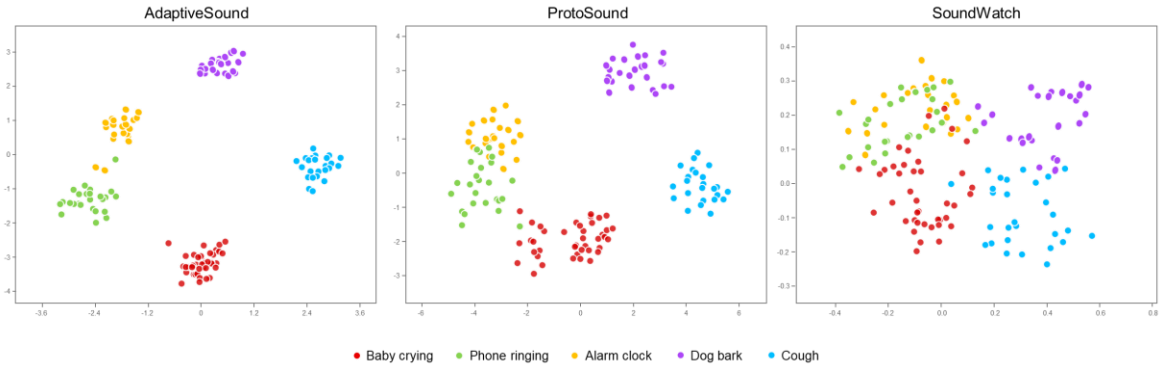

Figure 6: Low-dimensional t-SNE projections of embedding obtained from the three approaches for five sound classes. Note that the sound samples of the same class are clustered together for AdaptiveSound while they diverge from each other for the other two.

5 FIELD STUDY WITH DHH USERS

To assess how DHH users may feel about using AdaptiveSound–including giving correctional feedback–we deployed our system using a real-time Android application and performed a 3-day field evaluation with 12 DHH people.

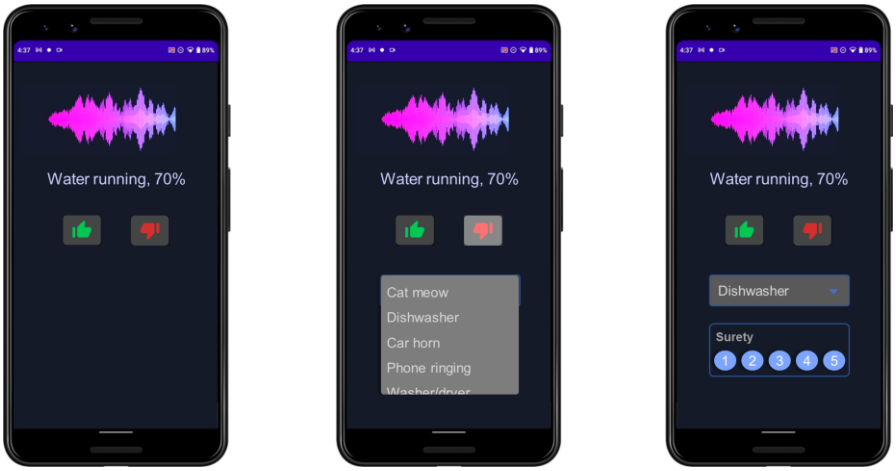

Figure 7: Our user-facing Android app to interact with AdaptiveSound pipeline.

5.1 Real-Time Android App

To allow participants to interact with our AdaptiveSound pipeline, we prototyped a user-facing Android application.

5.1.1 User Interface

The app user interaction flow is shown in Figure 7. After classification, the app displays the predicted sound along with the classification confidence (Figure 7 left). On this output, the user has an opportunity to enter whether the predicted sound is correct or incorrect by clicking on the thumbs up or thumbs down button (shown in green and

red respectively in the figure). If the user indicates that the predicted sound is correct (*i.e.,* they click the thumbs up button), the app continues predicting and stays in the same UI state shown in Figure 7 left. However, if the user indicates that the prediction is incorrect (thumbs down button), the app moves to the Figure 7 middle state, whereby it allows the user to indicate the correct sound label by selecting that label from the drop-down menu. After selecting a label, the user can input how certain they are about their choice by indicating a surety number on a 5-point scale (Figure 7 right). This is intended for a situation where a DHH user may not be sure that they picked the correct label. The 5-point scale was chosen because it provided a nice balance between user agency and effort. After surety value is selected, the feedback cycle ends, and the app returns to the first UI state (Figure 7 left). Beyond the feedback mechanism, the app also shows a three-second waveform at the top to help DHH users as they identify the correct sound for feedback.

