

**A Dual-Use Speech CAPTCHA: Aiding Visually Impaired Web Users
while Providing Transcriptions of Audio Streams**

Andy Schlaikjer
hazen+@cs.cmu.edu

CMU-LTI-07-014

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
<http://www.lti.cs.cmu.edu/>

November, 2007

Abstract

This work focuses on development of an alternative aural CAPTCHA technology based on a speech transcription task. The principal goals are (1) to provide a better quality of service for visually impaired web users who currently have few alternatives for accessing web services protected by CAPTCHAs based on visual perception tasks, (2) as a side effect of CAPTCHA use, collect meaning data, such as transcriptions of speech found in online audio streams, and (3) deploy the CAPTCHA publicly to increase the general awareness of accessibility concerns on the web for users with disabilities. Toward these ends, an experimental corpus of human transcriptions of short audio clips was created to support aural CAPTCHA experimentation. Results suggest that the techniques investigated represent viable aural reCAPTCHA mechanisms.

Contents

1	Introduction	4
2	Background	5
3	Corpus Development	6
3.1	System	6
3.2	User Study	6
3.3	Analysis	7
4	Experiments	9
5	Results	10
6	Conclusions	10
7	Future Work	13
8	Acknowledgements	13

1 Introduction

As the World Wide Web has matured, many web services have been developed by businesses and organizations to empower their clients, from online banking, email, and shopping, to other community-oriented applications such as blogs and web forums. Unfortunately, with the increasing adoption of these technologies, they have become targets for fraud and spam. Many of these services are routinely attacked by automated software agents which pose a substantial security risk and erode the quality of service experienced by normal users.

One of the primary defense mechanisms developed to ward off automated software agents is the *CAPTCHA* [13], or “**C**ompletely **A**utomated **P**ublic **T**uring test to tell **C**omputers and **H**umans **A**part.” A CAPTCHA is a test which is trivial for a human to solve, but which can’t yet be handled automatically by computers. By incorporating a CAPTCHA into a web application, automated attempts to abuse the application can be thwarted. Existing CAPTCHAs have mainly focused on visual character or object recognition tasks [8, 10, 2], and have been deployed on many online services¹. Typically, for these tasks, or “challenges”, a user is presented with a distorted image containing a short sequence of Latin or numeric characters and he or she must recognize and enter these characters in a text field. The image is generated by the website on request so each CAPTCHA is different from the last, but still easy to verify by comparing a users’ response string to the string used to generate the image.

Unfortunately, the use of a CAPTCHA limits the accessibility of a website, as some tests can be difficult for certain users to perform. In the case of visual CAPTCHAs, visually impaired web users are particularly disadvantaged. Site operators must therefore decide whether adoption of a CAPTCHA is appropriate for their application, weighing the relative benefits of automated access prevention and accessibility. Some sensitive applications, such as online banking, may warrant stringent safeguards against automated access at the sake of accessibility, while other applications, for instance public loges or forums, may not want to inhibit user participation by incorporation of a CAPTCHA. Given that there are services sensitive enough to merit deployment of CAPTCHA technology, alternative CAPTCHAs which meet the needs of user communities with disabilities should be investigated.

Initial attempts have been made to support CAPTCHAs based on aural tasks, instead of purely visual tasks. These aural CAPTCHAs are modeled closely on the character recognition tasks initially pioneered for visual CAPTCHAs; A sequence of random digits is generated, but instead of creating a distorted image from these digits an audio clip is made containing spoken digits and added noise. Unfortunately, because of the high performance of current Automatic Speech Recognition (ASR) technologies on recognition of spoken digits, the amount of noise that must be added to the signal to confound these systems also makes the task difficult for humans.

Another shortcoming of many existing CAPTCHAs is that the data they collect from users is thrown away once a test has been performed. A new type of CAPTCHA called *reCAPTCHA* improves upon this by attempting to create tasks whose answers capture data that can in some sense be “reused”. Currently, reCAPTCHA.net provides a reCAPTCHA service which generates visual CAPTCHAs from scanned images of real books. User responses to these CAPTCHAs help provide textual transcripts for these materials, thereby improving the books’ accessibility via search and other technologies. An aural reCAPTCHA is not yet supported by this service, and as a fall-back a simple audio digits CAPTCHA, as described above, is provided.

