A Human-in-the-Loop System for Sound Event Detection and Annotation

BONGJUN KIM and BRYAN PARDO, Northwestern University, USA

Labeling of audio events is essential for many tasks. However, finding sound events and labeling them within a long audio file is tedious and time-consuming. In cases where there is very little labeled data (e.g., a single labeled example), it is often not feasible to train an automatic labeler because many techniques (e.g., deep learning) require a large number of human-labeled training examples. Also, fully automated labeling may not show sufficient agreement with human labeling for many uses. To solve this issue, we present a human-in-the-loop sound labeling system that helps a user quickly label target sound events in a long audio. It lets a user reduce the time required to label a long audio file (e.g., 20 hours) containing target sounds that are sparsely distributed throughout the recording (10% or less of the audio contains the target) when there are too few labeled examples (e.g., one) to train a state-of-the-art machine audio labeling system. To evaluate the effectiveness of our tool, we performed a human-subject study. The results show that it helped participants label target sound events twice as fast as labeling them manually. In addition to measuring the overall performance of the proposed system, we also measure interaction overhead and machine accuracy, which are two key factors that determine the overall performance. The analysis shows that an ideal interface that does not have interaction overhead at all could speed labeling by as much as a factor of four.

CCS Concepts: • **Information systems** → *Search interfaces*; • **Human-centered computing** → *Interactive systems and tools*;

Additional Key Words and Phrases: Interactive machine learning, sound event detection, human-in-the-loop system

ACM Reference format:
Bongjun Kim and Bryan Pardo. 2018. A Human-in-the-Loop System for Sound Event Detection and Annotation. *ACM Trans. Interact. Intell. Syst.* 8, 2, Article 13 (June 2018), 23 pages.
https://doi.org/10.1145/3214366

1 INTRODUCTION

Sound event detection and annotation involves labeling of sound events and providing estimates of their time positions (i.e., start and end) in an audio recording. Sound event detection is a key technology with applications in many areas: labeling speech recordings with speaker names [26], labeling music recordings by predominant instrument [11], labeling nature recordings with the species of animals heard in the recording [19], and identifying gunshots in city recordings [33]. These labeling tasks can be manually done by human annotators or automatically done by machine.

The reviewing of this article was managed by special issue associate editors Marco Gillies and Rebecca Fiebrink.
This work was supported by National Science Foundation Grant 1617497.
Authors' addresses: B. Kim, Northwestern University, 2133 Sheridan Road, Evanston, IL, 60208, USA; email: bongjun@u.northwestern.edu; B. Pardo, Northwestern University, 2133 Sheridan Road, Evanston, IL, 60208, USA; email: pardo@northwestern.edu.

https://doi.org/10.1145/3214366

Even though manual sound event annotation by human experts leads to more accurate results than using automatic detection systems, there are many situations where hand-labeling events in recordings is prohibitively labor-intensive. For example, speech and language pathologists often wish to label sound and speech events in day-long (24-hour) recordings of patient environments. One such recording can take several work days to label. Therefore, many researchers have put significant effort into developing more accurate automatic sound recognition systems. The typical approach to building automatic sound recognition systems uses supervised machine learning. Examples include neural networks [15, 23], Gaussian Mixture Models (GMM) [37], decision trees [18], and Support Vector Machines (SVM) [25].

Making a general sound event detection device using these machine learning techniques typically requires a large number of labeled training examples (e.g., thousands or tens of thousands of labeled sounds). It also may require fine-tuning the model for the specific application, usually by machine learning experts. This is not feasible in the case where users do not have thousands of pre-labeled examples of the target sound class. It is also not feasible when providing enough labeled examples is equivalent to solving the task manually (e.g., the user labels the entire 24-hour recording to give the machine enough training data to label the 24-hour recording). Even with lots of training data and model tuning, machine labeling may not be good enough for many applications and frequently diverges from human labeling. For example, the current state-of-the-art environmental labeling for language assessment, LENA [40], agrees with human annotators only 76% of the time on a four-way forced-choice labeling task.

We wish to address sound event labeling tasks that fall in a middle ground: There is too much audio to be practically labeled by hand, yet there are too few training examples to train an accurate statistical model. We want to develop an efficient way to achieve human-level accuracy with much less human effort than is typical for manual annotation. We note that a successful method to speed labeling for these cases would also make it possible to generate enough labeled data for a statistical machine learner to be taught in cases where it is currently prohibitive to label enough data by hand. This would increase the range of sound-objects that could eventually be labeled fully automatically. This is not, however, our primary goal. The goal is to speed the labeling task at hand, rather than to train a generalized machine learning model for later use on different data.

To achieve our goal, we apply a human-in-the-loop approach to sound event detection and annotation. The idea is to engage users in an interactive process [32] to collaboratively label the audio with the machine. Human-in-the-loop machine learning is a technique that has received attention recently as one approach to resolving the limits of fully automated systems. It has been applied in many areas, such as image retrieval systems [32], image foreground extraction [27], image object labeling systems [28], biomedical image recognition [41], natural language processing [29], Network Alarm Triage [2], interactive visualization [31], musical performance [9], and audio source separation [6, 8, 22].

In this article, we present a new human-in-the-loop system for sound event detection and annotation. The system directs the user's attention to the most promising regions of audio for labeling. The user labels these regions and gives the system feedback by labeling and adjusting region boundaries. The system learns from this feedback and updates future recommendations for high-interest regions. To assess the effectiveness of the proposed system for the sound event annotation task, we build a web-based annotation tool and perform a human subject study with potential users of the tool. We evaluate how much the proposed tool speeds up sound event detection tasks. Finally, we also present a method to quantitatively assess the machine's performance and the interaction design separately, as these are two key factors that determine the performance of a human-in-the-loop system.

2 RELATED WORKS

2.1 A Human-in-the-Loop System for Multimedia Retrieval and Annotation

A common approach to labeling large amounts of multimedia data is through crowdsourcing [35] [36]. Even though crowdsourcing annotation is a great way to collect large-scale labeled data, it is not appropriate in a situation where the audio data should be annotated by domain experts or must not be distributed in public, such as audio recordings of patients for clinical purposes. We focus on the case where crowdsourcing is not an appropriate solution.

Interactive learning frameworks that depend on users' relevance feedback have been actively researched in image retrieval. *CueFlik* [10] is a web image search application that allows users to create and adjust rules for concepts (e.g., portraits of people) by providing the machine with positive and negative examples. The user feedback iteratively updates the rules to obtain more accurate image search results. Their interactive approach is aimed at training the best classifier to retrieve images relevant to a query. Our goal is to completely label the audio easily and quickly.

Recognizing objects in images (e.g., circle the cat in this image) is much faster than identifying sound events in audio. The images are viewable all at once. It is easy to scan one's eyes in any direction. However, since audio is time-based media, one listens start-to-finish at a fixed rate. This is typically much slower than scanning a still image, and the user cannot instantly direct her attention to any part of the audio file and perceive it. Another difference is that audio events are often fully overlapped (e.g., a cough is concurrent with television noise and the sound of a blender). This is not occlusion. This is simultaneous overlapped sounds, much more similar to observing something through a reflective glass window where two overlapping images occur. In general, interactive image annotation/retrieval systems provide a user interface where a user can look at sets of images and give the system feedback by clicking the images [1] or selecting sub-regions of the images [34]. The interface design for our sound detection tool focuses on directing the user's attention to promising sub-sections (i.e., the machine's recommendations) of a long audio track for labeling.

2.2 Sound Annotation Tools

Several audio editing applications such as *Audacity* [20] and *Sonic Visualizer* [7] provide an annotation environment where a user manually selects a sub-section of an audio track and labels it. *Sonic Visualizer*, *AudioSculpt* [3], *Audio Brush* [5], and *ASAnnoatation* [4] also provide low-level feature information (e.g., pitch content, repeated structure labeling) by using audio signal processing techniques, but they do not use high-level semantic labels (e.g., Bob's voice) and do not allow user-defined labels.

TotalRecall [17] is a semi-automatic multimedia annotation tool. It automatically detects speech regions on an audio track (speech or non-speech) for audio segments. It helps a user to find speech sections of an audio track easily, but is hard-coded to find speech and cannot be on-the-fly repurposed for detecting other kinds of events.

SoundsLike [13] is a tool to detect user-selected sound events in a movie. It provides a similarity graph that visualizes which audio segments are similar to the user-selected segment as an aid for easier navigation. The system does not update its similarity estimates based on user feedback. Therefore, if the system thinks two segments are similar and the user doesn't, there is no way to correct the system. They also did not evaluate how much the similarity graph helps the annotation process. Finally, the interface does not provide any machine prediction to speed up the labeling process.

Gulluni [14] suggested an interactive approach to analyze electroacoustic music by interactive machine/human labeling of sound objects within a music track. While Gulluni's system does not

allow a user to change the boundaries of segmented regions, our system utilizes boundary adjustment of segments as user feedback to retrain a model. Moreover, their approach uses clustering techniques that require a user to listen to the audio multiple times to determine the best segmentation level. Multiple listenings can be problematic for long audio files (hours long). They also did not conduct human subject studies to evaluate the effectiveness of the system and only tested their system in simulation. In contrast, we performed a human-subject study where participants used our tool and a manual editor to label audio, letting us observe the effectiveness of our approach for speeding labeling. Furthermore, they did not provide a publicly available system. We implemented a web-based sound annotator that anyone can use or build on.

