Respeak: A Voice-based, Crowd-powered, and Accessible Speech Transcription System

Pooja Sethi
Supervised by Richard Anderson

A senior thesis submitted in partial fulfillment
of the requirements for the degree of

Bachelor of Science
With Departmental Honors
Paul G. Allen School of Computer Science & Engineering
University of Washington
June 1, 2017

Presentation of work given on May 2, 2017

Thesis and presentation approved by _____

Date _____

Abstract

Respeak is a voice-based, crowd-powered, and accessible speech transcription system designed for users in the developing world. It capitalizes speaking ability rather than typing ability to transcribe audio files. Respeak has two main components: an Android application that users can download on their mobile device to complete voice-based transcription tasks, and a backend for merging crowdworkers' individual transcripts together to produce a final transcript. In its first deployment with 25 low-income, non-blind Indian college students, Respeak produced transcriptions with a Word Error Rate (WER) of 15.2% on Indian English audio files and 8.6% on Hindi audio files. In comparison, a state-of-the-art Automatic Speech Recognition (ASR) system (the Google Cloud Speech API) yielded a transcript with overall WER of 50% on the same Indian English audio and 54% on Hindi audio – a performance that Respeak beat over three times for Indian English and over six times for Hindi. The expected payout participants received for an hour of their time was 76 INR (USD 1.16), one-fourth of the average daily wage rate in India. In a second, small scale usability study, we enhanced the Respeak user interface to be more accessible to blind users and interviewed 4 blind participants local to Seattle, WA. The participants gave Respeak an average accessibility rating of 8.25 out of 10. Blind users reported enhanced earning potential as well as instrumental benefits, such as greater self-esteem, from using Respeak.

Contents

1 Introduction

Microtasking platforms, also known as crowdsourcing platforms, have potential to benefit low-income people in the developing world. Such platforms allow workers to earn money for completing microtasks without the need for formal employment contracts, physical presence in a particular location, and prior established relationships with employers [1]. Completing microtasks usually does not require a college education or a specialized skill set; for example, a microtask may ask a worker to provide a caption for an image or to listen to an audio file and transcribe it. Despite these advantages, there are several major barriers-to-entry that prevent low-income people in the developing world from widely participating in crowdsourcing platforms. Platforms such as Amazon's Mechanical Turk (MTurk) [15], CastingWords [17], and Samasource [25] have a limited selection of tasks available in languages spoken in developing countries, such as Hindi. These platforms require that workers have access to an Internet-connected computer. To receive compensation, workers must also have an active PayPal account that is connected to a banking institution. In India alone, 47% of the 15+ year population does not have a bank account [6] and 97% of households do not have access to a computer connected to the Internet [2]. Khanna et al. found that less than 3% of India-based Turkers are low-income workers making less than 1700 USD (75000 INR) per year, even though this demographic makes up 13% of India's labor force [1].

Finding work is also especially difficult for visually impaired people. In the United States, only 40.4% of working age visually impaired adults in the United States were employed [16]. Factors including unconscious biases exhibited by hiring managers, lack of transportation, and inaccessible workplace environments all contribute to the low employment rate. Problems in finding employment are only exacerbated for blind people in the developing world. 90% of the world's blind population lives in a low-income setting [7]. The majority of these people live in India [7], and face social stigmas and infrastructure challenges that prevent them from finding work. Again, crowdsourcing platforms have an opportunity to help low-income blind people in the developing world find employment. However, on top of all of the barriers-to-entry in the developing world previously described, blind workers are met with an additional set of constraints: existing crowdsourcing platforms like MTurk have inaccessible user interfaces, which we discuss in detail in section 2.

Designing appropriate microtasks and crowdsourcing platforms that enable users in the developing world, including blind users, to participate is an important and understudied challenge. We found that voice-based microtasks naturally lend themselves well to an accessible crowdsourcing system, as they can be done simply by using listening and speaking skills without any visual input [29]. Our crowdsourcing system, Respeak, allows users to complete speech transcription tasks entirely using their voice and in local languages such as Hindi. We chose to support speech transcription specifically since it is a highly demanded service; transcriptions are demanded for a wide variety of audio content, ranging from public speeches to university lectures, movies, music, television shows, news reports, phone calls, and interviews. Speech transcription fuels a multi-billion dollar industry, and transcription costs in the medical industry alone are projected to reach USD 60 billion globally by 2019 [11].

In addition to being an entirely voice-based system, Respeak allows user to complete transcription tasks from an Android smartphone rather than a computer. A Pew study found that the median smartphone ownership rate in developing and emerging countries was 37% in 2015, rising by over 15% from two years prior [31]. Finally, Respeak provides users with a reward of mobile airtime, eliminating the requirement of having a bank account.

Users are rewarded based on how accurately they complete transcription tasks.

The technical contribution of Respeak is a multistage speech transcription system that combines the benefits of crowdsourcing and ASR to produce transcripts for audio in languages and accents spoken in the developing world, specifically for Hindi and Indian English. From a crowd worker's perspective, Respeak is an Android smartphone application that allows the worker to complete transcription tasks. A Respeak task asks a worker to listen to a short audio segment of three to six seconds in length and repeat what they heard into their smartphone in a quiet environment. Using Android's built-in ASR system, a transcript for the segment is generated in real-time locally on the worker's phone. The worker is asked to submit the transcript once they are satisfied with it. For each segment, Respeak's backend merges the transcripts produced by multiple users with multiple string alignment and majority voting to a produce a best estimate transcript of the audio. Finally, transcripts from the many short segments are concatenated together in order to produce the final output transcript.

In the first half of this paper, we discuss the Respeak's technical design and the results of its first deployment with 25 non-blind college students in India. In the latter half, we discuss our recent accessibility changes to Respeak and the insights we gained from conducting a usability study with four blind participants local to Seattle, WA.

2 Background and Related Work

There are currently several types of solutions available on the market for doing speech transcription. The first type is manual transcription, where human workers listen to an audio file and then type what they hear. The second type is Automatic Speech Recognition (ASR), which relies on software to recognize and transcribe audio. However, each of these solutions have critical limitations. In the case of manual transcription, the cost can be prohibitive, the turnaround times tend to be high, and there is limited to no support for transcribing audio files in languages spoken in developing countries or localized accents of well-represented languages. Crowd-powered businesses such as CastingWords [17] and SpeechPad [27], for example, typically charge USD 1 - 6 per minute of audio; for the cheapest pricing their turnaround time is at least one week. Moreover, they support only well-represented languages such as English, French, and Spanish.