To improve CAPTCHA performance and accessibility for visually impaired web users, as well as provide a meaningful reCAPTCHA for audio data, this project investigates an alternative aural CAPTCHA based on an open-domain speech transcription task. It is our hypothesis that open-domain speech transcription represents a task appropriate for implementation of an effective aural reCAPTCHA technology. Section 2 provides more detail on existing aural CAPTCHAs and the proposed open-domain speech transcription paradigm for aural reCAPTCHA. Section 3 describes a web-based data collection system developed to facilitate user studies and creation of a development corpus. Experiments with this corpus are outlined in section 4. Results are discussed in section 5, and conclusions and future work in sections 6 and 7 respectively.

¹Registration pages at Google [5], Yahoo! [15], and Microsoft [7] websites all implement some form of CAPTCHA technology.

2 Background

As mentioned in Section 1 current aural CAPTCHAs are based on a spoken digit recognition task. With this technique, a series of random digits from zero to nine are generated and rendered as speech with one of a variety of methods. One common approach is to use a collection of prerecorded samples of speech for each digit and compose these samples based on the generated sequence of digits. After the separate clips are combined, various sources of noise (e.g. background speech) or distortion (e.g. sound stage modification or reverberation effects) are applied to the resulting signal to complicate the recognition task for automatic systems.

Unfortunately, there is evidence that this form of CAPTCHA is not easy for humans to perform. Analysis of runtime logs from the reCAPTCHA.net service indicate that the failure rate of humans on the (noisy) spoken digit recognition task is much higher than on the visual word recognition task. It may be the case that the amount of noise which must be added to audio clips to confound automatic recognition techniques is prohibitive for human recognition as well. However, this added noise is likely required to ensure the security of the CAPTCHA, as accuracy of state-of-the-art ASR systems on spoken digits is generally high, even when multiple source voices or types of noise are present in the audio signal being decoded.

Another reason spoken digit CAPTCHAs might be difficult for humans to solve is due to the randomness of the generated sequence of digits. Because any digit is equally likely to precede or follow any other digit, there is little contextual information humans may leverage to complete the task. Conversely, for visual word recognition tasks, humans may take advantage of context to help classify any particularly obscured character within a word. For this reason, most visual CAPTCHAs based on character recognition have moved away from using random sequences of characters, and instead use whole English words, or nonsense words whose characters follow distributional properties similar to those of real words. This allows humans to apply their knowledge of language to recognize contextual clues in these CAPTCHAs.

To move beyond spoken digit CAPTCHAs and attempt to overcome the limitations outlined above, we propose the use of open domain speech for aural CAPTCHAs. With this approach, a user is played a short segment of audio and must recognize and transcribe any speech they hear. The topical content of the speech, speaker dialect, gender, and noise characteristics of the audio segment are all kept variable. Open domain speech recognition is much more difficult for automatic techniques to handle, so less noise must be added to a given audio clip to confound automatic techniques. At the same time, recognition of open domain speech is arguably a more natural task for humans than recognition of random spoken digits, as it allows humans to apply their knowledge of language and the world at large to aid recognition.

Another motivating factor for a challenge based on open domain speech transcription is the potential for reuse of any data collected during CAPTCHA evaluation. Using the paired-challenge reCAPTCHA paradigm, two aural challenges could be presented to a user, one which can be validated using data collected from prior users, and another for which data is still being collected. If the user’s response to the former is successfully validated, then their response to the latter is recorded and used in the future to construct “ground truth” for the new challenge. If a sufficient number of users were to interact with such a system, it may be possible to transcribe a significant amount of previously unannotated audio data with reasonable accuracy. These transcriptions could then be used to improve audio search, generate textual summaries of audio, or to perform other tasks which generally improve the accessibility of the original source audio. Generating reCAPTCHAs from audio streams found online might have a considerable impact on accessibility of audio sources on the Web.

One of the principal difficulties in adapting the open domain speech transcription task for use as a (re)CAPTCHA is that the space of “correct” human transcriptions of a given audio clip is large. For character recognition tasks, humans can be expected to produce a specific response string which may be validated against a canonical “answer” string. However, for speech transcription, human responses to even 5 seconds of speech are likely to show large variation due to differences in spelling (or misspelled words), punctuation, inclusion of “filler” words (e.g. “ah”, “um”), representation of numbers and quantities, abbreviations, etc. This makes validation of responses more difficult, as the space of “correct” variability must be modeled for a given clip, and new responses compared in some way to this model.

Additionally, for this type of CAPTCHA language understanding is crucial. Performance of human users on this task is highly dependent on their fluency. This means younger users whose language skills are not well developed, non-native speakers, or users with language or learning disabilities may not perform well on

this type of CAPTCHA.