The interactive sound event annotation described in this article has been introduced in our previous work [16]. The work briefly presented the interactive labeling approach and a preliminary analysis of user logs from the human-subject study. The current article is an extended version of the previous work. We describe each component in the human-in-the-loop system in significantly greater detail, particularity focusing more on the user interaction. In the evaluation, we explain the experimental design protocol in more detail than we did in the previous work so that readers can clearly understand how effectively the experiment is designed to evaluate the effectiveness of the proposed tool. We analyze the user log with additional performance measures and conditions, and we measure the interaction overhead that is one of main factors determining the performance of the interface. Furthermore, while the previous work only conducted quantitative analysis, in this article we additionally perform qualitative analysis to understand user satisfaction with our tool.

3 INTERACTIVE SOUND EVENT DETECTION AND ANNOTATION

We now describe an interactive system that lets a single user greatly reduce the time required to label audio that is tediously long for a human (e.g., 20 hours), has target sounds that are sparsely distributed throughout a long-duration recording (10% or less of the audio contains the target), and has too few prior labeled examples (e.g., one) to train a state-of-the-art machine audio labeling system.

3.1 System Overview

Figure 1 shows how our system works with the user to label target sound events. First, a user uploads an audio track into the system and provides an example of the kind of sound she seeks (e.g., someone knocking on a door). This can be done either by selecting a region on the audio track containing a good example sound or by uploading a short example audio file containing an example target sound (e.g., someone knocking on a door). Note that the goal is not to find exact copies of the target sound in the audio file to be labeled. The goal is to find other sounds of the same category (e.g., other door knocking sounds) in the audio file.

The system segments the track into small regions whose length is the same as the initial example and measures features of the audio file (see Section 3.2). It then finds the n regions with features most similar to the example and directs user attention to them by showing them as candidates. The user labels the candidate regions as positive or negative (see Section 3.4), and adjusts the start/stop times of positive examples. Based on this user feedback, the importance of audio features is reweighted to move positive examples closer together and further from the negative examples. Given this new feature space, the system selects a new set of n relevant regions (see Section 3.3) for the user to evaluate. This process of selecting candidate regions for human evaluation is repeated for some number of rounds. As more examples are labeled by the user and the features are reweighted every round, the system's ability to suggest good regions improves.

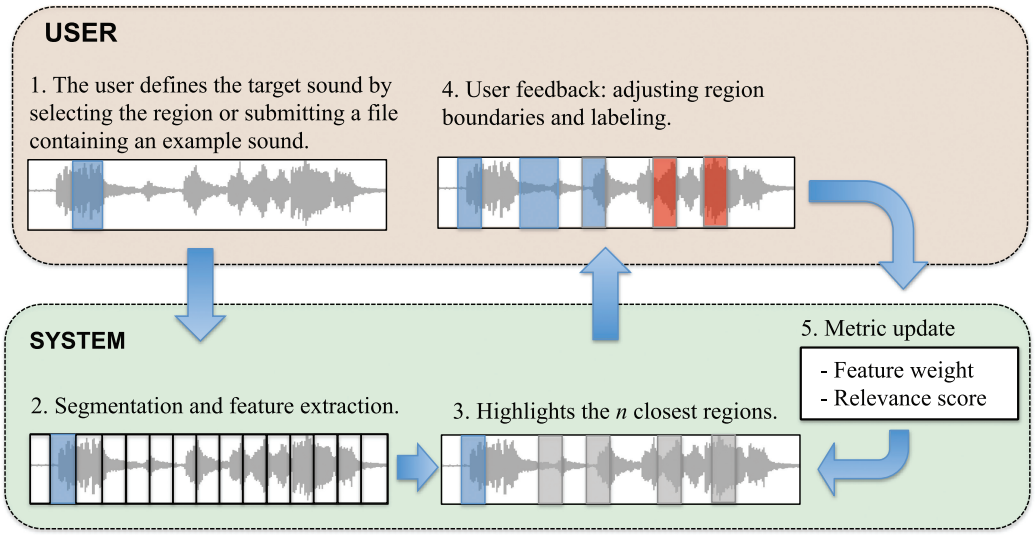

Fig. 1. System overview of the human-in-the-loop sound event annotation.

Active Learning refers to the case where the learner selects the examples to learn from, rather than passively receiving examples chosen by the teacher. This interaction between the user and the system can be thought of as a kind of active learning where the system is learning the feature weights for audio examples by presenting the unlabeled segments it currently estimates to be most similar to the set of positively labeled segments. We show the user the top n results and not the most ambiguous examples because our purpose is to help the user complete the annotation quickly by directing her attention to high-likelihood regions, not to train a machine learning model to fine-tune a decision boundary. When one has a long audio file and only a few target examples, showing the top n results lets users label them more quickly. This is a better strategy to speed up the labeling task at hand.

3.2 Segmentation and Feature Extraction

Once a user provides the initial example to the system (e.g., a 3-second region containing a bird call), the entire track is split into segments whose length is the same as the length of the initial example (e.g., 3 seconds). In the initial phase, all segments have the same length. However, once the user starts labeling the suggested regions, the length of user-labeled segments will vary because the user is allowed to adjust the boundaries of the regions. Given the possibility of varied-length segments, we need a way to measure distance between segments, regardless of segment length.

To measure distances between the segments including the initial example, audio features are extracted over each fixed-length segment. Our system extracts the first 13 Mel Frequency Cepstral Coefficients (MFCCs). MFCCs are widely used in a variety of sound recognition tasks [24]. As shown in Figure 2, each segment is split into a sequence of short frames (e.g., a frame size of 90ms with 50% overlap between adjacent frames), and MFCCs are computed on each frame. The MFCC features extracted frame-wise are pooled over each segment (e.g., 3 seconds) using mean and variance of instantaneous and delta values. The delta values are the difference between feature values of two consecutive frames. These represent basic temporal characteristics of the feature vectors in one segment. As a result, a 52-dimnesional feature vector is built for each segment (13 MFCC averages, 13 MFCC variances, 13 MFCC average delta, 13 MFCC average delta variance).

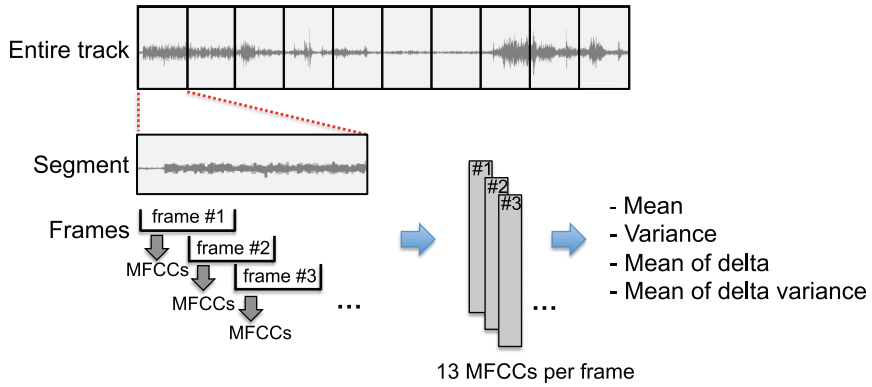

Fig. 2. Audio feature extraction for all segments of an audio file. MFCCs are extracted frame-wise and pooled over each segment using the means and variances of both the instantaneous and delta values.

Using this feature extraction method, varied length segments can be represented as a fixed-length of vector (i.e., 52 dimensions) so that distances between the segments are easily measured in the feature space.

3.3 Relevance Score

In each round, the system measures the distance between each unlabeled segment and the set of positively labeled segments (initially, this set contains only the original target example). Using this distance, it ranks them by decreasing level of relevance and presents the top n segments to a user. To compute the relevance score, we apply a simple nearest neighbor method [12]. The relevance score of an unlabeled audio segment s can be computed as

$$Rel(s) = \frac{d(s, s_n)}{d(s, s_n) + d(s, s_p)}, \tag{1}$$

where s_p is the nearest positively labeled segment to s and s_n is the nearest negatively labeled segment to s. Function $d(a, b)$ is the weighted Euclidean distance between two segments in the feature space. When there is no negative segment (there is always at least one positive example, which is the initial query), the relevance score is computed as

$$Rel(s) = \frac{1}{d(s, s_p)} \text{ if } |neg| = 0, \tag{2}$$

where $|neg|$ means the number of negative segments.

To obtain a more accurate relevance score in each round, the system reweights features using Fisher's criterion [39]. The weight of ith feature is computed as

$$w(i) = \frac{(avg(f_i^p) - avg(f_i^n))^2}{std(f_i^p)^2 + std(f_i^n)^2}, \tag{3}$$

where f_i^p and f_i^n are vectors whose elements are ith feature values in the 52 dimensional (MFCC-based) feature vectors of all positive and negative examples respectively. $avg(x)$ and $std(x)$ indicate the mean and standard deviation of elements in a vector x.