When the user is not actively using the app, it runs in the background with notifications for recognized sounds delivered as default Android notifications (not shown in the figure).

5.1.2 Training Backend

Model training occurs in the background, in batches of five data, label pairs. Each time a user gives positive or negative feedback, the correct data, label pair is stored until five pairs are obtained. After training, this data is deleted. Note that we also use the positive feedback for training, since doing so increases model reliability by adding additional samples to the correctly predicted class. For negative feedback, the surety value is also used for training using the algorithm described in section 3.3.

5.1.3 Implementation

The app is built using TensorFlow-Lite, which allowed us to implement the complete training pipeline locally on the device, without any interactions with the cloud. This on-device implementation helps protect data privacy, reduces the training time (since no upload/download is required), and alleviates the need to maintain an active internet connection.

To collect study data, we also implemented automated logging within the app that captures aggregated data on the number and the type of recognized sounds among the 22 supported categories (Table 1). We also log the user interactions to provide feedback. However, to protect participant privacy, no raw sound data is collected.

Our app code is included in the GitHub repository: https://github.com/AccessibilityLab/AdaptiveSound.

5.2 Participants

We recruited 12 DHH participants through social media posts, campus flyers, and snowball sampling. Six participants identified as men, five as women, and one as non-binary. The average age was 27.4 years (*SD*=18.5 years, *range*=19-57). Five participants self-reported a profound hearing loss, four reported severe, two reported moderate, and one reported moderate-to-severe. Eight participants onset as congenital, two reported one years of age, one reported four years, and one reported 11 years. Nine participants reported using hearing devices: five used hearing aids, and four used cochlear implants. For communication, six participants prefer using sign language, and six prefer to communicate verbally. As we presented some written instructions to the participants in the study, we asked about their fluency with reading English, especially since many members of the DHH community use sign as their first language. All participants reported complete fluency with reading English (5/5 on rating scale, 5 is best). For the study interviews, we provided an option to participants to request any disability accommodations: five

participants asked for a sign language interpreter, four asked for a real-time captioner, and three chose to participate verbally. Participants were compensated $60 for the study.

5.3 Study Procedure

We conducted a 3-day field evaluation, and used a pre-post study interview to collect feedback. In the pre-study interview, we invited the participants to our campus lab, where the lead researcher explained our AdaptiveSound app and provided a short demo. Participants were then given our Android phones and encouraged to use the installed AdaptiveSound app for 3-days in different locations they visit, while providing feedback to the model. After 3-days, participants came back to our lab and took part in a semi-structured interview, where the lead researcher asked about their overall experience with our app, interaction with the user interface, and any future improvement suggestions.

5.4 Data Analysis

We collected data using pre-post study interviews and through automatic app logs. The interviews were transcribed and analyzed using thematic analysis. We used Braun & Clarke's six-phased approach. One researcher skimmed the transcripts to familiarize themselves with the data (step 1) and consulted with the research team to generate an initial codebook (step 2). The researcher then iteratively applied the codes to the interview transcripts while refining the codebook (step 3). Another researcher then used the final codebook to independently code the transcripts (step 4). The inter-rater reliability between the two coders, measured using Krippendorff's alpha [24], was 0.84 (>0.8 is considered a very good agreement) and the raw agreement was 95.5%. The codebook was organized into sections (step 5), and the team collaboratively wrote the narrative (step 6). For the app logs, we used a combination of descriptive and inferential statistics to summarize them.

5.5 Findings

We describe the participants' usage patterns, overall experience, and specific comments on the user interface of the AdaptiveSound app. Quotes are drawn directly from the interview transcripts, but are slightly edited for grammar.