3 Corpus Development

To facilitate experiments with open-domain speech transcription for CAPTCHAs, a user study was conducted to collect data for a small development corpus. Objectives of the user study were two-fold; to investigate design and instrumentation issues for blind-accessible web-based aural CAPTCHAs, as well as collect many independent transcriptions of the same set of audio clips from different users. Having such a dataset affords us the opportunity to investigate sources of variability in human responses to this task, and methods to automatically encode this variability for CAPTCHA validation.

3.1 System

A web application, entitled *Audio reCAPTCHA*, was developed using Java Servlet and Java Server Page (JSP) technologies and hosted with Apache Tomcat and MySQL to support data collection. System organization roughly corresponds to a Model 2 (Model-View-Controller) architecture. A simple domain model describes **Users**, **Source** audio files, **Segments** of source audio files, **Responses** from users to segments, and “gold-standard” **Answer** transcriptions for segments. Persistence and retrieval of domain objects is supported by an abstract Data Access Object (DAO) layer with an accompanying MySQL implementation. The primary human interface, or “view” layer, is supported by a series of JSP pages, and controlled by an associated set of Servlets and high-level control components.

Special attention was paid to the structure of the JSP and resulting HTML output of the view layer, to ensure that visually impaired users would be able to navigate the web application with ease, and properly interact with its control mechanisms. Web accessibility guidelines from multiple online sources [12, 14, 11, 9, 4] were consulted to identify major sources of difficulty for accessible web page design. “Skip navigation” links were added at the beginning of all pages to allow quick navigation to content sections. Such links allow extraneous or redundant information, such as page headers, advertisements, or site links, to be skipped which might otherwise confound users who must interact with a linearization of web page content. Attempts were made to avoid the use of HTML markup for formatting. Instead, Cascading Style Sheets (CSS) were used to define visual layout and style, and HTML markup was used to describe the structural semantics of page content (e.g. major sections, lists, links, and forms). All embedded multimedia such as images were labeled with `alt` or `title` attributes to provide textual descriptions of their content.

All pages were tested with Internet Explorer 6 on Microsoft Windows XP, Mozilla Firefox 2 on Windows XP, Red Hat FC5, and Mac OSX, and the Lynx browser on Red Hat FC5. Tests with Lynx were particularly useful, as this browser renders page content as pure text, and in the process must linearize all content. This linearization provides some corollary to the speech and Braille presentation afforded blind users by screen reader software. Additional testing of the site was performed by Darrell Shandrow, a blind information technology professional, with the JAWS (Job Access With Speech) screen reader software. His feedback was positive on initial designs, and led to specific refinements of the application’s input forms and audio controls to address accessibility concerns for blind users. Specifically, although Adobe Flash² was chosen to facilitate audio playback in the web application due to its wide-spread adoption and MP3 decoding support, interactive Flash media often poses significant accessibility issues because embedded controls do not support keyboard navigation and interaction uniformly across different computing environments. For this reason, all Flash-based interface elements controlling audio playback were replaced with simple HTML-based controls.

3.2 User Study

Source audio for the study was drawn from the Internet Archive’s audio collection [1], and consisted of alternative news radio content. This type of data was targeted because it contains a large number of variable properties. Speaker gender, age, speech patterns, dialect, topical content, sources of noise, recording quality, and even genre of speech (e.g. poetry) are all variable throughout the source data. Any subset of these

²The XSPF Web Music Player [16] was used to support Flash playback of MP3 encoded audio files.

1. To begin audio playback, click the “Start audio playback” link below. A short tone will play, followed by the audio clip. These will repeat indefinitely, until you click the “Stop audio playback” link, or submit a response.
2. Listen to the clip and transcribe any English speech you hear in the “Transcription” text field. Don’t worry if you can’t recognize all words in the clip. Just write down those words you can understand, in the order in which you hear them. For instance, it is likely that words and the beginning or end of the clip will be cut off. These words may be ignored.
3. If you hear some speech in the clip, but can’t recognize a single word, or you can’t hear any speech at all, select the “Unintelligible”, or “No Speech” option respectively. Note that by selecting one of these options, any transcription provided will be ignored.
4. Submit your response by clicking the “Submit Response” button at the bottom of this page.

Figure 1: User study task instructions.

sources of variation can cause great difficulty for automatic speech recognition techniques, but for many human listeners don’t pose as significant an obstacle for recognition.