As the ith feature contributes more to better discrimination between positive and negative examples, its Fisher score will increase. The system reweights each feature with the Fisher score based on the current labeled segments (positive and negative) in each round, and the relevance

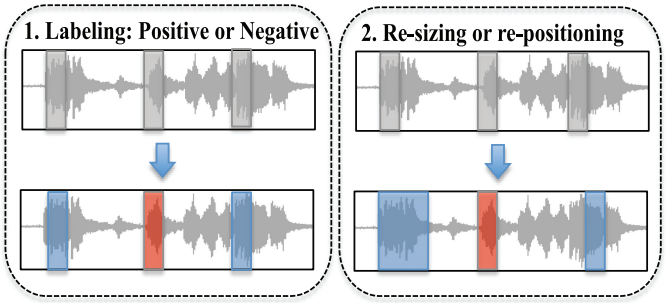

Fig. 3. As feedback, a user labels regions positive (blue)/negative (red) or changes the time position and size.

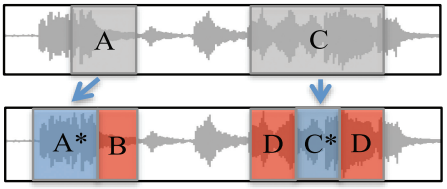

Fig. 4. The region A and C (gray) are the machine's suggestions. A user listened to them, changed the temporal location of A and adjusted boundaries of C, and labeled them positive (A* and C*). Then the system automatically labels the region B and D as negative since it is clear that they do not contain the target sound events.

scores (Equation (1)) over all segments are computed in the updated feature space. We expect that as more labeled segments are collected, the relevance score would become more accurate.

3.4 User Relevance Feedback

The system presents n segments to be labeled every round, and the user adjusts segment boundaries and labels them. Labeling segments plays an important role as feedback for the future rounds because the machine's suggestions for each round depend on the user feedback in past rounds.

A user can provide two types of feedback to the system, as shown in Figure 3. One is to apply positive or negative labels to each candidate example. This type of feedback has been widely used in interactive image retrieval systems [38]. The other type is to adjust boundaries of the suggested region when the region does not cover the whole duration of a target sound event. This type of feedback is typically not used in document or image retrieval systems but is useful for improving retrieval of regions of audio files.

Our system automatically collects additional negative examples from the user's boundary adjustments. As shown in Figure 4, for example, suppose the user changes the position of the region (A) and labels it as positive. In this case, we can obtain not only one positive example, but also one negative example, which is the region (B) that the user did not select after listening to it. In the same way, adjusting boundaries of region (C) generates negative examples (D). This automatic negative labeling is beneficial in two ways: (i) A user implicitly labels more regions in one iteration, speeding interaction, and (ii) since our system presents the most relevant examples to a user every round, the pool of labeled examples tends to skew toward positive, which could make measuring the relevance score problematic. Therefore, adding negative examples automatically helps in computing more accurate relevance scores of unlabeled examples.

Fig. 5. Screenshot of the interactive sound annotator.

4 INTERFACE DESIGN AND IMPLEMENTATION

We implemented a web-based interactive annotator as a client–server application. The back-end server is written in Python and deployed as a Flask server. The frontend user interface is written in Javascript. Readers can watch the demo video and try the application out at http://www.bongjunkim.com/ised/.

Figure 5 shows the main workspace of our annotation tool. It consists of these main sections: *Navigation Map*, *Annotation Track*, *Listen and Label These*, and *Already Labeled*. The *Navigation Map* displays a waveform of an entire track and the currently labeled regions so that a user navigates and listens to them easily. The *Annotation Track* is a zoomed-in version of *Navigation Map* where a user can select or adjust regions and label them by clicking and dragging a mouse. The *Listen and Label These* displays the top n candidate regions identified by the machine. These are to be labeled by the user every round.

Figure 6 describes how a user performs the labeling task in each round. The user listens to new regions by clicking the items in the *Listen and Label These* section. If a region does not contain an example of the target sound class, the user labels it as negative by clicking the *Negative* button. If the region has the target sound, the user first adjusts the boundaries of the region so that it fully covers one instance of the target sound class and labels it as positive by clicking the *Positive*

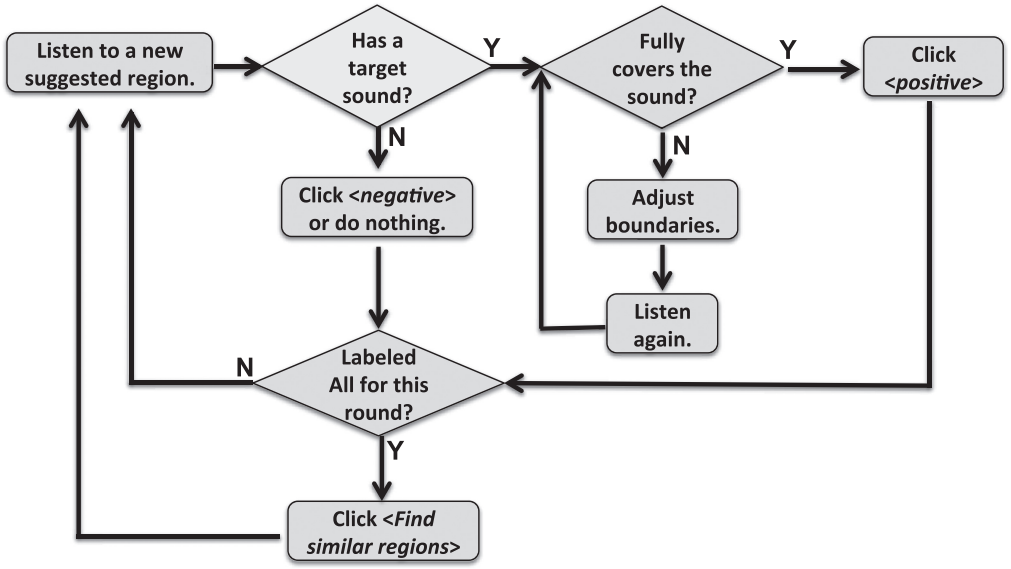

Fig. 6. User interaction flowchart of the interactive annotator.

button. The user labels all the regions in each round and clicks on the *Find Similar Regions* button to submit the feedback to the system and obtain a new set of candidate regions from the system.

The *Already Labeled* section shows regions labeled positive during past rounds. A user can listen to them by clicking each item in the list. Clicking on the *Export Results* button saves the annotation results to text file containing start and stop times of all positive regions.

5 EVALUATION

We conducted a user study to validate the effectiveness of the proposed interactive annotation approach compared to manual annotation. The experiment seeks to answer the following questions:

- Which interface enables participants to label the given audio track faster?
- How accurately did participants label the target sound events using each interface?
- How satisfied are participants with each interface?
- Do participants prefer the proposed interactive annotation to manual annotation?
- What user-interface overhead does the interactive annotation approach impose, compared to manual annotation?

5.1 The Two Interfaces We Compared

The two interfaces we compared were the interactive annotator proposed in this article and a manual annotator, similar to the standard interfaces currently used for manual annotation. In the interactive annotation tasks, a participant submits a file containing an initial target sound event to the system, and then the system presents the five most relevant regions to the user at each round. If the participant thinks a suggested region contains the target sound events, the participant labels it as positive. If the suggested region contains only some portion of the target sound, the participant adjusts the position of the candidate regions and labels it as positive. If it does not contain the target sound at all, it is labeled as negative. The participant keeps labeling in this manner for the given amount of time in each task (15 minutes).

In the manual annotation tasks, participants use an identical interface to the interactive annotator except for the removal of the recommendations from the system. They find target events by listening to the track sequentially (from the beginning to the end) or accessing any time position on the track. If they find the target sound event, they drag a mouse over the region containing the target sound. They keep finding as many sound events as they can for the given amount of time in a task (i.e., 15 minutes).

Both interfaces also provided a control for the user to adjust playback speed (1x, 2x, and 3x). Speeding playback is an alternative way to speed up the search that is commonly available in many multimedia players or annotators such as *Youtube*, *Quicktime Player*, or *Audacity*. It is also known that increasing up to double the rate produces no significant loss in comprehension of speech, but higher playback rates produce a loss in comprehension [21].

To navigate an audio track, one might use a scrubbing function that is available in modern Digital Audio Workstaion (DAWs), where a user drags a cursor on a waveform and the audio samples that the cursor is passing over are played. In our interfaces, the scrubbing feature was not provided because it is mostly useful to find a precise spot where sound characteristics dramatically change (e.g., silence between noisy sections), and it is not useful for recognizing relatively short sound events overlapping each other in a long audio track; therefore, we did not include a scrubbing feature in either interface.

5.2 The Audio Dataset

To evaluate our system we used the dataset from the IEEE Audio and Acoustic Signal Processing Technical Committee challenge on Detection and Classification of Acoustic Scenes and Events (DCASE) [30]. The DCASE dataset is one of a few public datasets for computational analysis of sound events and scene analysis. Moreover, since it has been used in a public challenge, it is well-designed enough to test submitted algorithms properly in various situations. Using a public dataset also allows future systems to compare to our approach under the same conditions.

To generate testing tracks for this experiments, we chose files used for the Office Synthetic (OS) Event Detection Task of the DCASE 2016 challenge.[1] These consist of 2-minute duration mono recordings of sequences containing overlapping acoustic events in an office environment (e.g., coughing, drawer, door knock, speech, etc.).