Other manual transcription businesses that do not rely on crowd workers but instead have a hired fleet of transcribers, such as Quick Transcription [24] and Scripts Complete [26], have a greater range of language support but charge starting at USD 5 per minute. In the case of ASR systems, such as Nuance Dragon [23] and Google Cloud Speech API [18], support is available for some local languages and accents and the cost is much less prohibitive; Google Cloud Speech API charges USD 0.024 per minute [29]. Unfortunately, the quality of plain ASR is quite poor for audio in languages spoken in developing countries. Our experiments showed Google Cloud Speech API had a Word Error Rate (WER) of 50% on Indian English audio and 54% on Hindi audio [29]. This suggests that ASR on its own is not enough to transcribe audio in such languages and necessitates using human skills. Prior work has looked at ways that humans can work in conjunction with ASR systems to produce more accurate transcripts at lower cost. Several researchers have created tools that allow crowd workers to edit and correct transcripts generated by ASR. For example, Parent and Eskenazi [30] as well as Lee and Glass [3] designed a two-phased approach to reduce the cognitive load on crowd workers doing transcription. Such systems inspired our design

of Respeak.

Other crowdsourcing platforms, such as Samasource [25] and MobileWorks [4], are designed for workers in developing countries; however, they still require acceptable typing skills in English or a local language. Speech transcription platforms that hire crowd workers, like Speechpad [27], also require a minimum typing speed of 40 WPM. In past experiments, we found that the typing speed of Indian college students is only around 29.5 WPM [29]. Existing crowdsourcing platforms are also ill-equipped to support the needs of visually impaired workers. Zyskowski et al. [10] evaluated MTurk's accessibility by interviewing disabled adults, including visually impaired people, and found several critical issues. For example, they found to create an MTurk account, one must get past a CAPTCHA. Though one user they interviewed was able to call Amazon to get around the issue, the extra step just to sign-up could be prohibitive to others. Another drawback is that some microtasks are timed. If a user does not finish a microtask within the allotted time, they receive no compensation. Spending a lot of time and energy in a task, and then receiving no compensation because it timed out, was discouraging and prevented some users from wanting to do crowd work. In addition, microtasks on MTurk can redirect to external web pages, which may not have proper accessibility support. Finally, users can get negative reputation points for not being able to finish a task in time, even if inaccessibility issues rather than any fault of their own are preventing them from completing it.

Prior work has primarily focused on using crowsourcing to create human-powered access technologies. For example, Brady and Bigham [8] do a case study on Scribe, which calls on non-expert typists to create real-time video captions for deaf users. RegionSpeak [9] calls on Amazon Mechanical Turk workers to answer visual questions for blind users. However, to our knowledge little work has been done to enable visually impaired users to participate as workers in crowdsourcing platforms.

3 Respeak Design and Implementation

Respeak is a multistage speech transcription system. We begin by providing a high-level overview of how the system works, then describe each component in detail.

3.1 Process Overview

There are five main steps that the Respeak speech transcription system takes to go from an audio file to a transcript, as illustrated in Figure 1.

1. **Audio Segmentation:** An audio file to be transcribed is received on the server-side. A segmentation algorithm determines where natural pauses occur, such as when the speaker has finished a word or sentence. The audio file is then broken into small segments at the natural pause boundaries. The segments are purposely short so that they are easy for users to remember.

2. **Distribution to Crowd Workers:** Every audio segment is randomly distributed to multiple Respeak users. A user opens up the Respeak app on their Android smartphone. In the app, the user will see that she has been assigned a new task. A task corresponds to a single audio segment. The user is asked to listen to the audio segment.

3. **Transcription by Crowd Workers using ASR:** Once the crowd worker feels confident that she remembers the segment, she records herself "respeaking" what she just

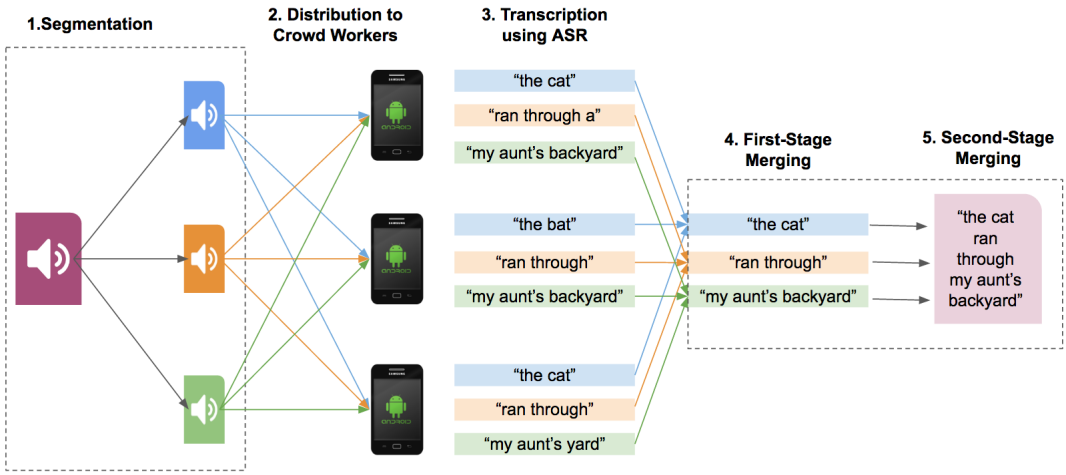

Figure 1: A high-level illustration of the five main steps in the Respeak process showing how an audio file is transcribed. Areas inside dotted lines represent processes that happen in Respeak's server-side backend. Note that a user only receives one Respeak task at a time, though we have shown multiple tasks here.

heard using the Respeak app. Behind the scenes, the Android SpeechRecognizer API [28] converts her voice into text. Once the user is done speaking, a preliminary transcript for the segment is displayed on the app. If the user feels that the transcript is close to what she heard, she can submit it to receive a new Respeak task (i.e. a new audio segment to listen to). If the user feels that there are too many errors in the transcript, she can try respeaking again. At this stage, it is expected that the WER will be relatively high as the user may be trying to minimize the amount of time she has to spend on a single task, and also may not fully understand everything being said in the audio.

4. **First-Stage Merging:** Once a predefined threshold of users have submitted their individual transcripts for a segment, we apply Multiple String Alignment (MSA) and majority voting to merge the transcripts into a single best-estimate transcript. If the errors from the ASR system are randomly distributed, this reduces the overall WER since the correct word is usually recognized for the majority of the users. To reward the user, we compare her individual transcript to the best-estimate transcript. Her accuracy rate in comparison to the best estimate determines how high her reward will be. Once a user reaches a cumulative reward amount of INR 10 across all tasks she has completed, she is sent a mobile airtime reward of INR 10.