This source audio was automatically segmented into small, five second clips by a simple “sliding window” technique. Ideally, some form of speech endpointing should first be applied to source audio to identify regions of speech and non-speech data. Audio clips could then be created from only those regions of the signal thought to contain speech in order to lessen the number of clips generated which might not contain any speech at all. However, for this study over 400 clips were manually inspected and annotated with information regarding the types of speech present in each. From these annotated clips, 100 were chosen and ordered to define a common audio clip presentation order for all participants. Having all participants observe the same sequence of audio clips ensured multiple transcriptions are accumulated for those clips at the start of the sequence. These initial clips were chosen strategically to cover certain phenomena, such as proper names, quantities / punctuation, different speakers, voice qualities / dialects / accents, genre, background noise, and silence.

Participants for the data collection task were recruited via email from various sources, including the School of Computer Science community at Carnegie Mellon University, individuals outside of CMU, as well as specific distribution lists for those interested in accessibility issues for visually impaired users.

Study participants were asked to visit the web site hosting the web application introduced above, where they could read a brief overview of the task they would be asked to perform. If they chose to continue, participants were asked to create an account with the web application, specifying a username, password, and email address³, as well as whether or not they were a native speaker of English, and a qualitative measure of their level of visual impairment. These qualitative measures were: *low* (user has little or no trouble with his/her vision), *medium* (user prefers high contrast displays), and *high* (user is blind, or prefers non-visual interfaces).

After successful account creation, participants were shown a “home page” where they could perform one of three operations: listen to an audio clip, modify their account information, or log out of the application. By selecting the first of these choices, participants were presented with a list of task instructions, a set of audio playback controls, and the primary data entry interface for submission of audio clip transcriptions.

It is important to note that task instructions, reproduced in Figure 1, were intentionally kept brief to promote a certain amount of user interpretation of task constraints. In a real CAPTCHA scenario, having a complex set of instructions may put off many human users, so keeping task instructions simple is a realistic requirement of a viable CAPTCHA technology. In addition, ambiguity in task constraints promotes variability in user responses, which is one of our goals for corpus development.

3.3 Analysis

The resulting data collected by the user study included responses from 55 unique participants, about 12% of whom specified their level of visual impairment as either “medium” or “high”. It would have been desirable

³Because email addresses were collected representing personally identifiable information about participants, this study was submitted for Carnegie Mellon University IRB review and received Board approval on October 4, 2007 [3].

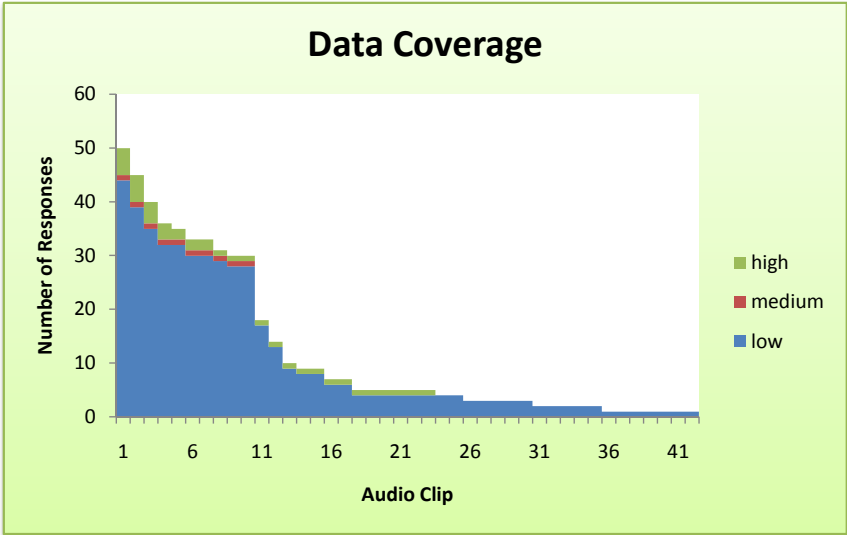


Figure 2: Graph depicting the number of responses collected from users with different levels of visual impairment for each audio clip.