5.5.1 Usage Summary

According to the app logs, participants used the app for about six hours each day (*avg*=6.1hours, *range*=4.6-9.4 hours), for a total of about 18 hours of usage per person over the 3-day study period. Regarding specific locations, participants used the app at homes, urban streets, parks, in-transit (*e.g.,* in car or bus), at work, restaurants, and malls. On average, 264.3 sound events were recognized by our per day (*SD*=73.5). Participants provided feedback on about 25% of those recognized sounds on average (*avg*=67.4, *SD*=21.6). As expected, the amount of feedback decreased over time: it was highest on the first day (*avg*=81.7, *SD*=12.3) and lowest on the third day (*avg*=45.3, *SD*=29.1). When inquired about the cause, participants mentioned that since they the app was used in certain context (*e.g.,* at homes) and for certain sounds (*e.g.,* water running) repeatedly, they *"did not feel like I had to provide more [feedback] input on the same sounds"* (P6). This is also synonymous with AdaptiveSound's design, where the user only needs to provide limited initial feedback in a given context for the system to behave accurately in that context, and then no more feedback needs to be given.

When asked about what strategies they used to discern the correct sound label to provide feedback, participants mentioned using *"visual cues"* to look for the source of sound mostly (*N*=12), but also, in some cases, using the help

of hearing roommates (*N*=3). All participants rightly explained that looking for the source was not always possible, but they were *"happy to train when the [source] can be seen"* (P1), in the hope that *"when it is not visible, the app can notify me of the sound."* (P12)

5.5.2 Overall Experience

Since the app was evaluated with DHH users, who may not hear all sounds, we could not get an accurate quantitative estimate of the system's accuracy from the users. However, qualitatively, all participants expressed that the app's accuracy did increase after providing feedback, saying, for example:

> *"I could see that initially, my cat's sound was not being detected. It kept showing it as a microwave beep. So, I entered into the app that it's a cat's meow. [...] After a couple of times, it started detecting my cat accurately and never went wrong after that" (P2)*

Another said:

> *"It takes some time to work. Initially, it would misclassify 1 out of every 3 or 4 sounds. And then, when I did click, click, correct sound [initiating entering feedback into the app], [...], it started to work so much better." (P9)*

Initially, AdaptiveSound behaves like a genetically trained model system like SoundWatch [19] since it directly uses the pre-trained model. After a few feedback steps, the model starts performing better as our participants detailed above, indicating the improved accuracy of the incremental reinforcement learning approach over a baseline pre-trained model system. To further provide anecdotal evidence, five participants in our pool have experienced generically trained sound recognition systems before–including iOS's sound recognition feature and SoundWatch–and all explicitly expressed that AdaptiveSound is *"much more reliable"* (P7) and *"accurate"* (P5, P10).

This increased accuracy enabled participants to get awareness about important sounds and helped with supporting everyday tasks–such as *"understand[ing] that [my partner] has come home by knowing that a door has been opened"* (P5) or "*that it's time to get my dishes out from dishwasher"* (P1) or *"[that] the faucet is dripping [...] in [the] bathroom"* (P3).

5.5.3 User Interface

All participants found the process to input feedback "straightforward", "intuitive", and "effortless". They particularly appreciated the UI feature to specify the surety of their feedback, stating for example,

> *"In many cases, I wasn't sure whether the sound I entered was the one that occurred. I was somewhat sure but not 100% sure. [...] It was nice that I can tell this to the system, oh, this time I am about 3 sure [on a scale of 5]." (P4)*

However, many participants (*N*=7) did mention that giving feedback repeatedly could get burdensome. For example, P6 said:

> *"I could see that this [giving feedback] could get annoying for me. I tried the app for 3 days which was just the right amount of time for me, but if am [providing feedback] over and over at home, I wouldn't do it"*

On explaining the system design (that the system only needs a few initial sets of feedback, *e.g.,* for about 10 minutes to adapt to a context), participants were more receptive. For example, the same participant (P6) added:

> *"Oh, so now you explained it to me, I am fine with it. If I only need to do it a couple of times in a set location, I am happy to do it. [...] And then if I go to a new location that I have not gone before in, then I will do it again. [...] Oh yes, it's not gonna be a problem then."*

P7 shared a similar sentiment:

> *"My observation was that, once I provided some responses… maybe about 10 times for every sound… it was able to do well, at a point that I find acceptable. [...] Of course, if I provided more responses, it would do better, but for me about 10 times was fine."*

P7's comment indicates another point: that the system's accuracy is configurable and the participants can choose the level of effort to put in based on the desired accuracy. This is important since different users may have different preferences or hearing levels and may not desire the same level of accuracy from the system as others. P1 adds that the desired accuracy may also depend on the context:

> *"At home, I usually know what's going on since I am aware. So, I just need [the system] to inform me that something is going on [i.e., a sound is occurring but not what type it is], and I can just look around to figure out what it might be. [...] When I am at work though, I want to know exactly what it was. [So,] I will spend more time configuring the system."*

On a different note, participants also provided improvement suggestions for the system. The most common comment was about supporting additional sounds, especially those that are personal to one's situation such as *"my espresso machine's beep"* (P12) or *"my desk's clicking noise"* (P8). This was outside the scope of our current work, but indeed, future work should look into extending AdaptiveSound's capability to support personalized sounds. Another common suggestion, mentioned by four participants, was to show a spectrogram alongside a waveform to help discern the correct sound to provide feedback, but others (*N*=5) were concerned about their ability to learn and recognize patterns from a spectrogram, indicating mixed preferences.

5.5.4 System Performance

For the study, we provide participants with a mid-range Android phone to evaluate our app (Samsung Galaxy S10, 8GB RAM, 2.8GHz processor, release year: 2019). When asked, participants did not report observing a major degradation of battery life or any lags in the app during use, suggesting that AdaptiveSound can run and train on modern Android devices in real-time. This is corroborated with our lab experiments, where we did not observe a significant performance degradation while running our low-effort incremental training pipeline.

6 OPEN SOURCE

For researchers and practitioners to deploy and extend AdaptiveSound, our device agnostic python implementation (with our pre-trained model) and our specific Android app implementation are open sourced on GitHub: https://github.com/AccessibilityLab/AdaptiveSound.

7 LIMITATIONS AND FUTURE WORK

Our evaluations show that AdaptiveSound can provide accurate real-time sound recognition to DHH users with minimal end-user effort. However, our system and our study have limitations, which we discuss below along with future directions.

7.1 System Limitations

7.1.1 Fixed Set of Sounds

AdaptiveSound supports a fixed number of sound classes (in our study, we used 22 sound classes). While the classes in our study were inspired from sounds preferred by DHH people in several past work [4, 8, 17, 19], some participants wanted the system to support more personalized sound categories such as a custom home appliance or furniture sounds. However, supporting a customizable set of sound categories is challenging since a user's newly entered sound class (*e.g.,* tapping) may be very similar to an existing category (*e.g.,* knocking), making it difficult for the resultant model to distinguish among the two classes. Nevertheless, future work should examine approaches to enable this. One idea to explore is an interactive ML (or IML) system [13, 36] where the end-users can be taught to train an accurate machine learning model by choosing distinct sound classes on their own.

7.1.2 Fixed Sampling Window

AdaptiveSound processes data using a fixed sampling window of three-seconds. While this worked for our purposes, sound classes vary considerably in length—from short-lived (*e.g.,* a gunshot) to longer events (*e.g.,* thunder). If the sampling window is too small, long-term variations will not be captured. Conversely, if the window is too long, detecting boundaries between consecutive sound events is difficult. Thus, future work should explore acoustic event detection techniques (*e.g.,* sub-frame processing [21]) to automatically segment sound events from microphone data.

7.1.3 Does Not Filter Abrupt Noises

Our noise filtering strategy, while removing constant background noise, does not deal with abrupt noises (*e.g.,* an object dropped on the floor, a sudden shout). Filtering abrupt noises is challenging and is an ongoing research topic. We invite future iterations of our approach that allows filtering of these sudden unwanted sounds.

7.1.4 Requires DHH Users to Enter Feedback

AdaptiveSound requires DHH users to enter the correct sound label. This may not always be possible since DHH users may not be able to hear or access those sounds. We reaffirm that AdaptiveSound does not require people to enter feedback every time. It can adapt the model with minimal feedback (about 100 steps as our experiments indicate), whenever it can be given (*e.g.,* when the user can see the sound source). Furthermore, we implemented two features to assist DHH people in entering feedback: (1) a visual cue in the form of a waveform that can indicate that a sound may be occurring, and (2) a rating scale to enter surety of the feedback. Future work should explore other interface features such as showing spectrograms, the location of origin, or low-dimensional embeddings of real-time audio to further assist DHH people in identifying the correct sound for feedback.