We anticipate 10 minutes as the minimal length of track where someone might wish to speed search. Therefore, we created two different 12-minute-long audio tracks by concatenating six short tracks in the DCASE dataset to create each track. Each track contains 11 different sound classes with 18 examples of each class in the track. All sound events are randomly distributed over a track. The two tracks, while containing similar sound events, order these events differently. This prevents the learning of ordering details of one track influencing performance on the other.

The sound classes in the two tracks include the following 11 sound events: *door knock, door slam, speech, human laughter, clearing throat, coughing, drawer, keyboard typing, keys, phone ringing, page turning*. The DCASE dataset provides audio files with three different levels (-6, 0, and 6 dB) of the average Signal-to-Noise Ratio (SNR) of events over the background texture. Readers can find more detailed information about the dataset in Stowell et al. [30]. We chose audio files with -6 dB SNR (i.e., the target sound is 6 dB softer than intruding background noises) when generating the testing tracks. This reduces chances that users could obtain additional information about the sound events by looking at the shape of waveform displayed on the screen, as the waveform is not noticeably larger when the target sound class is present if the target class is 6 dB softer than the background.

[1] http://www.cs.tut.fi/sgn/arg/dcase2016/task-sound-event-detection-in-synthetic-audio.

It also makes the problem more challenging for the machine recommender system as sounds softer than the background noise are more difficult for automated systems to detect.

The summed total time that all of the instances of the target class take up in a track is roughly 4% of the entire length of the track. The density of target events and distractor events are roughly 1.5 and 15 (events per minute), respectively. The average inter-onset-intervals of the 18 examples of a sound class is 38 seconds. Since environmental recordings are usually long and have many different kinds of sound events happening in our everyday life, we believe that the generated audio tracks, which have sparsely distributed sound events of each class, are appropriate to evaluate the proposed sound annotator.

5.3 Participant Recruiting

The target users of our tool are people who need to find and label sound events within an audio track for their research activities. These groups of users include speech and language pathologists who need to analyze relationships between children's language development and their listening environment by recording their everyday life. Another group of potential users are researchers who study machine listening since building an automated sound event labeler with supervised machine learning usually requires a number of correctly labeled audio examples of various sound classes as training data. Therefore, we recruited people who study speech and language development or machine learning in the audio domain. We also recruited people who have experienced audio editing or labeling software. Even if people in this group may not be the main target user group, we believe they are appropriate subjects to validate the efficacy of the proposed system against manual annotation since they are familiar with critically listening to audio recordings.

To ensure participants capable of recognizing sound events for the experiments, we performed a hearing pre-test. Participants were given a labeled target sound event to listen to (e.g., speech) and 10 different sound events (3 speech events and 7 other events) were presented. They were asked to select all sound events that belong to the same label as the target sound (e.g., speech), and they had to correctly label all target sound events to pass the test. We performed the hearing test twice per subject with two different target sounds (i.e., speech and door knock). These were the target sound classes used in the actual annotation tasks. This let the listening test also implicitly train participants on the range of variation to expect for target sounds in the actual tasks.

We did not limit gender and age of participants as long as they belonged to the target group mentioned, but all recruited subjects were older than 18 since we recruited people who have experienced at least college-level research activities related to sound (e.g., speech and hearing, machine listening). Subjects' native language did not matter as long as they could understand the experimental instructions written in English. In total, 20 subjects who met all requirements were recruited, and each subject performed two annotation tasks using two different interfaces. All sessions were conducted using a desktop computer and headphones in a quiet room.

5.4 Task Procedure

Each subject participated in one session. One session included a hearing test, training on the interfaces, two annotation tasks (one on the proposed interface and one on the manual interface), and survey questions about their experience. Each session lasted about 1 hour.

To reduce the chances that the order of presentation of interfaces would influence the results, half of the participants tested the proposed interface first and the other half tested the manual interface first. As with interfaces, half of the participants were presented Task 1 (the first audio file) first, and half were presented Task 2 (the second audio file) first. As a result, 20 participants were divided into four groups so that task and interface order was balanced:

- Group1: Manual, Task 1 → Interactive, Task 2
- Group2: Manual, Task 2 → Interactive, Task 1
- Group3: Interactive, Task 1 → Manual, Task 2
- Group4: Interactive, Task 2 → Manual, Task 1

We selected two different sound type for users to detect: one from physical objects, *door knock* and the other from human voice, *human speech*. Thus, participants labeled *door knock* in Task 1 and *human speech* in Task 2. These sounds were selected to be quite different from each other, as human speech is complex, varied, and harmonic, while door knocks are transient, percussive sounds. We hypothesized that there might be differences in how good the system recommendations would prove for these qualitative different sounds.

Each task includes a training session for a subject to learn how to label audio using each interface. In the training session, participants were given the exact same task as in the testing phase, except for the recording to be labeled. For training, we chose a 2-minute-long recording from the dataset for the DCASE 2013. It contains 15 classes of sound events (including speech and door knock sound) recorded in an office environment. Participants were required to spend at least 4 minutes practicing the labeling task. If they wanted to practice more, they were allowed to practice the labeling task as long as desired. No participant, however, chose to spend more than 4 minutes on training. Participants were also allowed to ask any questions about the task and the interface during and after the training session.

For each task, participants were asked to find as many regions containing the target sound class (e.g., door knock) as they could within 15 minutes. We believe that 15 minutes are enough for a user to label a 12-minutes-long audio track even when they listen to the entire track sequentially from beginning to end. For the interactive annotator, an example target sound file was provided for the user to submit to the system as the initial query, which is one of two ways of submitting an initial query to the interactive annotator, as described in Section 3.1. We chose this method because we only wanted to measure the benefits of the interactive loop against the manual method. Time that users would spend finding the first query in audio to be labeled would vary depending on the position of the target sound on the audio track and how they search for it. The initial query given to users is not one of the 18 examples of the target classes on the audio track. Therefore, regardless of which interface (manual or interactive) was used, each participant was given 18 examples to label in each task.

After each task, the participant was asked to identify his or her level of agreement with the following statements:

- I had a clear understanding of the task.
- I understood how to use this interface to achieve the given goal.
- I was satisfied with using this interface.
- I was able to label target sound events easily.

To do this, the participant was given a slider for each question that was labeled as ranging from strongly disagree (0) to strongly agree (1). After the entire session (both sound labeling tasks) was done, participants were asked a set of questions comparing the two interfaces:

- Which interface was easier to use?
- Which interface was easier to learn?

Additionally, they were provided a free-form comment box where they could leave any feedback about interfaces or tasks.

5.5 Performance Measures

The machine's role in our proposed system is to direct the human's attention to the most likely portions of the audio, not to determine whether something is a member of the target class or not. The human makes the final determination. One can think of this system as an attention model instead of a classifier. Therefore, we sought to measure how quickly a user found regions containing target sound events within a recording. How classification accuracy of a machine learning model changes over time is not our focus.

As participants labeled audio, the system recorded the positions of labeled regions whenever labeled regions were added or updated. Based on the recorded log, we evaluated a user's labeling performance by measuring the proportion of sound events correctly detected by a user as a function of time spent on the task (0 to 15 minutes). We considered a sound event correctly labeled if the temporal position of user-labeled region overlaps sufficiently with the temporal position of its ground truth with 1-second tolerance.

5.6 Results

Log data and answers to questionnaires were collected from 20 participants. The participants reported having an average of 42 months of experience using audio editing or labeling software. Moreover, 13 out of the 20 participants have used audio editing software for longer than 1 year, and 7 of them have prior experience using software to label sound events.

5.6.1 The Absolute Performance from Log Data. Each participant was asked to find as many target sounds as they could for 15 minutes within a 12-minute recording using two different interfaces, a manual and an interactive annotator. Figure 7 shows how quickly the participants labeled target sound events over time. Figure 7(a) shows the proportion of the target sound events detected by the 20 participants as a function of time they spent. Figure 7(b) and (c) shows the performance of the two different tasks respectively (labeling door knock and speech). Figure 7(d) shows the performance from two different group of participants (7 people who have prior experience using software to label sound events and the 13 who do not have this experience). In Figure 7(a), (b), and (c), lines indicates median, and the dark and light bands of each color show the 75th and 25th percentiles. Lines in Figure 7(d) indicate the median value of a pair of each user group and interface. The median user of the proposed interface labeled all target examples within the time given.

Overall, it took an average of 517 seconds for participants to label all target sound events using the interactive annotator (15 rounds per user). This is roughly half the time it took the manual labelers. Specifically, Figure 7(a) shows that our interactive annotator lets one find about 80% of the target sound events in about 350 seconds. To achieve the same performance using the manual annotator, it takes 720 seconds (1-second tolerance). Therefore, we can conclude that the interactive annotator helped participants find sound events roughly twice as fast as did the manual annotator.

Figure 7(b) and (c) shows which task took participants more time in labeling each target sound. It turned out that labeling speech events took less time than labeling door knock events regardless of which interface they used. This difference probably came from the acoustic characteristics of the two sound events. One instance of knocking sound consists of multiple nonharmonic and short sound events (e.g., *knock-knock-knock*). So, participants had to listen to the audio to find the exact start and end positions of one instance of a knocking sound event. On the other hand, recognizing the start and end positions of speech is relatively easy.