5. **Second-Stage Merging:** Finally, second-stage merging takes all of the output transcripts from first-stage merging and concatenates them in order into one large file. This is the final transcript for the original audio file.

3.2 Audio Segmentation

The first step in the Respeak system is dividing the audio files to be transcribed into short segments that are easy to remember. This segmentation is done on the server-side before distributing tasks to Respeak users. There were several parameters we had to keep in mind when designing the segmentation algorithm in order to reduce cognitive load on users – primarily, what length of segment is easiest to remember, whether segments should be divided on natural pauses or on arbitrary breaks, and what order the segments should be presented in. Through cognitive studies [29] we found that audio segments should not exceed 5 seconds, as WER becomes significantly higher when segments are past this length. If a given segment exceeds a threshold of 5 seconds, we recursively divide it further until it reaches an acceptable length. In addition, we found that audio files should be split on natural pauses. Clipped segments that begin or end in the middle of a word are distracting to most users, making it difficult for them to remember the content. Finally, we found that content understanding and retention is higher when segments are played sequentially, and therefore segments should be presented in this way when possible.

3.3 Android Application

Respeak crowd workers perform tasks using an Android application. We decided to build Respeak for Android, rather than for other mobile operating systems such as iOS and Windows, primarily because Android has the largest market share worldwide and is the most popular operating system for low-end smartphones used in the developing world.

Android applications are built out of four key components: `Activity`, `Fragment`, `BroadcastReceiver`, and `ContentProvider` classes. An `Activty` corresponds to a screen of the application where the user can take some action. Respeak consists of six activities: `MainActivity`, `PayScaleActivity`, `HistoryActivity`, `LaunchActivity`, `LoginActivity`, and `SettingsActivity`. It also has two utility classes containing implementation details for the audio player and timer.

3.3.1 Respeak Tasks Home

The Respeak Tasks Home page, corresponding to the `MainActivity`, is the go-to screen for crowd workers to complete Respeak tasks. The user interface is designed to provide workers with intuitive queues that guide them through task completion. The workflow for completing tasks is shown in Figure 2.

When the user opens up the app, she is shown a screen that asks her to play and listen to an audio recording. After she presses the play button and listens to the audio, the record section appears. The user can then press the record button, bringing up a dialog asking her to respeak what she listened to. The dialog automatically exits when the user is done respeaking and displays the output text back to her. If she is satisfied with the output, she can submit the transcript. This completes the current Respeak task, and a new task is fetched from the server so the process can repeat. Every Respeak task also has a corresponding reward value. This is the maximum amount that the user can earn by submitting this task. In the screenshots in Figure 2, the reward is shown in INR since the first deployment of the app was in India. A user may not receive the entire reward, however, if the WER of the transcript produced is high in comparison to transcripts produced by other crowd workers. In the case where the user is not satisfied with the ASR output, she can record herself respeaking again. She is also free to play the audio again at any

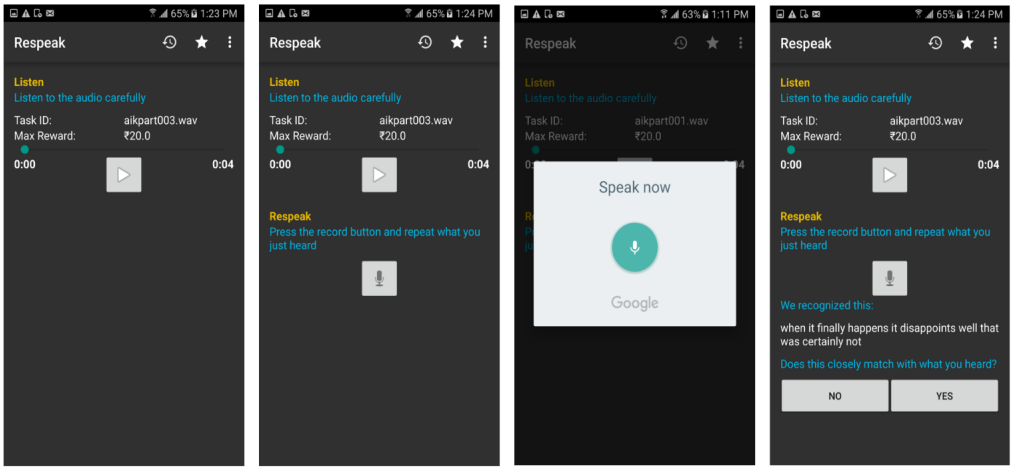

Figure 2: The Respeak workflow on the home page is shown from left to right as the user progresses through the instructions.

point through out the entire process. The sequence of all possible actions for completing a Respeak task can be modelled as a finite state machine as shown in Figure 3.

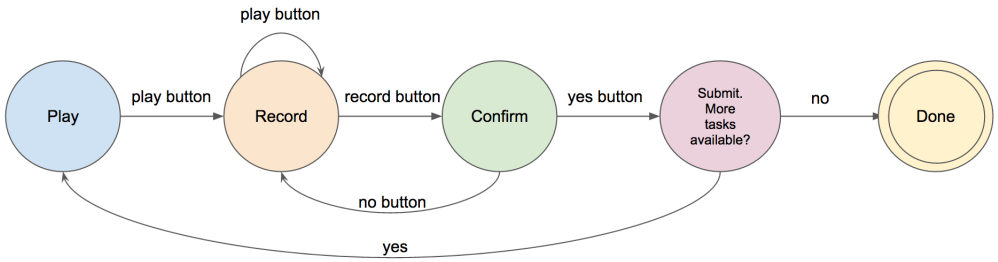

Figure 3: The sequence of states and transitions that the user follows while completing a task on the Respeak home screen.

Transitions in the state machine are represented in the Android app by setting listeners on user events, such as pressing the play, pause, record, and submission buttons or dragging the seek bar. We also set listeners on the `AudioPlayer` that keep track of when the audio progress reaches 100% (i.e. the audio file is done playing). The states correspond to different `View`s that are shown to the user. More pieces of the `View` are displayed as the user progresses forward in the state machine, and they are hidden if the user takes a transition going back to a previous state.