7.2 Study Limitations

7.2.1 Constrained Dataset

We evaluated AdaptiveSound's performance on a real-world dataset compiled from two prior works [18, 19] instead of on common machine learning benchmarks (*e.g.,* ESC-50 [35], UrbanSound8k [38]) since these high-quality benchmarks of clean sound files and do not accurately represent real-world acoustic conditions (*e.g.,* overlapping sounds, background noise). A notable exception is Google's *AudioSet* [11], a common benchmark that contains sounds from 8-million YouTube videos, many of them recorded in real-world locations. However, the label quality of this dataset is very poor [2] and does not allow for accurate performance evaluation. We enthusiastically invite researchers to collect and evaluate our approach's performance with larger, more varied real-world datasets.

7.2.2 Short Evaluation Period

Our field study evaluated AdaptiveSound for a brief period of 3-days. This demonstrates the promising potential of our incremental learning approach, but does not account for longitudinal use where the long-term effects of technology novelty and fatigue will be more visible. Future work should conduct longitudinal deployments and compare findings.

7.2.3 Sound Recognition may not be Universally Desired

While our work is heavily informed by DHH perspectives and past work [18, 19], we do not assume it is universally desired or that it will necessarily work, as designed, for all users. Some DHH people may feel negatively towards sound recognition technology, especially those who identify as part of the Deaf culture [6, 26]. However, two recent large-scale surveys [4, 8] that many DHH individuals find sound recognition valuable. AdaptiveSound can be constrained to identify a small subset of sounds (*e.g.,* a baby crying) to provide essential situational awareness, while otherwise avoiding the hearing world. Still, future work should co-design and evaluate with a broad sample from the DHH population, given diverse preferences and interests.

8 CONCLUSION

We presented the design and evaluation of AdaptiveSound, the first feedback-loop sound recognition system that allows deaf and hard of hearing (DHH) individuals to provide subtle feedback to a pre-trained sound recognition model to adapt the model to diverse acoustic environments. AdaptiveSound is inspired by past evaluations of pre-trained sound recognition systems, where DHH users desired the ability to provide feedback to the model output. Our evaluations with real-world sound datasets and with 12 DHH people in the field suggests that AdaptiveSound significantly outperforms state-of-the-art sound recognition systems, and achieves usable accuracy when trained on conventional devices (*e.g.,* consumer smartphones) with little end-user effort.

ACKNOWLEDGMENTS

We acknowledged the contribution of people from Professor Dhruv Jain's previous lab at the University of Washington, where the initial work for this research was performed, particularly Jon E. Froehlich, Leah Findlater, and Steven Goodman. We also thank Hriday Chhabria and Liang-Yuan Wu for their help on the AdaptiveSound Android application.

REFERENCES

[1] Adavanne, S., Politis, A. and Virtanen, T. 2019. TAU Moving Sound Events 2019 - Ambisonic, Anechoic, Synthetic IR and Moving Source Dataset [Data set]. Zenodo.

[2] AudioSet Label Accuracy: *https://research.google.com/audioset/dataset/index.html*. Accessed: 2021-04-06.

[3] BBC Sound Effects: *http://bbcsfx.acropolis.org.uk/*. Accessed: 2019-09-18.

[4] Bragg, D., Huynh, N. and Ladner, R.E. 2016. A Personalizable Mobile Sound Detector App Design for Deaf and Hard-of-Hearing Users. *Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility* (New York, New York, USA, 2016), 3–13.

[5] Cavender, A. and Ladner, R.E. 2008. Hearing impairments. *Web accessibility*. Springer. 25–35.

[6] Cavender, A. and Ladner, R.E. 2008. Hearing impairments. *Web accessibility*. Springer. 25–35.

[7] Chrysos, G.G., Kossaifi, J. and Zafeiriou, S. 2018. Robust conditional generative adversarial networks. *arXiv preprint arXiv:1805.08657*. (2018).

[8] Findlater, L., Chinh, B., Jain, D., Froehlich, J., Kushalnagar, R. and Lin, A.C. 2019. Deaf and Hard-of-hearing Individuals' Preferences for Wearable and Mobile Sound Awareness Technologies. *SIGCHI Conference on Human Factors in Computing Systems (CHI).* (2019), 1–13.

[9] Finn, C., Abbeel, P. and Levine, S. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. *International Conference on Machine Learning* (2017), 1126–1135.