We also examine the difference between participants who have prior experience using software to label sound events and participants who do not. As shown in Figure 7(d), there is no big difference between them in terms of how quickly they found the target sound events. We can conclude

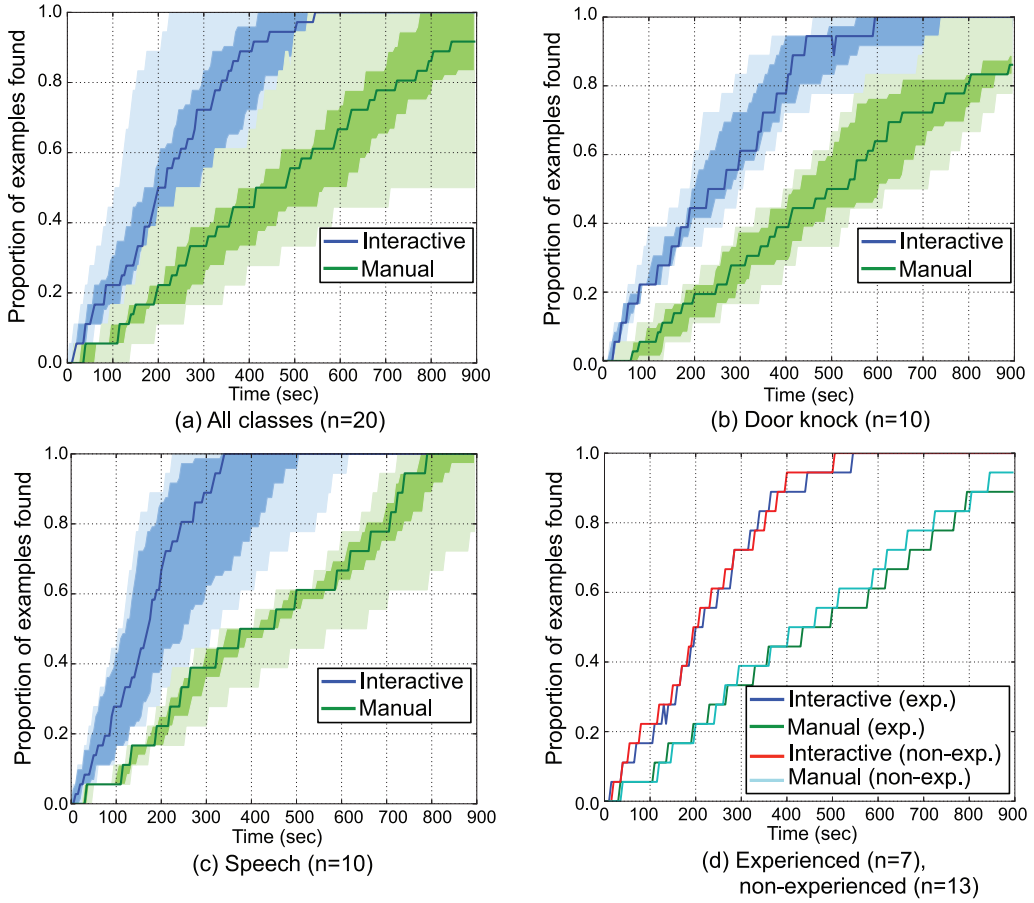

Fig. 7. The proportion of target sound instances found as a function of time spent on the task (quantized every 5 seconds) using two different interfaces: the proposed interactive interface and the manual annotator. Here, N = 20, as each of 20 participants tried both interfaces in a session. (a)-(c): The proportion of examples found as a function of time spent labeling for all classes (a), knock only (b), and speech only (c). Lines indicates median, and dark and light bands of each color show 75th and 25th percentile. (d): The performance of two different participant groups: experienced and nonexperienced. Lines indicates median value of a pair of each user group and interface.

that our interface speeds labeling by roughly the same amount regardless of a user's prior experience with labeling audio.

5.6.2 Self-Reported Performance. Figure 8 shows participants' level of agreement (0 to 1.0) with statements about their experience with each interface. We performed statistical tests to validate whether there is a significant difference between mean responses to questions about experiences with the interactive and manual annotator. We used Wilcoxon signed-rank since the data are not normally distributed.

The responses range from 0 (strongly disagree) to 1 (strongly agree). For statement 1 ("I had a clear understanding of the task" and statement 2 ("I understood how to use this interface to achieve the given goal"), there is no significant difference between the two interfaces. This means

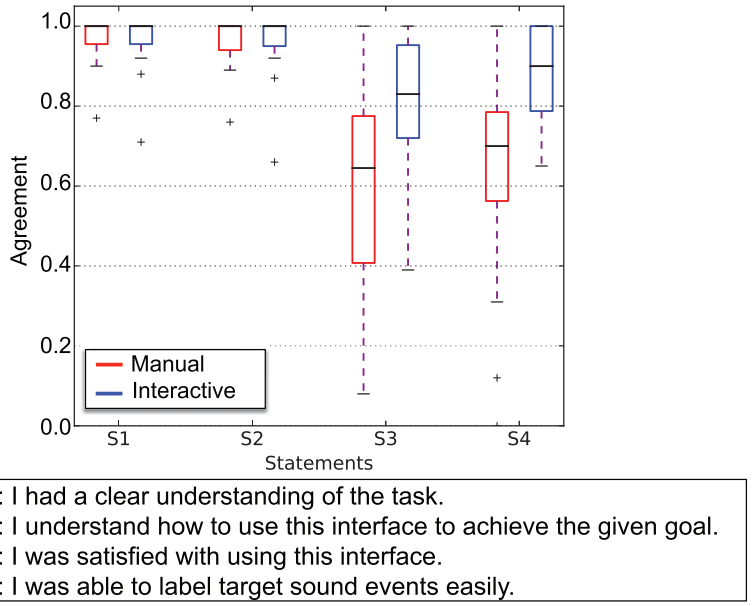

Fig. 8. User responses to questions about experience with each interface. Responses range from 0 (strongly disagree) to 1 (strongly agree). N = 20 for each box plot.

that most of the participants clearly understood how to use both interfaces in the tasks. On the other hand, for statements 3 and 4, there is a significant difference between their mean responses. The mean responses to statement 3 ("I was satisfied using this interface") are 0.588 (manual) and 0.785 (interactive), and they are significantly different ($p < 0.05$). The mean responses to statement 4 ("I was able to label target sound events easily") are 0.667 (manual) and 0.878 (interactive) and they are also significantly different ($p < 0.005$). This indicates that participants felt the annotation task was easier using the interactive annotator.

We also compared responses from the two groups (7 experienced and 13 nonexperienced participants). As shown in Figure 9, in the nonexperienced group, the mean responses to statement 3 are 0.578 (manual) and 0.798 (interactive). The mean responses to statement 4 are 0.657 (manual) and 0.889 (interactive). In the experienced group, the mean responses to statement 3 are 0.579 (manual) and 0.797 (interactive). The mean responses to statement 4 are 0.620 (manual) and 0.857 (interactive). Based on this analysis, we can conclude that the interactive annotator was preferred regardless of users' prior experience using labeling software.

5.6.3 Participants Comments. Participants wrote comments about their experience with each interface in the free-form response box. Many people liked the effectiveness of the recommendation feature of the interactive annotator compared to the manual annotator. Representative quotes include:

"Using the manual annotator, I could not listen to the entire sound signal in the given time. It seems the only way of finding all the target moments is to listen to all the signals, which includes a lot of irrelevant sound samples."

"The interactive annotator did a great job at quickly finding the relevant regions. I even had time to go back and adjust the boundaries of some of the regions."

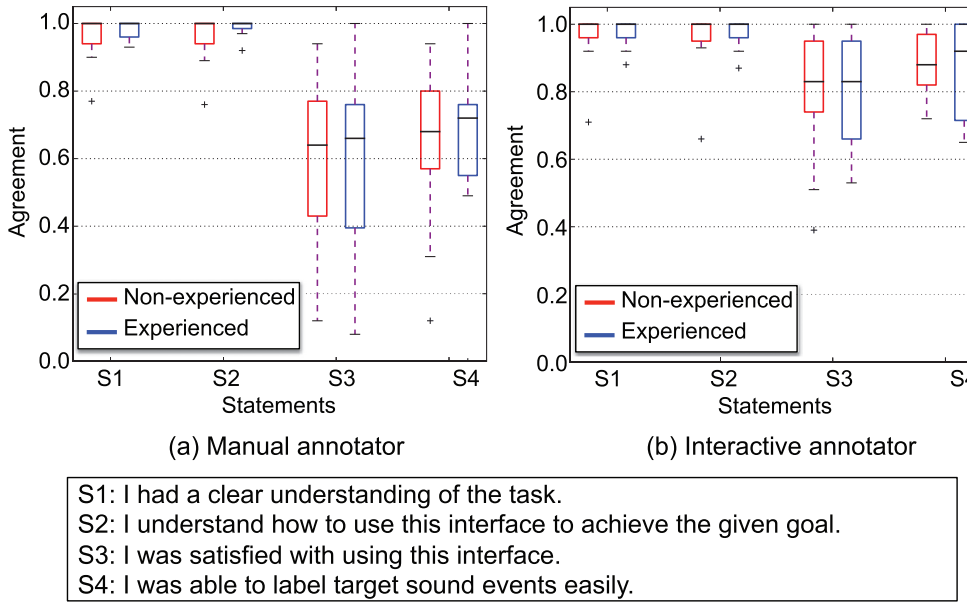

Fig. 9. Comparison between survey responses from two different participant groups. (a): Response about the manual annotator. (b): Response about the interactive annotator. Responses range from 0 (strongly disagree) to 1 (strongly agree). N =13 for nonexperienced participants and N = 7 for experienced participants.