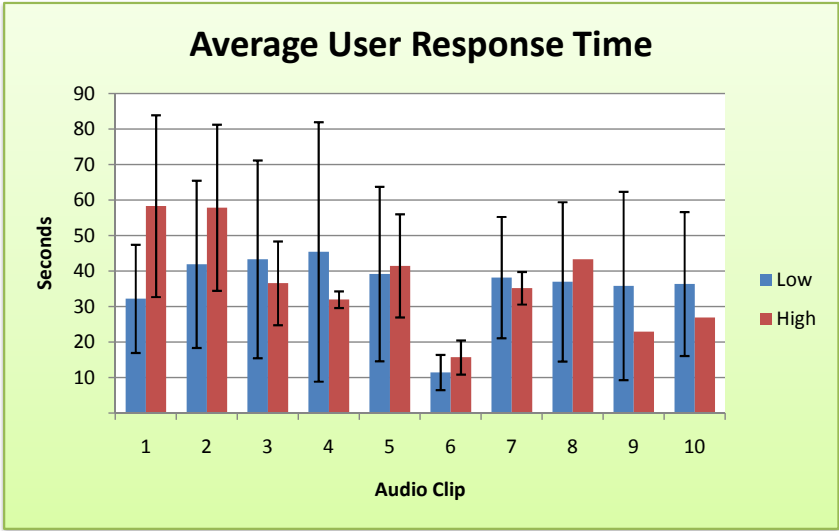


Figure 3: Graph depicting the average response time of participants with low and high levels of visual impairment to the first 10 audio clips. Error bars represent standard deviation.

to have a higher level of participation on the part of visually impaired users, but at least those who did participate did not report having any issue interacting with the site. Overall, about 500 transcriptions were collected across the first 42 audio clips. Figure 2 shows a graph depicting the total response coverage of the sequence of audio clips. As can be seen from this figure, most participants only responded to the first ten audio clips, which was the minimum number of transcriptions requested of each participant. Figure 3 shows a plot of the average response time in milliseconds of participants with low and high visual impairment to each of the first ten audio clips. Here we can see what might be a slightly steeper learning curve for participants with high visual impairment than for those with low visual impairment. However, after the first two clips, it appears that visual impairment does not significantly influence participants’ average response time. Note that audio clip 6 contained only silence, and response times for participants with both low and high visual impairment are consequently much faster than for the other 9 clips.

4 Experiments

To complement the data collected during corpus development, a number of automatic speech recognition systems were applied to the same data, to provide some baseline automatic responses with which to compare the human ones. ASR systems are roughly parameterized by four models; a *vocabulary* specifying all of the words the recognizer can successfully identify, a *phone dictionary* mapping words from the vocabulary to their various pronunciations, an *acoustic model* binding features of the audio signal itself to the phones present in the phone dictionary, and a *language model* or *grammar* describing constraints on the expected ordering of words. Most ASR systems are highly sensitive to the specification and training of these models.

The first ASR system applied to this data was the CMU Sphinx [6] family of speech recognizers. The publicly available HUB4 acoustic and language models were used as of all the available models for Sphinx, these support the largest vocabulary (roughly 64,000 words). Unfortunately, using a vocabulary of this size already suggests that this model’s application to open domain speech, whose vocabulary is likely much larger, is inappropriate. However, the development of a more comprehensive set of models for Sphinx was beyond the scope of this project.

The second ASR system applied was Nuance Dragon™ NaturallySpeaking® 9, a commercial voice dictation software package. This system supports online speaker adaptation where as the user speaks he or she may correct any mistakes the recognizer makes, allowing the system to refine its models to improve recognition performance. The system also provides a batch transcription function meant to allow transcription of audio files recorded by the primary user. Using a new user profile (without attempting to retrain the profile for recognition of a specific speaker) this function was applied to our test clips.

Once automatic data was collected for the corpus audio clips, a simple model of “ground truth” was generated for each clip to encode the clip’s space of “correct” transcriptions. As a baseline, this model was defined by a small, randomly selected subset of the available human responses for a given clip. All other responses, both human and automatic, could then be compared with this basic model in a variety of ways, to determine what kinds of distance metrics might be useful for differentiating human from automatic responses. Experiments with more complex models of ground truth have been planned, including selection of characteristic transcriptions based on some initial clustering of available human responses, but results of these experiments are left for future work.

Many approximate string matching techniques are applicable here. Hamming distance, the more general Levenshtein (edit) distance, or n-gram distance are a few possible metrics for comparing new responses with those responses selected as ground truth. Because response strings are assumed to be closely related, having been generated by transcription of the same source audio, but are not likely to have uniform lengths, experiments were conducted here with Levenshtein distance, using uniform insertion and deletion penalties to ensure symmetry of the metric. Additionally, edit distance was normalized based on the length of the larger of the two strings being compared (the maximum edit distance between the strings), such that calculated distances could be compared with one another.

$$D(s, t) = \begin{cases} 0 & \text{if } \max(|s|, |t|) = 0. \\ \text{Levenshtein}(s, t) / \max(|s|, |t|) & \text{otherwise.} \end{cases} \quad (1)$$

It should be noted that the complexity of Levenshtein distance metric between strings s of length m and

t of length n is $O(m * n)$. Because the response strings we are dealing with here are usually fewer than 100 characters in length, this complexity does not impose a serious performance bottleneck for a real CAPTCHA implementation. However, this metric is much more expensive than the regular string matching used for validation of conventional CAPTCHAs, which is generally linear. Optimizations of Levenshtein distance would be worth considering for large-scale deployment of a CAPTCHA which incorporates this metric for validation.