Because the UI of the `MainActivity` involves many moving components, we had to carefully design how data was sent and fetched from the server in order to prevent the UI Thread from becoming irresponsive. We managed this by creating three separate `AsyncTask`s: `FetchNewTask`, `DownloadAudioTask` and `SendOutputTask`. `FetchNewTask` runs when the Respeak user opens the app for the first time or has just completed a task and fetches the

task ID to give to the particular user. On post execution, it then calls `DownloadAudioTask` with the given task ID. `DownloadAudioTask` fetches and caches the audio segment to play in the application's local storage on their device. This is helpful because when the user closes the app when they have not completed the current task, it is still available to them without the need to download it again. When the task is completed, the cache is cleared to prevent the app from hogging resources on the user's phone. Finally, the `SendOutputTask` is executed when the user submits the current task. This sends the user's transcript back to the server. It also sends back analytics about how the user interacts with the app and metadata about the transcript, such as the ASR confidence scores and the amount of time in seconds the user spent on the task.

Respeak is also currently capable of doing translation tasks in multiple languages. Though we currently provide support for United States English, Indian English, and Hindi, it can be easily extended for other languages as well. A bilingual user can also receive tasks in multiple languages. For example, a crowd worker in India can perform Respeak tasks in both Indian English and Hindi, giving her a greater range of earning opportunities.

3.3.2 Speech Recognition

Automatic Speech Recognition is done on the client-side using Android's SpeechRecognizer API [28], available on Android API level 8 and higher. Before deciding to use the built-in SpeechRecognizer API, we also considered using an ASR software such as Nuance Dragon [23], which charges a monthly license fee. The biggest advantage of using the SpeechRecognizer API is that it does not require making any server-side calls, which can be expensive and slow especially in low-resource environments. Nuance Dragon, on the other hand, would have to be set up on the server-side. SpeechRecognizer also has support for a wide variety of languages and accents, including Hindi and Indian English. Another key advantage of SpeechRecognizer is that it does not require any pre-training on the user's voice. This reduces the time barrier between the Respeak user downloading the app and actually being able to perform tasks. SpeechRecognizer not only has an engineering advantage over Nuance Dragaon but also a cost advantage. While using the built-in Android API is free-of-charge, Nuance Dragon software starts at several hundred dollars per license.

3.3.3 Payment Scale

Every Respeak task has a maximum possible monetary reward value associated with it. Respeak crowd workers are paid by how accurately they can complete transcription tasks. The higher their accuracy rate, the higher the percentage of the maximum reward workers will receive. There are three payment grades, as shown in Figure 4. This corresponds to the `PaymentScaleActivity` in the app. The user can view the Payment Scale at any time by selecting the corresponding `ActionBar` icon in the upper right corner of the Respeak home screen.

The crowd worker's accuracy rate for a segment is calculated by comparing the transcript she produced to the best-estimate transcript, produced by the collective group of crowd workers that also performed the same task. If her transcript's accuracy is greater than or equal to 80% in comparison to the best-estimate, the worker receives 100% of the reward value for the task. Else, if her accuracy is greater than or equal to 50%, she only receives 50% of the listed reward value for the task. Submissions below an accuracy of 50% receive no payment at all.

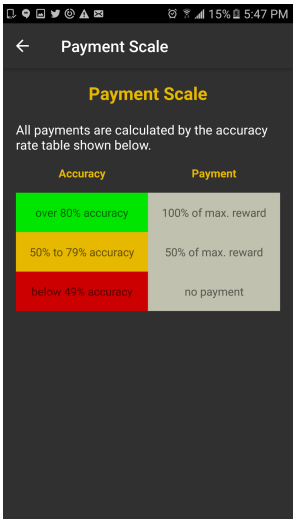

Figure 4: The Payment Scale that a Respeak user can check at any time to find information about the reward vs. the accuracy.

This reward structure is designed to incentivize users to complete Respeak tasks as accurately as they possibly can. This means listening to the original audio carefully and re-playing and re-recording when necessary before making their final submission. However, the user is not heavily penalized unless they make a relatively large number of mistakes.

We also wanted to control the cost of transcription so that it was below USD 1 per minute. The maximum reward value for every task was calculated by multiplying the length of the segment with 0.2 INR. We calculated that at most, if 5 workers complete a single Respeak task each with an accuracy greater than 80%, the cost of transcription would be USD 0.92 per minute.

3.3.4 Task History

Respeak users can also view their history of completed tasks under the Task History page, or the `HistoryActivity`. The history page allows the user to see aggregate information about the tasks that they've completed, including the date that their last mobile-top up was awarded, the total amount that they've earned to-date, and their overall accuracy rate. They can also scroll through a list that provides details about every single task that they've completed, including the maximum reward, their accuracy rate, and the amount they earned for the individual task.

This information is computed on the server-side and fetched asynchronously using the `FetchHistoryTask`. When the `HistoryActivity` is stopped, the data is cached in `SharedPreferences` to be restored when the activity is resumed or to be used as a fallback when there is no internet connection. If the activity is destroyed (such as when the user closes the app and re-opens it) the latest data is fetched from the server. When the user starts performing tasks, she might see that they are marked as "Under Review" in the Task History. This is because a certain threshold of other users also needs to perform the same Respeak task before the best-estimate transcript and user accuracies are computed on the

server-side. An example of the Task History screen is shown in Figure 5.

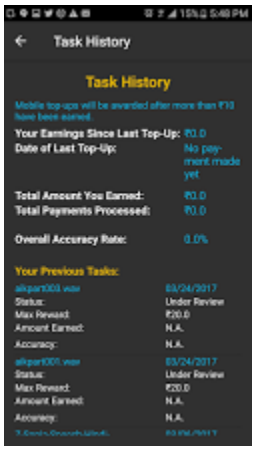

Figure 5: The Task History displays aggregate as well as individual information about previously completed tasks.

3.4 Server-Side Backend

The Respeak Android application receives audio data and sends back completed transcripts by communicating with the backend, which is a Google App Engine app. The backend's API endpoints are exposed using the Cloud Endpoints Framework [12]. Data such as a user's previous task history is stored in a Cloud SQL [20] database that is queried when the Respeak application makes an API call. Other information that does not need to be persisted, like the user's average accuracy rate, is calculated on-the-fly. The database schema consists of three tables: `task_details`, which contains information about particular tasks such as their ID, max reward, and language; `user_details`, which contains information about users such as their ID and analytics metadata (e.g. how many times a user replayed an audio file); and finally `task_assignment`, which contains information about which tasks are assigned to which users, how much users earned for a particular task, etc. Transcript merging is also done on the server-side, but does not need to be exposed via an API to clients since users should never need to see the final output transcript.

3.5 Transcript Merging

Once a threshold of Respeak users have submitted their transcripts for a single segment, first-stage merging is done to create the best-estimate transcript for that segment. Merging is done to reduce the WER under the assumption that the errors made by Respeak users are randomly distributed. Our merging algorithm is adapted from the implementation presented by Naim et al. [5], which uses weighted A* search and MSA to combine partial captions. It aligns segments word-by-word rather than character-by-character. The second-stage merging step simply takes the outputs of first-stage merging and concatenates them together in the correct order to create a final transcript of the original audio file.