[10] Fonseca, E., Pons Puig, J., Favory, X., Font Corbera, F., Bogdanov, D., Ferraro, A., Oramas, S., Porter, A. and Serra, X. 2017. Freesound datasets: a platform for the creation of open audio datasets. *Hu X, Cunningham SJ, Turnbull D, Duan Z, editors. Proceedings of the 18th ISMIR Conference; 2017 oct 23-27; Suzhou, China.[Canada]: International Society for Music Information Retrieval; 2017. p. 486-93.* (2017).

[11] Gemmeke, J.F., Ellis, D.P.W., Freedman, D., Jansen, A., Lawrence, W., Moore, R.C., Plakal, M. and Ritter, M. 2017. Audio set: An ontology and human-labeled dataset for audio events. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2017), 776–780.

[12] Gepperth, A. and Hammer, B. 2016. Incremental learning algorithms and applications. *European symposium on artificial neural networks (ESANN)* (2016).

[13] Goodman, S.M., Liu, P., Jain, D., McDonnell, E.J., Froehlich, J.E. and Findlater, L. 2021. Toward User-Driven Sound Recognizer Personalization with People Who Are d/Deaf or Hard of Hearing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*. 5, 2 (2021), 1–23.

[14] Goodman, S.M., Liu, P., Jain, D., McDonnell, E.J., Froehlich, J.E. and Findlater, L. 2021. Toward User-Driven Sound Recognizer Personalization with People Who Are d/Deaf or Hard of Hearing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*. 5, 2 (2021), 1–23.

[15] Gosain, A. and Sardana, S. 2017. Handling class imbalance problem using oversampling techniques: A review. *2017 international conference on advances in computing, communications and informatics (ICACCI)* (2017), 79–85.

[16] Hands-on with iOS 14's Sound Recognition feature that listens for doorbells, smoke alarms, more: *https://9to5mac.com/2020/10/28/how-to-use-iphone-sound-recognition-ios-14/*. Accessed: 2021-03-08.

[17] Jain, D., Lin, A.C., Amalachandran, M., Zeng, A., Guttman, R., Findlater, L. and Froehlich, J. 2019. Exploring Sound Awareness in the Home for People who are Deaf or Hard of Hearing. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (2019), 94:1-94:13.

[18] Jain, D., Mack, K., Amrous, A., Wright, M., Goodman, S., Findlater, L. and Froehlich, J.E. 2020. HomeSound: An Iterative Field Deployment of an In-Home Sound Awareness System for Deaf or Hard of Hearing Users. *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2020), 1–12.

[19] Jain, D., Ngo, H., Patel, P., Goodman, S., Findlater, L. and Froehlich, J. 2020. SoundWatch: Exploring Smartwatch-based Deep Learning Approaches to Support Sound Awareness for Deaf and Hard of Hearing Users. *ACM SIGACCESS conference on Computers and accessibility* (2020), 1–13.

[20] Jain, D., Nguyen, K., Goodman, S., Grossman-Kahn, R., Ngo, H., Kusupati, A., Du, R., Olwal, A., Findlater, L. and Froehlich, J. 2021. ProtoSound: A Personalized, Scalable Sound Recognition System for d/Deaf and Hard of Hearing Users. *SIGCHI Conference on Human Factors in Computing Systems (CHI)* (2021), 1–16.

[21] Karbasi, M., Ahadi, S.M. and Bahmanian, M. 2011. Environmental sound classification using spectral dynamic features. *2011 8th International Conference on Information, Communications & Signal Processing* (2011), 1–5.

[22] Kay, M., Kola, T., Hullman, J.R. and Munson, S.A. 2016. When (ish) is my bus? user-centered visualizations of uncertainty in everyday, mobile predictive systems. *Proceedings of the 2016 chi conference on human factors in computing systems* (2016), 5092–5103.

[23] Kingma, D.P. and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*. (2014).

[24] Krippendorff, K. 2018. *Content analysis: An introduction to its methodology*. Sage publications.

[25] Ladd, P. and Lane, H. 2013. Deaf ethnicity, deafhood, and their relationship. *Sign Language Studies*. 13, 4 (2013), 565–579.

[26] Ladd, P. and Lane, H. 2013. Deaf ethnicity, deafhood, and their relationship. *Sign Language Studies*. 13, 4 (2013), 565–579.

[27] Laput, G., Ahuja, K., Goel, M. and Harrison, C. 2018. Ubicoustics: Plug-and-play acoustic activity recognition. *The 31st Annual ACM Symposium on User Interface Software and Technology* (2018), 213–224.