"It is hard to say with 100% certainty that I was able to label the audio more accurately with the interactive annotator, but I spent the last 9–10 minutes not even hearing any human voices, which is assuring."

While their log data showed they could find sound events more quickly, some participants felt the interactive annotator was inefficient or boring because it started to suggest only irrelevant regions after all target events were labeled. Representative quotes about this issue include:

"The interactive one gave me many negative parts."

"I found that, at the beginning of the test, the software found the target sound correctly, but at the end, accuracy decreased. I guess this happened since, at the end, there are not many door knock sounds left, so the software recommends any sound even though it is not like a door knock sound. Even in this case, the software should not recommend wrong target sounds."

"Toward the end of the task, I had established for myself that I had found all the target sounds in the last 5 minutes of the recording. While it was nice not to have to listen to the entire track, it was frustrating each time the system suggested a sound that was not the target sound."

From this set of feedback, it is obvious that, for future work, the interactive annotator needs to have features where a user can be informed of the confidence of current recommendations or can determine when to stop the iterations.

Some participants were not satisfied with the interactive annotator because it was sometimes difficult to figure out sound events only by listening to small snippets of audio, so they often had to make the suggested regions longer and listen to them. From this feedback, it seems that suggesting longer snippets of audio to users than actual prediction may be a good approach.

5.7 Interaction Overhead

Our interactive approach speeds up the labeling task by a factor of two in our user study. Can we make it even faster? There are two primary ways to speed labeling further. The first is to increase

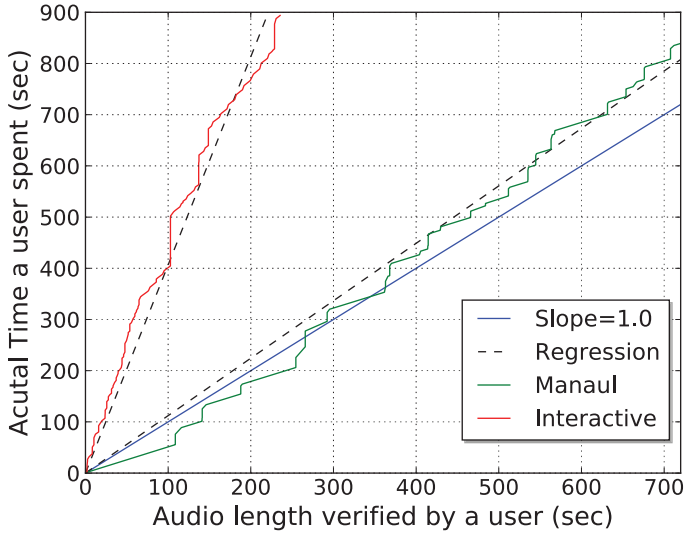

Fig. 10. Total time a participant (one participant is selected as an example) spent on the task as a function of the total amount of audio verified/labeled. The interactive system requires more time to label a fixed amount of audio than the manual system. This interaction overhead partially offsets the speedup of the overall task the interactive system provides by having the users label the most promising segments of audio first.

the accuracy/relevance of the system's predictions. The second is to lessen the overhead imposed on the user by interaction with the system. In this section, we address the issue of interaction overhead.

Recall that the interactive system presents users specific audio segments to listen to and label. Large portions of the audio were never listened to by users of the interactive system as the system deemed them irrelevant for the labeling task. To quantify the amount of interaction overhead, we analyzed how long it took the average participant to label the audio they *did* listen to and compared it to how long it took them to label a similar duration of audio using the manual approach.

Figure 10 shows how long it took one participant to evaluate (i.e., labeling either positive or negative) a fixed amount of audio. Two different colored solid lines (red and blue) indicate the actual experiment data, and the dotted lines are their linear regression lines. For this particular participant, the slopes of the two regression line are 4.07 for the interactive annotator and 1.22 for the manual annotator. The slope value 4.07 means that it takes 4.07 seconds to verify 1 second of audio.

To quantify the extra interaction overhead caused by the interactive approach compared to the manual approach, we use the ratio of the two slopes (i.e., $overhead_{interactive}/overhead_{manual}$). For the participant shown in Figure 10, this ratio is 3.62 to 1. We collected the interaction overhead values for 14 participants (we do not have interaction overhead data for the other 6 participants because we started collecting the log for computing the interaction overhead from the 7th participant). As shown in Figure 11, this ratio ranges from 2.95 to 7.69. The mean is 4.68 (median: 4.22), which means that the interactive system has 4.68 (mean) times more interaction overhead than the manual system. Despite this ratio, the interactive approach still doubles the speed at which users completed the overall annotation task by having the user to label the most promising segments of audio first. It is, however, obvious that reducing the interaction overhead would improve the speedup even further. This way of quantifying interaction overhead is useful when various

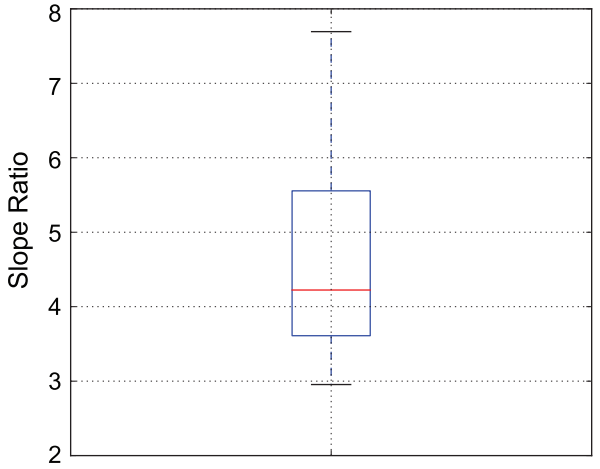

Fig. 11. Overhead ratio (interactive/manual) for 14 participants (Median: 4.22, mean: 4.68).

human-in-the-loop systems need to be compared because it separates out the influence of the recommendation system and the interface for the user to do labeling.

5.8 Machine Accuracy

In addition to reducing interaction overhead, increasing machine accuracy is also a key way to further speed up labeling. So, it is important to measure the performance of the human-in-the-loop system without considering the interaction overhead. Assuming there is no interaction bottleneck in the interactive annotator, we simulated the labeling tasks of a perfect simulated oracle. We measured what proportion of the audio would need to be verified to find all target sounds if the verification were done by the perfect oracle instead of a human without any interaction overhead. The user (i.e., the Oracle) submits one sound target event as the initial query, and then the system presents the five most relevant regions to the Oracle at each round. If each suggested region is matched to one in the ground truth with 200ms tolerance, the region is labeled as positive. If they are partially overlapping each other, the Oracle adjusts the position of the candidate regions and labels it as positive. If they are not overlapping at all, the Oracle labels the region as negative. The Oracle keeps labeling until all sound events are labeled.

We ran this Oracle simulation with the same audio and target sounds (i.e., speech and door knock) as in the user study. Figure 12 shows the proportion of detected sound events as a function of the proportion of audio verified by the Oracle. As shown in Figure 12, the oracle found all sound events of the two classes by evaluating less than 126 seconds of the audio (17.5% of the audio).

5.9 Estimating Actual Annotation Time Using User Simulation and Interaction Overhead

The user simulation is a good way of measuring a machine's accuracy. We can run the simulation on many different kinds of audio tracks easily (which costs a lot for the actual user study). However, this is not a proper method for evaluating the overall performance of our interface. We are not able to measure the actual time that a user would spend to achieve the goal. The result of user simulation is always too optimistic because the interaction overhead is not considered in the simulation.

To solve this issue, we can use the quantified interaction overhead introduced in Section 5.7. In the previous section, we figured out that the interaction overhead of our interactive annotator is

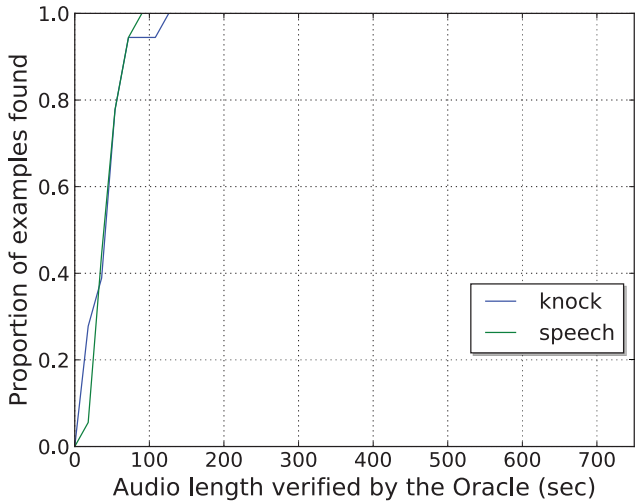

Fig. 12. The user simulation. The proportion of examples found according to the proportion of audio the Oracle evaluated. All sound events are detected by evaluating less than 126 seconds of the audio (17.5% of the audio).