4 Field Deployment in India

Our first deployment of Respeak was with 25 university students in Mumbai, India. For brevity, we provide a summary of our experiment methodology and findings. For the full analysis, please refer to our previous paper [29]. We particularly chose to do our deployment with university students as they owned smartphones connected to the internet and also had financial constraints that motivated them to try using Respeak in exchange for mobile airtime. After an initial face-to-face training session where we helped the students install Respeak on their personal Android phones, we allowed them to use Respeak on their own for a period of one month. To evaluate our findings at the end of the month-long period, we used a mixed-methods approach spanning quantitative analyses of performance, cost, and turnaround time, and qualitative interviews. By combining the transcripts of five randomly selected users, we found that Respeak transcribed audio content in Hindi and Indian English with a WER of 8.6% and 15.2%, respectively. The WER for Hindi tasks was much lower than that for English tasks because of users' better listening and speaking skills in Hindi. The cost of speech transcription was USD 0.83 per minute, and the turnaround time was 39.8 hours, substantially less than the industry standard of USD 5 per minute for Hindi and Indian English transcription. The expected payout for an hour of their time was 76 INR (USD 1.16), one-fourth of the average daily wage rate in India [21].

5 Accessible Respeak User Interface Changes

Making the Respeak Android application accessible for blind users required redesigning components of the user interface. Blind users can navigate Android apps using a free screen-reading software called TalkBack [19]. TalkBack comes pre-installed on Android phones and allows users to explore items on the screen by touch or by using swiping gestures, known as linear navigation. The Android Developer documentation [13] provides guidelines on how to leverage accessibility features of the Android platform, such as TalkBack, and suggests design principles that can be used in order to create a better user experience. Some of the accessibility changes we implemented are shown in a side-by-side comparison in Figure 6. These changes are described in more detail in the following sections.

5.1 Labeled UI Elements

One of the simplest and most immediate ways that we were able to make Respeak more accessible was by adding labels known as `android:contentDescription` to UI elements that TalkBack can then read aloud. An example of a content description label is shown in Figure 7.

Without content descriptions, TalkBack would simply make generic announcements such as "image button" if the user pressed on a UI element such as an `ImageButton`, which provides no information about its actual functionality.

5.2 Large Touch Targets

It is difficult for blind users to navigate small touch targets on a device's screen, so it is important to make them large whenever possible. Android's guidelines suggest that touchable areas be a minimum of 48dp x 48dp. As shown in Figure 6, we minimized empty space and increased button and font sizes.

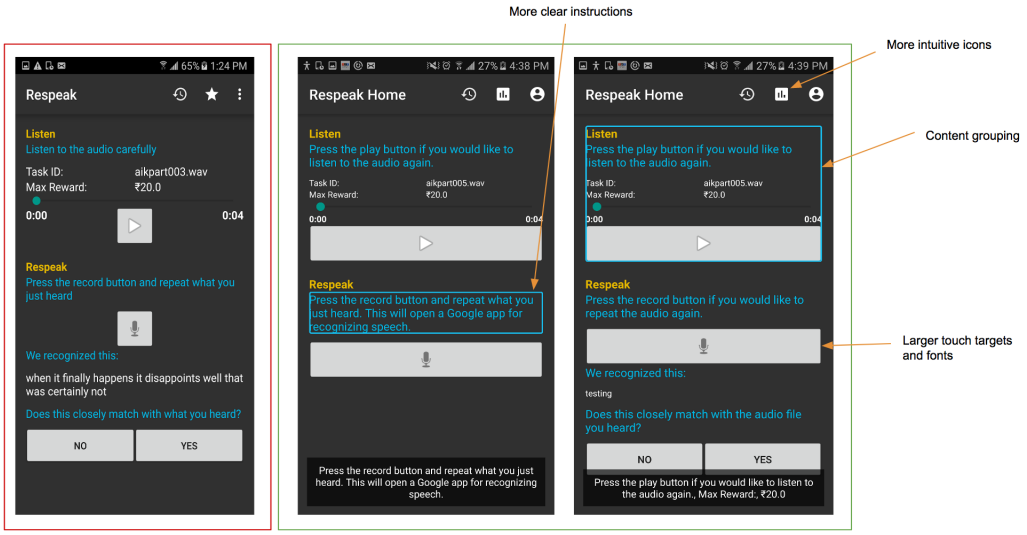

Figure 6: The screenshot on the left inside a red box shows the original Respeak home page before we improved its UI to be accessible for blind users. The screenshots on the right, shown inside a green box, show the Respeak home page after making accessibility improvements. Buttons are bigger, there are improved instructions and icons, and there is content grouping. The bottom of the screen contains a caption that is being read out loud using TalkBack when the user presses on the focused touch target, shown inside a blue box.

```
<ImageButton
    android:id="@+id/recordButton"
    android:contentDescription="Record"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:src="@android:drawable/ic_btn_speak_now" />
```

Figure 7: An example of a content description label for the record button.

5.3 Explicit Instructions

We found that simple verbal instructions that are good enough for sighted users may not be clear for blind users, who can't rely on visual cues that sighted users can take for granted. For example, we found that we had to clarify the instructions for recording audio on the Respeak home screen. Initially, the verbal instructions were just *"Press the record button and repeat what you just heard."*. In the accessible version of Respeak, it now reads, *"Press the record button and repeat what you just heard. This will open a Google app for recognizing speech."*. While sighted users can see a separate Google app opening up (as in the third screenshot of Figure 2) and don't need to be explicitly told to expect it, blind users cannot. It is helpful to let them know that a separate app will pop up. Once the user is done respeaking, the recording dialog automatically closes and makes a soft beeping noise to indicate that it is done. The recording instructions dynamically changes to *"Press the record button if you would like to repeat the audio again."* This is useful for providing users with a clear indication that they are no longer in recording mode, although they can try

respeaking again if they desire. These instruction changes can be seen in Figure 6.

5.4 Content Grouping

The most important interface change we made was grouping content into single announcements. Content grouping allows you to treat multiple UI elements as a single, focusable container. As long as the user presses any single element within the group, the entire content of the container is announced out loud by TalkBack.

We found content grouping particularly important for making more complicated screens easier to navigate. We provide an example of this in the Task History screen in Figure 8. The Task History screen is particularly difficult to navigate because it contains many individual elements. Without grouping, a blind user would have to individually touch text labels or swipe many times in order to read all the elements on the screen. Grouping makes this less cumbersome, as logically related elements can be accessed all-at-once.