5 Results

Unfortunately, due to large differences in content and format of the development corpus audio clips to the HUB4 models, the Sphinx recognizer was unable to produce any reasonable decoding results for these inputs. Various parameter settings were tested, but no settings were found to substantially improve performance. One reason for this may be that the HUB4 language model was built with specific utterance begin and end markers, which influence what sequence of words the decoder predicts for a given input. Since the boundaries of our clips were not aligned with utterance boundaries, the language model weights for probabilities of words or sequences of words at the beginning and end of each input segment are drastically misrepresented. Additionally, the HUB4 acoustic model likely does not match well the characteristics of the development corpus clips.

More success was found with the commercial voice dictation software. For some of the clearer audio clips, the accuracy of the software approached that of human transcriptions. Figure 4 depicts some human and automatic transcriptions for a few of the audio clips in the development corpus. Note that the automatic results seem to change from one run to the next for the same input. This is likely do to some form of model adaptation being performed during batch transcription, although it is unclear exactly what the voice dictation software is doing here.

Figure 5 displays a set of histograms depicting the average number of matching ground truth responses (“answer” strings) by normalized edit distance for both human and automatic responses. At a given normalized edit distance x on the horizontal axes, any blue mass represents the average number of answer strings matched by *human* response strings at that distance. Any red mass represents the average number of answer strings matched by *automatic* response strings at that distance. The blue and red quantities are stacked if both occur at the same horizontal position. From these graphs we can see that using normalized Levenshtein distance alone human and automatic response strings can be differentiated from one another. The blue and red masses are generally separable at some normalized edit distance. For those clips which were more difficult for users to transcribe, the blue and red masses are not as easily differentiated. In this way, “difficult” clips can be identified, and possibly prohibited from use as “validating” CAPTCHAs in the reCAPTCHA scenario. Of course, data could still be collected and recorded for these clips, but simply go unused for further validation.

6 Conclusions

A CAPTCHA must be simple to construct, accept most human responses, reject most automated responses, and be simple to validate. Of course, the degree to which each of these requirements must hold is dependent on application requirements. However, from the analysis performed here, it is clear that open-domain speech transcription represents a viable alternative aural CAPTCHA mechanism in light of these requirements. In addition, the nature of the open-domain speech transcription task lends itself to collection of useful data, fulfilling the added requirement of any reCAPTCHA solution.

Contributions of this project include the development of a web-based speech transcription data collection framework, whose design may influence further development of a large-scale web-based aural reCAPTCHA solution. Utilities were created to facilitate rapid creation of audio clips from source audio streams, handling various signal processing tasks and data management. With this data collection framework, a development corpus was created, which facilitated our experiments, and may still pose useful for future work. Preliminary experiments were performed with ASR for digit and large-vocabulary recognition. The viability of open-domain speech transcription for CAPTCHA was demonstrated by data analysis and feasibility study using our development corpus, and baseline algorithms for CAPTCHA validation were presented.

- Clip 1 (h) nothing is going to change, and that's why we don't see anything
(h) nothing is going to change and that's why we don't see anything
(h) nothing is going to change and thats we dont see anything
(a) not going and that we will be a thing
(a) not going to train and why we don't see anything
(a) nothing is going to train and why we don't see anything
- Clip 2 (h) michel israel age 18 are walking 3000 miles from san francisco
(h) michael israel h18 are walking 3000 miles from sanfranciso
(h) michael israel age eighteen are walking three thousand miles from san francisco
(a) is I will be house in San Francisco
(a) as you know 1803 housed in an
(a) as you know 1803 to housed in an
- Clip 3 (h) studs terkel mumia abu jamal
(h) (un)derwood – studs turkel – mumia abu jamal – for more
(h) studs turkle mia awugemal for more
(a) in and of him
(a) I say there was a
(a) I say we let him
- Clip 4 (h) It's, it's, it'll, it'll have to be a kind of
(h) it's it's ... it'll have to be a kind of temporizing(?)
(h) and it's it's It'll have to be a kind of temporizing
(a) it will all have a card that I
(a) if it'll it'll have to be kept alive
- Clip 5 (h) she talks about the humanity of the lebanese people palestinians and the refugees
(h) She talks about the humanity of the Lebanese people, the palestinians and the refugees
(h) she talked about the humanity of the lebanese people, the palestinians and the refugees
(a) out humanity let me go teen
(a) and out humanity let me go Lintner
- Clip 7 (h) holding flag the blood the soul supremacy the swallow stars
(h) Flags of blood, the souls of grimace, the sallow stars
(h) the soul supremacy the swallow stars
(a) will allow the souls of Gramercy is lower the
(a) little random blood and souls and see Islam as a
(a) mangled blood and souls and see Islam as a