[28] Live Transcribe & Sound Notifications – Apps on Google Play: *https://play.google.com/store/apps/details?id=com.google.audio.hearing.visualization.accessibility.scribe*. Accessed: 2021-04-04.

[29] Matthews, T., Fong, J., Ho-Ching, F.W.-L. and Mankoff, J. 2006. Evaluating non-speech sound visualizations for the deaf. *Behaviour & Information Technology*. 25, 4 (Jul. 2006), 333–351.

[30] Mesaros, A., Heittola, T. and Virtanen, T. 2016. TUT Sound events 2016.

[31] Mirza, M. and Osindero, S. 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*. (2014).

[32] Moore, M.S. 1992. *For Hearing people only: Answers to some of the most commonly asked questions about the Deaf community, its culture, and the" Deaf Reality"*. Deaf Life Press.

[33] Narwane, S. V and Sawarkar, S.D. 2019. Machine learning and class imbalance: A literature survey. *Ind. Eng. J.* 12, (2019).

[34] Network Sound Effects Library: *https://www.sound-ideas.com/Product/199/Network-Sound-Effects-Library*. Accessed: 2019-09-15.

[35] Piczak, K.J. 2015. ESC: Dataset for environmental sound classification. *Proceedings of the 23rd ACM international conference on Multimedia* (2015), 1015–1018.

[36] Ramos, G., Meek, C., Simard, P., Suh, J. and Ghorashi, S. 2020. Interactive machine teaching: a human-centered approach to building machine-learned models. *Human--Computer Interaction*. 35, 5–6 (2020), 413–451.

[37] Ribicic, H., Waser, J., Gurbat, R., Sadransky, B. and Gröller, M.E. 2012. Sketching uncertainty into simulations. *IEEE Transactions on Visualization and Computer Graphics*. 18, 12 (2012), 2255–2264.

[38] Salamon, J., Jacoby, C. and Bello, J.P. 2014. A Dataset and Taxonomy for Urban Sound Research. *22nd {ACM} International Conference on Multimedia (ACM-MM'14)* (Orlando, FL, USA, 2014), 1041–1044.

[39] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A. and Chen, L.-C. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), 4510–4520.

[40] Sicong, L., Zimu, Z., Junzhao, D., Longfei, S., Han, J. and Wang, X. 2017. UbiEar: Bringing Location-independent Sound Awareness to the Hard-of-hearing People with Smartphones. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*. 1, 2 (2017), 17.

[41] Strand, O.M. and Egeberg, A. 2004. Cepstral mean and variance normalization in the model domain. *COST278 and ISCA Tutorial and Research Workshop (ITRW) on Robustness Issues in Conversational Interaction* (2004).

[42] Torrey, L. and Shavlik, J. 2010. Transfer learning. *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*. IGI global. 242–264.

[43] UPC-TALP dataset: *http://www.talp.upc.edu/content/upc-talp-database-isolated-meeting-room-acoustic-events*. Accessed: 2019-09-18.

[44] Wang, Y., Yao, Q., Kwok, J.T. and Ni, L.M. 2020. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys (CSUR)*. 53, 3 (2020), 1–34.

[45] Wang, Z., Chen, C. and Dong, D. 2021. Lifelong incremental reinforcement learning with online Bayesian inference. *IEEE Transactions on Neural Networks and Learning Systems*. 33, 8 (2021), 4003–4016.

[46] Wardekker, J.A., van der Sluijs, J.P., Janssen, P.H.M., Kloprogge, P. and Petersen, A.C. 2008. Uncertainty communication in environmental assessments: views from the Dutch science-policy interface. *Environmental science \& policy*. 11, 7 (2008), 627–641.

[47] Wu, J., Harrison, C., Bigham, J.P. and Laput, G. 2020. Automated Class Discovery and One-Shot Interactions for Acoustic Activity Recognition. *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (2020), 1–14.

[48] Wu, X., Huang, W., Wu, X., Wu, S. and Huang, J. 2022. Classification of thermal image of clinical burn based on incremental reinforcement learning. *Neural Computing and Applications*. (2022), 1–14.

[49] Zajadacz, A. 2015. Evolution of models of disability as a basis for further policy changes in accessible tourism. *Journal of Tourism Futures*. 1, 3 (2015), 189–202.