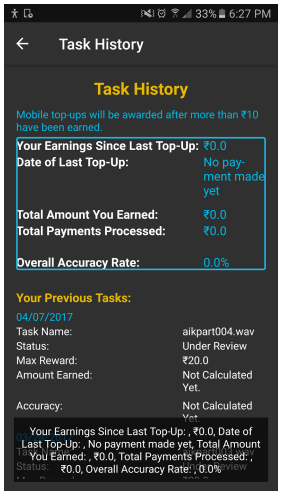

Figure 8: The Task History's page complex interface especially benefited from grouping content. The focused blue box highlights everything that is read as a single announcement.

6 Usability Study Design

The Android Developer documentation [13] suggests that developers test the accessibility of their applications. Though developers can manually test their apps themselves, conducting user tests allows them to gain insights they may have otherwise overlooked. The documentation encourages developers "to reach out to local organizations, colleges, or universities that provide training for people with disabilities" in order to find people who can give feedback. We thus designed a usability study with four blind participants local to Seattle primarily to get qualitative feedback on how we could improve the interface of Respeak. We were able to contact the participants via snowball sampling by reaching out to a student at the University of Washington and her network.

6.1 Methodology

We conducted usability studies one-on-one with each participant in 45-minute to 1-hour long sessions. We provided the participant with a Nexus 5 Android phone with Respeak pre-installed for the duration of the session. We seeded the Respeak backend with audio segments from a comedy skit, in United States English.

At the beginning of the session, we read a brief passage to the participant explaining the overall purpose of Respeak. We then asked the participant to perform a series of tasks. Our instructions were purposely vague and open-ended to encourage the participants to explore the app, as if they were using Respeak on their own in the real world. The tasks we asked participants to complete are given below.

1. Try to complete the Respeak task on your own for the first trial.

   (Optional: We only provided hints if the participant was stuck.)

   (a) Play the audio file. Make sure to listen to it carefully.

   (b) Press the record button and then repeat what you just heard. You can also play the audio again if you would like before recording.

   (c) Find the output text that appears after you finish recording. Verify that the output of the audio file matches what you expect it to be.

   (d) Find the question that asks "Does this closely match with the audio file you heard?" and press the "No" button.

   (e) Re-record yourself repeating the audio file.

   (f) Verify that the output of the audio file matches what you expect it to be.

   (g) Find the question that asks "Does this closely match with the audio file you heard?" and press the "Yes" button.

2. Now, try to complete another task.

3. What is the reward for the current task?

4. For an accuracy over 80%, what percentage of the payment do you get?

5. What is your overall accuracy rate?

6. What is the maximum reward of the last task that you completed?

Throughout the session, we observed the participant's behavior and encouraged them to ask questions if they got stuck or to share observations as they were using the application. Once the participant finished the above tasks, we asked them quantitative and qualitative questions about their experience using Respeak.

We used a modified, 10-point NASA-TLX scale [22] to quantify the perceived workload our tasks put on a user. The scale measures workload across five factors: mental demand, physical demand, overall performance, effort required to complete the task, and frustration. We asked each participant to rate the Respeak Android app across these factors on a scale of 1 to 10, with 1 being very low and 10 being very high. The questions we asked to measure each factor are given below:

- **Mental Demand** How mentally demanding were the tasks?

- **Physical Demand** How physically demanding were the tasks?
- **Overall Performance** How successful were you in accomplishing what you were asked to do?
- **Effort Required** How hard did you have to work to accomplish your level of performance?
- **Frustration** How insecure, discouraged, irritated, stressed, and annoyed were you?

.

At the end of the session, we also conducted a semi-structured interview and asked the participant to rate the Respeak app on its accessibility, provide feedback as to what they liked and didn't like about the app, and to provide general thoughts on what they thought of the Respeak system as a whole. All of the interviews were recorded, transcribed, and analyzed using open coding.

6.2 Participant Demographics

In total, we interviewed one woman and three men who lived in Seattle. Their ages ranged from late 20s to early 50s. All of them were college-educated, and all but one was employed. None of our participants were familiar with Android, but all owned and were comfortable with iPhones that had VoiceOver screen-reading software.

	P1	**P2**	**P3**	**P4**
Age	28	53	35	37
Gender	Female	Male	Male	Male
Education	PhD	Masters	Bachelors	Masters
Salary (Annual)	50k	100 - 150k	70 - 80k	24k

Table 1: The demographics of our four participants.

A summary of the participant demographics is given in Table 1. Throughout the rest of this paper we will refer to the participants as P1 through P4.

7 Usability Study Findings

7.1 Quantitative Feedback

The NASA-TLX ratings of the Respeak application that we received from our participants are summarized in Table 2. We found that users were able to complete tasks with very little physical demand but with a higher average mental demand of 5.5. This is expected, since Respeak asks users to carefully listen to audio files and try to memorize them. It was also the first time that any of the participants had used Respeak, so there was a slight learning curve in understanding the interface. This could have also been responsible for the slightly higher average effort rating of 5.75. When we asked users how well they felt they had performed in completing the tasks that were asked of them, they gave an average rating of 8. This indicates that the participants were comfortable with understanding the instructions and felt like what was asked of them was achievable. Finally, the average frustration rating

was 3.5. All of the participants told us that their frustration rating would have been even lower if they had not run into several issues regarding the Android TalkBack interface. We discuss this in further detail in section 7.2.3.

	P1	P2	P3	P4	Average
Mental Demand	5	3	6	8	5.5
Physical Demand	1	2	1	1	1.25
Performance	8	6	8	10	8
Effort	5	3	6	9	5.75
Frustration	5	5	2	2	3.5

Table 2: Participant ratings across 5 factors measured by the NASA-TLX scale.

In addition to rating Respeak on the NASA-TLX scale, we also asked the participants to discuss what they though about the overall accessibility of the application. When directly asked if they thought the application was accessible, every participant said "yes." On a scale of 1 to 10 (with 1 being the least accessible, 10 being the most), their average accessibility rating was 8.25. This indicates that overall, blind participants felt comfortable using Respeak and did not run into severely restricting accessibility issues.

7.2 Qualitative Feedback on Accessibility

Though quantitative feedback was helpful for us to get a global picture of the accessibility of the Respeak app, our best insights came from discussions with users and by asking them to provide the reasoning behind their ratings.