Figure 4: Samples of human (h) and automatic (a) transcriptions for audio clips 1-5, and 7.

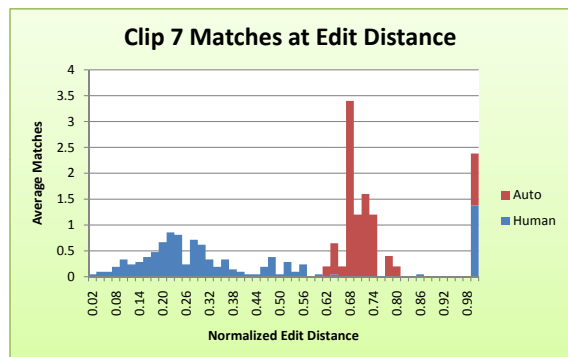
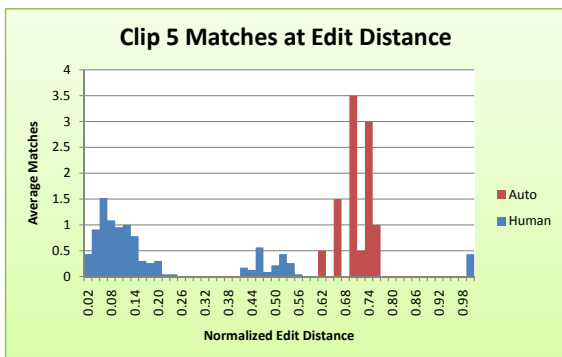
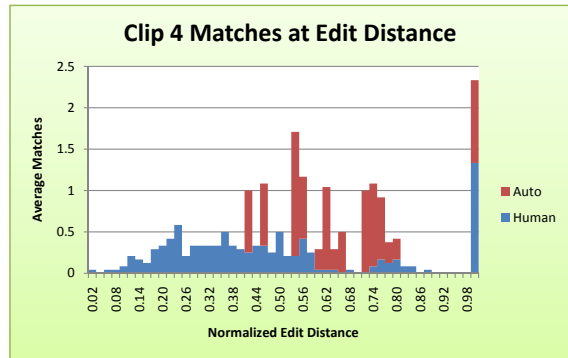
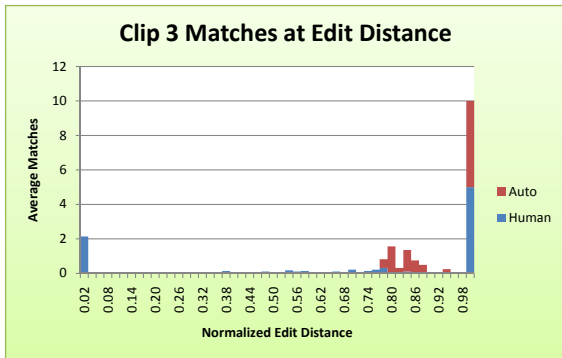
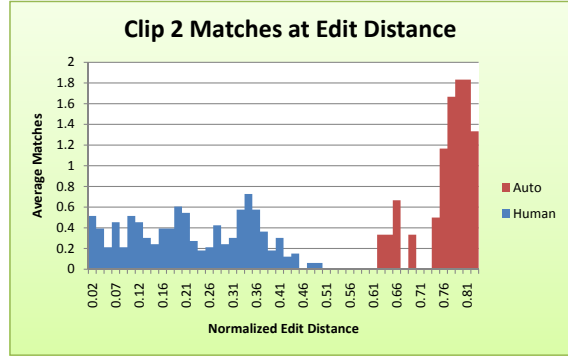
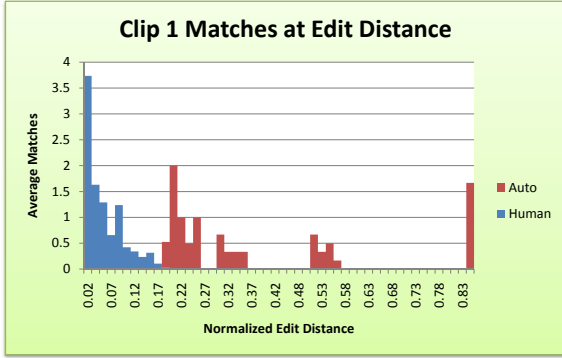