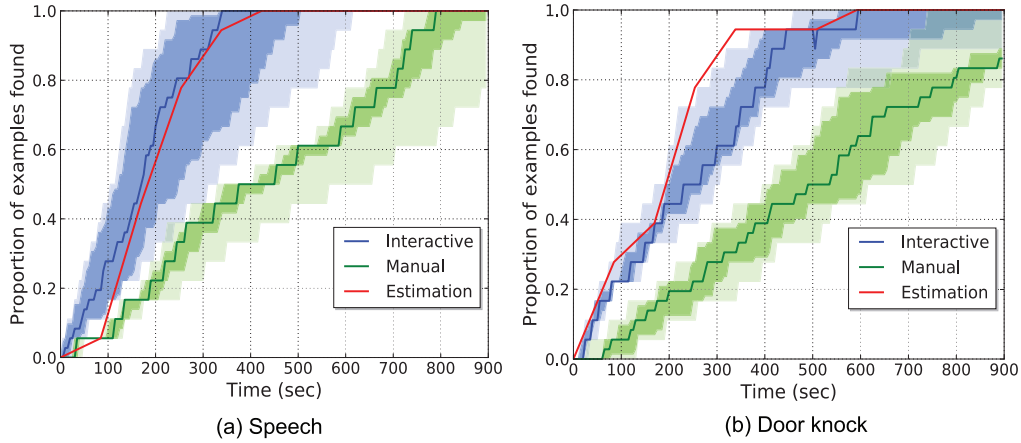

Fig. 13. The machine-estimated user performance overlayed on the actual user data from Figure 7(b) and (c).

4.68. We can apply this value to the simulation result by stretching the graph by 4.68 times on the x-axis. Figure 13 shows that applying this scaling factor makes the estimated graph very similar to the result from the actual user study.

6 DISCUSSION

In this section, we discuss limitations of the proposed system in terms of user interaction. As presented in Section 5.7, the interactive approach causes some amount of interaction overhead compared to a fully automated system or manual labeling. One piece of future work will be to redesign the interface to reduce the interaction overhead. We recognized two bottlenecks in the interaction of our tool by analyzing the log of users' behavior and their comments from the survey.

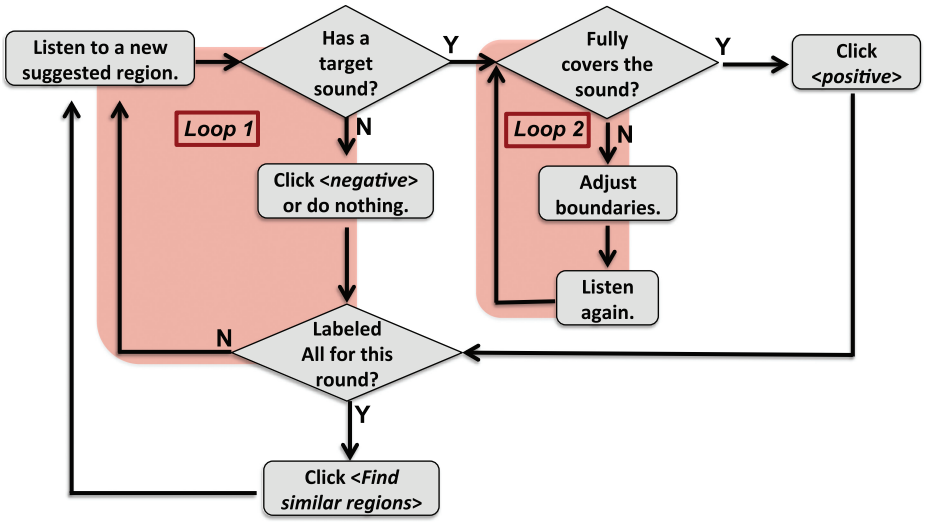

Fig. 14. There are two bottlenecks (Loop 1 and Loop 2) in the interaction flowchart of the interactive annotation.

Figure 14 highlights the two closed loops in the interaction flowchart presented in Section 4. In the experiment, users spent most of their time in these two loops during the annotation task.

Loop 1 is a process where a user verifies negative examples. In the case where the length of the initial query (i.e., a target sound example) is short and target sounds are sparse in the audio (e.g., 5% of the audio), the user loops the process many times to verify negative examples. In manual annotation, the process that verifies negative examples takes just as much time as it takes to listen to the audio. In our annotation tool, a user has to click the short regions to listen to them at most 5 times per round. In our experiment, it took an average of 15 rounds for participants to label all target sound events. Based on this analysis, one possible design choice for future work would be to let a user listen to all 5 suggested regions just by clicking a button once, instead of selecting each of selected regions to listen to them. It would help users verify negative regions more quickly.

In *Loop 2*, users keep adjusting boundaries of the suggested region while repeatedly listening to it. Some participants commented that even if a suggested region covers one instance of the target sound, they had to listen to the audio before and after the suggested region to understand the context and make sure that the start and end of the selected region are correctly positioned. In other words, verifying if there is no target sound around the suggested region causes one or more unnecessary listens. Based on this feedback, one solution to this issue for the next version of our tool would be to present a longer region to a user than the actual length of the initial target sound.

Implementing these changes on the two bottlenecks in the interaction loop, *Loop 1* and *Loop 2,* should reduce interaction overhead, allowing an even greater speedup of the annotation process.

7 CONCLUSION AND FUTURE WORK

We presented a human-in-the-loop interface for sound event detection and annotation that helps a user quickly label target sound events in a long audio file. The machine incrementally learns about the target sound event through user's relevance feedback. We performed a human-subject study to evaluate its effectiveness. The result showed that the proposed system lets users find sparsely distributed target sounds roughly twice as fast as manually labeling the target sounds. The survey

response and free-form comments showed that most participants were more satisfied with the interactive annotator than with the manual annotator.

We also presented methods to measure interaction overhead and machine accuracy of the proposed system. Since reducing the interaction overhead and increasing the machine accuracy are two primary ways to speed labeling further, it is important to measure them separately to evaluate the interface. We discussed the limitations of user interaction in our tool and suggested a future direction for the interface design.

In addition to the new interface design, there are a couple of ways to improve the proposed tool. First, we should be able to improve this speedup by exploring alternate feature representations, feature reweighting techniques, and search techniques. The nearest-neighbor searching and the Fisher-criterion-based feature reweighting used in this article assume that sound events of the same class are placed together in the feature space. If examples of the target class form multiple clusters apart from each other in the feature space, the nearest-neighbor search would show poor performance. Since the main focus of this work is not to improve machine accuracy, we did not explore other advanced techniques. However, speeding up labeling performance by applying advanced techniques is one area for future work. Second, to prevent users listening to unnecessary negative regions after all sound events are detected, it is very important to inform users when to stop labeling. Therefore, developing a systematic stopping criterion is another area for future work.

One other key factor that might affect the performance of users' labeling is visualization. For future work, we could explore alternate visualizations, not just waveforms. For example, the system could emphasize transient sounds or stationary noise in an audio sample in a way that general users can understand and allow them to get help to label sound events quickly. Finally, in this article, we did not focus on human error. The problem we are trying to solve is one where the human can easily label the audio if they have enough time, but have they so much audio to be scanned that they cannot complete the task. The tasks we chose were designed to be easy enough so that human labeling error was negligible (e.g., recognize human speech vs. non speech and door knocks vs. other sounds). Therefore the current system accepts all human feedback as valid. However, humans make mistakes in other circumstances. The implicit negative labeling occurring with users' erroneous behavior could lead to false-negative labels being fed into the system. Although the current interface does not have a perfect solution for this issue, it has a panel where users can quickly review regions they have labeled up to the current round, and they can fix it at any time. The system updates its model with the corrected labels every round. For future work, we will develop strategies to control or filter out human error.

REFERENCES

[1] Saleema Amershi, James Fogarty, Ashish Kapoor, and Desney Tan. 2010. Examining multiple potential models in end-user interactive concept learning. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'10)*. ACM, 1357–1360. DOI: http://dx.doi.org/10.1145/1753326.1753531

[2] Saleema Amershi, Bongshin Lee, Ashish Kapoor, Ratul Mahajan, and Blaine Christian. 2011. CueT: Human-guided fast and accurate network alarm triage. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'11)*. ACM, 157–166. DOI: http://dx.doi.org/10.1145/1978942.1978966

[3] Niels Bogaards, Axel Röbel, and Xavier Rodet. 2004. Sound analysis and processing with AudioSculpt 2. In *Proceedings of the International Computer Music Conference (ICMC)*. Michigan Publishing, 1–1.

[4] Niels Bogaards, Chunghsin Yeh, and Juan José Burred. 2008. Introducing ASAnnotation: A tool for sound analysis and annotation. In *Proceedings of the International Computer Music Conference (ICMC)*. Michigan Publishing, 1–1.