7.2.1 Positive Feedback

Several common themes came out from each discussion we had with users. All of our participants told us that they liked that Reseapk had a simple user interface and provided straightforward instructions. Participant P1 told us:

> *As far as the app, I think it's really nice. It's really simple, it's really accessible... The app itself is very easy to use and clear.*

Participants felt that it was easy to listen to audio files and record themselves respeaking, and they appreciated the cues that the app provided prompting that recording was beginning and ending. For example, one user told us he appreciated the tone that played once recording finished. In addition, users told us that they felt the audio files they were asked to listen to were short, clear, and easy to remember. Participant P4 said:

> *I was able to understand what was being said clearly. It was short enough [that] I could memorize it. I liked that I could go back and listen to audio files again.*

Two of the participants were intrigued and impressed by the ASR speech recognition accuracy. One of the usability study sessions was conducted in a coffee shop on a Sunday morning, and happened to be particularly noisy and busy, but the built-in Android ASR system was still able to pick up on participant P3's voice:

> *I was impressed at the accuracy of the transcription. I definitely felt that the voice transcription accurately captured what I had said.*

7.2.2 Critical Feedback

We also asked users what we could do to make the Respeak app more accessible. One of the biggest suggestions we received was to make it easier to navigate the app quickly and skip past instructions that have already been announced before. Though sighted people can take for granted the ability to simply ignore instructions they've already seen, blind people must rely on TalkBack, which does not have the intuition to skip over announcements and instructions the user already understands. This was an easy oversight to make that we did not catch from our own manual testing of the Respeak interface. Participant P4 suggested:

> *"This [instruction for recording audio] should have the option of being taken out. If I'm brand new to this app and I'm trying to learn what I need to do, then it could be helpful. There should be an option where we can eliminate it for the advanced user."*

Another problem with over-announcing was that it made it more difficult to keep the content of the audio that the user had listened to in working memory. After listening to and memorizing an audio file, users would have to listen to a long instruction that said *"Press the record button if you would like to repeat the audio again. This will open a Google app for recognizing speech,"* as shown in Figure 6, before actually being able to record. This was meant to help users learn what behavior to expect from the app. However, having this intermediate instruction in between could also be distracting and make it easier for users to forget the audio file they had just listened to. In a future iteration of Respeak, we would add an option to hide extra instructions for users who are already comfortable with the interface.

In a similar vein of over-announcing, while content grouping is generally a good practice, as we described in section 5.4, we found that it also should be used in moderation. On the Task History page, for example, we grouped together all of the account summary information, such as total amount earned and total payments processed, into one large announcement that takes around 30 seconds for TalkBack to read. Breaking this down into smaller sub-groups would be more manageable and make it easier to understand.

Some users found the audio file names such as `aikpart003.wav` in the Task History page confusing. While sighted users can see that this represents an audio file name with a `.wav` extension, blind users miss this information as TalkBack does not know how to properly pronounce the file name. In the future, we will make names identifying audio files more meaningful and easy to understand and drop file extensions. Finally, an additional suggestion we received from a participant was to add an initial, interactive tutorial at the beginning of the app that guides new users through the workflow.

7.2.3 Limitations of Android

The most common accessibility complaint we received was not with the Respeak application itself, but in the unresponsiveness of Android to user swipe gestures. Participant P2 told us:

> *I feel like when I flick through sometimes it works and sometimes it doesn't.*

Participants would sometimes have to repeat swipe or double-tap gestures multiple times before the device performed the intended action, which was frustrating. All of our participants were iPhone users and were used to the swiping behavior working seamlessly.

Unfortunately, this problem was outside of the scope of the Respeak application itself and seems to stem from lack of sensitivity in the hardware or from Android operating system issues. It will take further investigating for us to get behind the root cause.

7.3 Financial and Instrumental Benefits

In addition to collecting feedback on the Respeak application, we were also interested in understanding what participants thought about the Respeak ecosystem. We asked them questions regarding the financial and instrumental benefits of Respeak, and encouraged them to share any additional thoughts they had about the project.

We directly asked our participants, "Do you think the visually impaired population would receive any benefits in using the Respeak application? Please elaborate." Though all of our participants themselves either had full-time jobs or family financial support, they all thought Respeak's biggest benefit was the monetary support and employment it could provide. Participants felt that Respeak would be particularly helpful to unemployed visually impaired people, as it would provide a supplementary source of income. Participant P4, who was natively from India, thought Respeak could be successful in developing regions:

> *It's not much work... It would be very handy to make a little bit of money on the side. I would say a lot of poor people would use it if they have access to internet and an Android phone.*

Participant P3 felt that tasks could be completed quickly with a relatively high payout. However, he felt that if the reward per task was low, users might try to complete as many tasks as quickly as possible and not worry about their accuracy. He suggested increasing the accuracy threshold required to obtain the full reward:

> *I could rack up some money pretty quick. All the tasks seem to be 10 seconds or so, you're basically just hearing and repeating, so that's pretty simple... I think I would increase the accuracy to 90% or above to get the full reward.*

Our participants described benefits of using Respeak beyond just earning money. Many of them viewed employment as a way to increase independence and confidence. Participant P1, who had once used Amazon's MTurk for a school project, described having a frustrating experience because of the inaccessibility issues MTurk posed. She said that although developers may not be intentionally leaving out accessibility features, making a platform inaccessible sends out an implicit message that blind or other disabled people are not meant to use it, which can be damaging to self-esteem. In comparison, she found Respeak's accessibility empowering:

> *Mechanical Turk really *** pisses me off. When these work-related tasks are not accessible that tells me I'm not supposed to work, that I shouldn't work, that I can't work... [Respeak] is empowering, there's a work task I can proceed in without doing anything extra or having accessibility barriers.*

Other participants felt similarly about the empowering ability of employment. Participant P2 suggested that Respeak could be a way of improving social status and independence:

> *It benefits your own self-esteem, the respect you get from other people... They view you with less pity if they know you're economically making your way yourself in the word instead of depending on others.*

Finally, several participants said that listening to and completing Respeak tasks was a fun exercise. Participant P1 wanted to guess who the speaker was after listening to audio segments. Participant P2 felt that Respeak could allow users to improve their listening skills and short-term memory. Finally, participants P3 and P4 thought that Respeak was interesting from a technical perspective, and thought that people might want to use it to help improve speech recognition quality or as an opportunity to get more technology-savvy.

7.4 Future Work

Our participants had several interesting suggestions for future directions of Respeak. Several participants suggested gamifying the Respeak app by adding a leaderboard that ranks people by the number of tasks they have completed and their accuracy. One participant thought that it would be nice to be able to donate the money earned on Respeak to a charitable cause, since she didn't need the money herself but wanted to be able to help others. Finally, another participant suggested creating a Respeak skill for Amazon's Alexa [14] device. Although Alexa has limited language support as of this writing, and cannot be used on languages like Hindi, this may be interesting to investigate in the future.