Figure 5: Graphs depicting average number of matching answer strings by normalized edit distance for human and automatic responses to various audio clips. Note that clip 6 is not included here because it was used as a control; it contained only silence, which all human users properly recognized.

7 Future Work

A number of experiments have been planned to investigate further refinements to the baseline validation algorithms outlined here. For instance, construction of “ground truth” for a given audio clip could be improved by adopting more complex response sampling strategies, including the use of an initial clustering of responses to inform sampling. Alternative approaches for encoding ground truth using probabilistic models are also planned. Further adaptation of open-source ASR systems is required to improve the automated baseline used to determine decision boundaries for response validation, though alternate means for determining these thresholds without ASR output will also be investigated. Finally, further optimizations will be performed to improve validation throughput in preparation for deployment of this technology within a larger CAPTCHA framework, such as reCAPTCHA.net.

8 Acknowledgements

Many thanks go to those who helped support this work including V-Unit Coordinators Barnardine Dias, Manuela Veloso, Sarah Belousov; Project Advisor Luis von Ahn; Microsoft, Google, Yahoo, and reCAPTCHA.net; Study participants! And for many helpful discussions Eric Nyberg, Ziad Al Bawab, Antoine Raux, Alex Hauptmann, Alan Black, Susanne Burger, Matthew Bell, Lingyun Gu, David Huggins-Daines, Mike Crawford, Justin Betteridge, Matthew Bilotti, Darrell Shandrow, and Shiry Ginosar.

References

- [1] Internet Archive. Internet archive: Audio archive. website, August 2007. <http://www.archive.org/details/audio>.
- [2] Henry S. Baird, Michael A. Moll, and Sui-Yu Wang. ScatterType: a legible but hard-to-segment CAPTCHA. In *Proceedings of the 8th International Conference on Document Analysis and Recognition*, Seoul, Korea, August 2005.
- [3] Robyn M. Dawes. Certification of IRB approval, October 2007. IRB Registration No: IRB00000603.
- [4] Freedom Scientific. Surfing the internet with JAWS. website, June 2007. http://www.freedomscientific.com/fs_products/Surfs_Up/_Surfs_Up_Start_Here.htm.
- [5] Google. Google Accounts, April 2007. <https://www.google.com/accounts/NewAccount>.
- [6] CMUSphinx Group. *CMUSphinx: The Carnegie Mellon Sphinx Project*. Carnegie Mellon University, April 2007. <http://www.cmusphinx.org/>.
- [7] Microsoft. Sign Up, April 2007. <https://accountservices.passport.net/reg.srf>.
- [8] Greg Mori and Jitendra Malik. Recognizing objects in adversarial clutter: breaking a visual CAPTCHA. In *Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03)*, June 2003.
- [9] Mark Pilgrim. Dive into accessibility. website, August 2007. <http://diveintoaccessibility.org/>.
- [10] Amalia Rusu and Venu Govindaraju. Handwritten CAPTCHA: Using the difference in the abilities of humans and machines in reading handwritten words. In *Proceedings of the 9th International Workshop on Frontiers in Handwriting Recognition (IWFHR'04)*, pages 226–231, 2004.
- [11] James W. Thatcher. Web accessibility for section 508. website, August 2007. <http://www.jimthatcher.com/webcourse1.htm>.
- [12] United States Access Board. Web-based intranet and internet information and applications (1194.22). website, August 2007. <http://www.access-board.gov/sec508/guide/1194.22.htm>.

- [13] Luis von Ahn, Manuel Blum, Nick Hopper, and John Langford. CAPTCHA: Using hard AI problems for security. In *Proceedings of Eurocrypt 2003*, pages 294–311, 2003.
- [14] W3C. Web accessibility initiative (WAI). website, August 2007. <http://www.w3.org/WAI/>.
- [15] Yahoo! Yahoo! Registration, April 2007. https://edit.yahoo.com/config/eval_register.
- [16] Fabricio Zuardi. *XSPF Web Music Player*, August 2007. <http://musicplayer.sourceforge.net/>.