[5] C. G. v. d. Boogaart and R. Lienhart. 2006. Audio brush: A tool for computer-assisted smart audio editing. In *Proceedings of the 1st ACM Workshop on Audio and Music Computing Multimedia (AMCMM'06)*. ACM, 115–124. DOI: http://dx.doi.org/10.1145/1178723.1178741

[6] Nicholas J. Bryan, Gautham J. Mysore, and Ge Wang. 2014. ISSE: An interactive source separation editor. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'14)*. ACM, 257–266. DOI: http://dx.doi.org/10.1145/2556288.2557253

[7] Chris Cannam, Christian Landone, Mark B. Sandler, and Juan Pablo Bello. 2006. The sonic visualiser: A visualisation platform for semantic descriptors from musical signals. In *Proceedings of 7th International Conference on Music Information Retrieval, ISMIR 2006*. 324–327.

[8] Jean-Louis Durrieu and Jean-Philippe Thiran. 2012. Musical audio source separation based on user-selected F0 track. In *Proceedings of the 10th International Conference on Latent Variable Analysis and Signal Separation (LVA/ICA'12)*. Springer-Verlag, 438–445. DOI: http://dx.doi.org/10.1007/978-3-642-28551-6_54

[9] Rebecca Anne Fiebrink. 2011. *Real-time Human Interaction with Supervised Learning Algorithms for Music Composition and Performance*. Ph.D. Dissertation. Princeton, NJ, USA. Advisor(s) Cook, Perry R. AAI3445567.

[10] James Fogarty, Desney Tan, Ashish Kapoor, and Simon Winder. 2008. CueFlik: Interactive concept learning in image search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'08)*. ACM, 29–38. DOI: http://dx.doi.org/10.1145/1357054.1357061

[11] Ferdinand Fuhrmann, Martín Haro, and Perfecto Herrera. 2009. Scalability, generality and temporal aspects in automatic recognition of predominant musical instruments in polyphonic music. In *Proceedings of the 10th International Society for Music Information Retrieval Conference, ISMIR 2009*. ISMIR, 321–326.

[12] Giorgio Giacinto. 2007. A nearest-neighbor approach to relevance feedback in content based image retrieval. In *Proceedings of the 6th ACM International Conference on Image and Video Retrieval (CIVR'07)*. ACM, 456–463. DOI: http://dx.doi.org/10.1145/1282280.1282347

[13] Jorge M. A. Gomes, Teresa Chambel, and Thibault Langlois. 2013. SoundsLike: Movies soundtrack browsing and labeling based on relevance feedback and gamification. In *Proceedings of the 11th European Conference on Interactive TV and Video (EuroITV'13)*. ACM, 59–62. DOI: http://dx.doi.org/10.1145/2465958.2465979

[14] Sébastien Gulluni, Slim Essid, Olivier Buisson, and Gaël Richard. 2011. An interactive system for electro-acoustic music analysis. In *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011*. University of Miami, 145–150.

[15] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. Wilson. 2017. CNN architectures for large-scale audio classification. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 131–135. DOI: http://dx.doi.org/10.1109/ICASSP.2017.7952132

[16] Bongjun Kim and Bryan Pardo. 2017. I-SED: An interactive sound event detector. In *Proceedings of the 22Nd International Conference on Intelligent User Interfaces (IUI'17)*. ACM, 553–557. DOI: http://dx.doi.org/10.1145/3025171.3025231

[17] Rony Kubat, Philip DeCamp, Brandon Roy, and Deb Roy. 2007. Totalrecall: Visualization and semi-automatic annotation of very large audio-visual corpora. In *Proceedings of the 9th International Conference on Multimodal Interfaces*, Vol. 7. ACM, 208–215.

[18] Yizhar Lavner and Dima Ruinskiy. 2009. A decision-tree-based algorithm for speech/music classification and segmentation. *EURASIP Journal on Audio, Speech, and Music Processing* 2009, 1 (Dec. 2009), 178:1–178:14. DOI: http://dx.doi.org/10.1155/2009/239892

[19] Chang-Hsing Lee, Chin-Chuan Han, and Ching-Chien Chuang. 2008. Automatic classification of bird species from their sounds using two-dimensional cepstral coefficients. *IEEE Transactions on Audio, Speech, and Language Processing* 16, 8 (2008), 1541–1550.

[20] Beinan Li, John Ashley Burgoyne, and Ichiro Fujinaga. 2006. Extending audacity for audio annotation. In *Proceedings of 7th International Conference on Music Information Retrieval*. ISMIR, 379–380.

[21] David B. Orr, Herbert L. Friedman, and Jane C. C. Williams. 1965. Trainability of listening comprehension of speeded discourse. *Journal of Educational Psychology* 56, 3 (1965), 148.

[22] Alexey Ozerov, Cédric Févotte, Raphaël Blouet, and Jean-Louis Durrieu. 2011. Multichannel nonnegative tensor factorization with structured constraints for user-guided audio source separation. In *Proceedings of the 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 257–260. DOI: http://dx.doi.org/10.1109/ICASSP.2011.5946389

[23] Giambattista Parascandolo, Heikki Huttunen, and Tuomas Virtanen. 2016. Recurrent neural networks for polyphonic sound event detection in real life recordings. In *Proceedings of the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 6440–6444. DOI: http://dx.doi.org/10.1109/ICASSP.2016.7472917

[24] Geoffroy Peeters. 2004. A large set of audio features for sound description (similarity and classification) in the CUIDADO project. *Technical Report, IRCAM* (2004).

[25] Jose Portelo, Miguel Bugalho, Isabel Trancoso, Joao Neto, Alberto Abad, and Antonio Serralheiro. 2009. Non-speech audio event detection. In *Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 1973–1976. DOI: http://dx.doi.org/10.1109/ICASSP.2009.4959998

[26] Douglas A. Reynolds, Thomas F. Quatieri, and Robert B. Dunn. 2000. Speaker verification using adapted Gaussian mixture models. *Digital Signal Processing* 10, 1 (2000), 19–41.
[27] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. 2004. "GrabCut": Interactive foreground extraction using iterated graph cuts. In *ACM SIGGRAPH 2004 Papers (SIGGRAPH'04)*. ACM, New York, NY, USA, 309–314. https://doi.org/10.1145/1186562.1015720
[28] Olga Russakovsky, Li-Jia Li, and Li Fei-Fei. 2015. Best of both worlds: Human-machine collaboration for object annotation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2121–2131.
[29] Burr Settles. 2011. Closing the loop: Fast, interactive semi-supervised annotation with queries on features and instances. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1467–1478.
[30] Dan Stowell, Dimitrios Giannoulis, Emmanouil Benetos, Mathieu Lagrange, and Mark D. Plumbley. 2015. Detection and classification of acoustic scenes and events. *IEEE Transactions on Multimedia* 17, 10 (2015), 1733–1746.
[31] Justin Talbot, Bongshin Lee, Ashish Kapoor, and Desney S. Tan. 2009. EnsembleMatrix: Interactive visualization to support machine learning with multiple classifiers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'09)*. ACM, 1283–1292. DOI: http://dx.doi.org/10.1145/1518701.1518895
[32] Bart Thomee and Michael S. Lew. 2012. Interactive search in image retrieval: A survey. *International Journal of Multimedia Information Retrieval* 1, 2 (2012), 71–86.
[33] Giuseppe Valenzise, Luigi Gerosa, Marco Tagliasacchi, Fabio Antonacci, and Augusto Sarti. 2007. Scream and gunshot detection and localization for audio-surveillance systems. In *Proceedings of the 2007 IEEE Conference on Advanced Video and Signal Based Surveillance*. IEEE, 21–26. DOI: http://dx.doi.org/10.1109/AVSS.2007.4425280
[34] Sudheendra Vijayanarasimhan and Kristen Grauman. 2009. Multi-level active prediction of useful image annotations for recognition. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 1705–1712.
[35] Sudheendra Vijayanarasimhan and Kristen Grauman. 2014. Large-scale live active learning: Training object detectors with crawled data and crowds. *International Journal of Computer Vision* 108, 1–2 (2014), 97–114.
[36] Luis Von Ahn and Laura Dabbish. 2004. Labeling images with a computer game. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 319–326. DOI: http://dx.doi.org/10.1145/985692.985733
[37] L. Vuegena, B. Van Den Broeckb, P. Karsmakersb, J. F. Gemmekea, B. Vanrumsteb, and others. 2013. An MFCC-GMM approach for event detection and classification. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 1–3.
[38] Xiang-Yang Wang, Bei-Bei Zhang, and Hong-Ying Yang. 2013. Active SVM-based relevance feedback using multiple classifiers ensemble and features reweighting. *Engineering Applications of Artificial Intelligence* 26, 1 (2013), 368–381.
[39] Emin Wu and Aidong Zhang. 2002. A feature re-weighting approach for relevance feedback in image retrieval. In *Proceedings of the International Conference on Image Processing*, Vol. 2. IEEE, II–581.
[40] Dongxin Xu, Umit Yapanel, and Sharmi Gray. 2009. *Reliability of the LENATM Language Environment Analysis System in Young Children's Natural Home Environment*. Technical Report.
[41] Seid Muhie Yimam, Chris Biemann, Ljiljana Majnaric, Šefket Šabanović, and Andreas Holzinger. 2015. Interactive and iterative annotation for biomedical entity recognition. In *Proceedings of the International Conference on Brain Informatics and Health*. Springer, 347–357. DOI: http://dx.doi.org/10.1007/978-3-319-23344-4_34

Received December 2016; revised October 2017; accepted December 2017