Our immediate next steps are to do a larger scale usability study of Respeak with 10 to 15 low-income, blind users in India. This will help us gain additional insights about the accessibility of Respeak.

8 Conclusion

To our knowledge, Respeak is the first voice-based, crowd-powered, and accessible speech transcription system. Our system is designed with the purpose of helping low-income, blind people in the developing world participate in meaningful crowdsourcing work, while improving speech transcription quality and lowering cost for languages spoken in the developing world. In its first deployment with non-blind users in India, Respeak produced transcripts with a Word Error Rate of 15.2% on Indian English audio files and 8.6% on Hindi audio files. Our recent work has been focused on preparing Respeak for a second deployment with blind, low-income users by improving its user interface. A usability study with blind participants in Seattle suggested that Respeak has an accessible user interface. In addition, participants told us that Respeak may provide financial as well as instrumental benefits such as improved self-esteem to its users.

9 Acknowledgments

I would like to give a sincere thanks to Aditya Vashistha for his guidance on this project, mentorship, and inspiration over my past four years as an undergraduate. Thank you to Richard Anderson for his kind and constant support. In memory of Gaetano Borriello, who introduced me to research and helped me believe in my potential. Finally, a thanks to the wonderful members of the UW ICTD lab for their friendship and discussions.

References

[1] Shashank Khanna et al. "Evaluating and Improving the Usability of Mechanical Turk for Low-income Workers in India". In: *Proceedings of the First ACM Symposium on Computing for Development*. ACM DEV '10. London, United Kingdom: ACM, 2010, 12:1–12:10. ISBN: 978-1-4503-0473-3. DOI: 10.1145/1926180.1926195. URL: http://doi.acm.org/10.1145/1926180.1926195.

[2] Government of India. "Press Note on Release of Data on Houses, Household Amenities and Assets, Census 2011". In: (2011). URL: http://www.pewglobal.org/2016/02/22/smartphone-ownership-and-internet-usage-continues-to-climb-in-emerging-economies/.

[3] Chia-ying Lee and James Glass. "A Transcription Task for Crowdsourcing with Automatic Quality Contro". In: Interspeech, 2011.

[4] Prayag Narula et al. "MobileWorks: A Mobile Crowdsourcing Platform for Workers at the Bottom of the Pyramid". In: *Proceedings of the 11th AAAI Conference on Human Computation*. AAAI Press, 2011. URL: http://dl.acm.org/citation.cfm?id=2908698.2908723.

[5] Iftekhar Naim et al. "Text Alignment for Real-Time Crowd Captioning". In: HLT-NAACL, 2013.

[6] *Global Findex 2014 - Financial Inclusion*. 2014. URL: http://datatopics.worldbank.org/financialinclusion/country/india.

[7] *Visual Impairment and Blindness*. 2014. URL: http://www.who.int/mediacentre/factsheets/fs282/en/.

[8] Erin Brady and Jeffrey P. Bigham. "Crowdsourcing Accessibility: Human-Powered Access Technologies". In: *Found. Trends Hum.-Comput. Interact.* 8.4 (Nov. 2015), pp. 273–372. ISSN: 1551-3955. DOI: 10.1561/1100000050. URL: http://dx.doi.org/10.1561/1100000050.

[9] Yu Zhong et al. "RegionSpeak: Quick Comprehensive Spatial Descriptions of Complex Images for Blind Users". In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. CHI '15. Seoul, Republic of Korea: ACM, 2015, pp. 2353–2362. URL: http://doi.acm.org/10.1145/2702123.2702437.

[10] Kathryn Zyskowski et al. "Accessible Crowdwork?: Understanding the Value in and Challenge of Microtask Employment for People with Disabilities". In: *CSCW*. 2015.

[11] *Medical Transcription Services Market - Global Industry Analysis, Size, Share, Growth, Trends and Forecast, 2013 - 2019*. Tech. rep. 2016.

[12] *About Cloud Endpoints Frameworks*. 2017. URL: https://cloud.google.com/endpoints/docs/frameworks/java/about-cloud-endpoints-frameworks.

[13] *Accessibility Developer Checklist*. 2017. URL: https://developer.android.com/guide/topics/ui/accessibility/checklist.html.

[14] *Amazon Alexa*. 2017. URL: http://alexa.amazon.com/spa/index.html.

[15] *Amazon Mechanical Turk*. 2017. URL: https://www.mturk.com/mturk/welcome.

[16] *Blindness Statistics*. 2017. URL: https://nfb.org/blindness-statistics.

[17] *CastingWords Transcription*. 2017. URL: https://castingwords.com/.

[18] *Cloud Speech API*. 2017. URL: https://cloud.google.com/speech/.

[19] *Get started on Android with TalkBack*. 2017. URL: https://support.google.com/accessibility/android/answer/6283677?hl=en.

[20] *Google Cloud SQL*. 2017. URL: https://cloud.google.com/sql/docs/mysql/.

[21] *India Average Daily Age Wage Rate*. 2017. URL: http://www.tradingeconomics.com/india/wages/forecast.

[22] *NASA TLX: Task Load Index*. 2017. URL: https://humansystems.arc.nasa.gov/groups/tlx/.

[23] *Nuance Dragon*. 2017. URL: https://www.nuance.com/dragon.html.

[24] *Quick Transcription Service*. 2017. URL: http://www.quicktranscriptionservice.com/Hindi-Transcription.html.

[25] *Samasource*. 2017. URL: https://www.samasource.org/.

[26] *Scripts Complete*. 2017. URL: https://scriptscomplete.com/Hindi-Transcription-Services.php.

[27] *SpeechPad*. 2017. URL: https://www.speechpad.com/.

[28] *SpeechRecognizer*. 2017. URL: https://developer.android.com/reference/android/speech/SpeechRecognizer.html.

[29] Aditya Vashistha, Pooja Sethi, and Richard Anderson. "Respeak: A Voice-based, Crowd-powered Speech Transcription System". In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. CHI '17. Denver, Colorado, USA: ACM, 2017. URL: http://doi.acm.org/10.1145/3025453.3025640.

[30] Chia-ying Lee and James Glass. "Toward better crowdsourced transcription: Transcription of a year of the Let's Go Bus Information System data". In: SLT.

[31] Jacob Poushter. *Smartphone Ownership and Internet Usage Continues to Climb in Emerging Economies*. URL: http://www.pewglobal.org/2016/02/22/smartphone-ownership-and-internet-usage-continues-to-climb-in-emerging-economies/.