Human Interactive Proofs for Spoken Language Interfaces[*]
(Extended Abstract)

Daniel Lopresti *Chilin Shih* *Gregory Kochanski*

Bell Labs, Lucent Technologies Inc.
600 Mountain Avenue
Murray Hill, NJ 07974 USA
{dpl,cls,gpk}@research.bell-labs.com

The wealth of information available on the Internet has created an incentive for the development of fully automated programs that try to exploit online services intended for human users. As a result, there is a compelling need for automatic methods (*i.e.*, algorithms) for telling whether the entity attempting to access a service is a human or a machine.

Coates, Baird, and Fateman have developed just such a test based on a task from the field of vision [1], influenced by work being done by von Ahn, *et al.* in the CAPTCHA project at Carnegie-Mellon University [2]. Their ideas are based on the observation that state-of-the-art optical character recognition (OCR) systems are not as adept as humans at reading degraded text. With a large dictionary, a library of differing font styles, and a variety of synthetic noise models, a nearly endless supply of word images can be generated. The user is then asked to type on the keyboard a randomly-chosen word displayed on the screen. It is easy (indeed, trivial) to verify whether the user has passed.

While most of today's web surfing takes place using computers that have monitors and keyboards, speech interfaces are proliferating rapidly and will play important roles in mobile devices and at times when the user wants her hands or eyes free. Although the problems in building a "bot" to navigate a spoken language interface may seem formidable, they are tractable, especially if the system depends on a fixed sequence of predefined prompts. Hence, we can anticipate a demand for similar methodologies to prevent machines from abusing speech-based resources intended for human users. Since most of the scenarios involved here offer either no screen (services accessed while driving a car) or at best a small one (cell phones), and often no keyboard, the test described by Coates, *et al.* cannot be applied.

There are a number of strategies we can explore for differentiating between humans and machines when it comes to the perception of speech:

- Making the dialogue difficult for machines (high level: syntactic or semantic).
- Making the speech signal difficult for machines (low level: recognizing phonemes).

[*] Presented at the *Workshop on Human Interactive Proofs*, Palo Alto, CA, January 2002.

- Processing the speech to create aural illusions that affect humans but not machines.

This overview summarizes our first steps towards studying this new type of Human Interactive Proof (HIP). We discuss several possible approaches to the problem of determining whether a user is a human or a machine when the mode of interaction is speech. We also suggest that such tests, whether spoken or graphical, do not necessarily have to be designed so that they are hard for the machine and easy for the human; the converse, which exploits idiosyncrasies in human perception, is equally valid.

1 Designing Dialogs that are Difficult for Machines

As did Coates, Baird, and Fateman, we would like to exploit the fact that certain pattern recognition tasks are significantly harder for machines than they are for humans. In this case, we will use text-to-speech synthesis (TTS) to generate tests, and take advantage of the fact that automatic speech recognition (ASR) is still a very difficult problem.

The obvious analog to the graphical test of Coates, *et al.* is to synthesize and speak a word and ask the user to spell it. Unfortunately, there are three significant hurdles in the way of this scheme: (1) no keyboard is available for inputting the results, (2) similar-sounding words (homonyms) can have different spellings, and (3) most people are poor spellers. The first of these could be addressed by employing ASR to process the user's spoken response (and hoping that it is good enough to handle this part of the test, but not so good as to be able to defeat the test). The homonym issue could perhaps be resolved by careful word choice when generating the test, or by accepting any legal spelling. The final point, however, seems to be a serious obstacle. Defining a class of words that all users could be expected to spell correctly would be very difficult, if not impossible.

In the case of applications where a keypad is available (*e.g.*, cell phones), a better approach would be to alter the test and have the system say: "Please enter the following digits on your keypad: ..." followed by a short, random digit string. Keep in mind that the speech must be synthesized in a way that ASR is likely to fail the test. This could be accomplished by distorting it via the random variation of key parameters (durations, $f0$ curves, energy levels) and/or by adding "difficult" background noise to the signal.

With some experimentation, this HIP could probably be made to work quite well, although the ability of people to remember and repeat digit sequences may prove to be an issue. The Coates, *et al.* test has an advantage here, as words are not simply random sequences of characters; they make sense to humans. We can resolve this by keeping the digit sequences short (but not too short, as we wish to take advantage of the fact that segmentation is a challenging task in ASR) and repeating the test multiple times, as shown in Figure 1. Hence, there is likely to be an "optimal" length for the random digit strings which could be determined empirically. They should be long enough to make ASR challenging, but short enough that people can easily remember them and key them into a cell phone.

The previous approach depends on the user having access to a keypad. Without one, the user can be asked to repeat the digit sequence by speaking it. This does introduce a possible way for a machine to "cheat," however; it could simply record the test utterance, segment out the intended digit string, and play it back again as its response without having

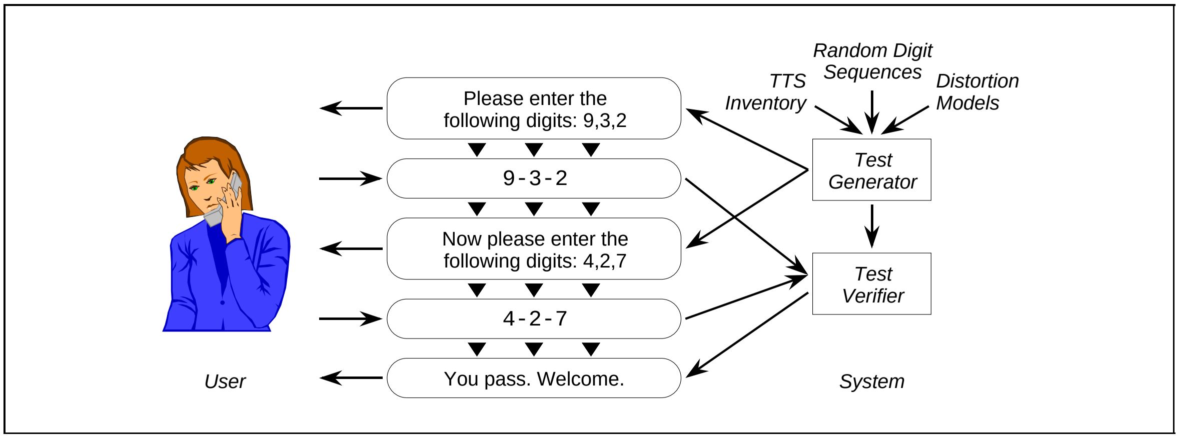

Figure 1: Protocol for a spoken language Human Interactive Proof.

to perform ASR. A safeguard against this attack would be for the computer administering the test to compare the speech signal for the response to that for the original digit string to be sure they are sufficiently different.

Another, perhaps cleverer, solution would be to pose a different kind of challenge:

- "What number comes after (or before) m?"
  where m is a modest-sized integer, *e.g.*, $0 \leq m \leq 1,000$.

- "Which day of the week comes after (or before) X?"
  where X signifies one of the seven days of the week.

Note that each of these "templates" does not represent a single query, but rather a class of questions that could be generated automatically. They are also fundamentally different from the challenge depicted in Figure 1 in that they involve more than recognizing a pattern and repeating it back; they exploit a simple kind of shared domain knowledge, the standard sequences of integers and days.

An overview of a system for performing a spoken HIP is shown in Figure 2. Proceeding from left to right, first a challenge is randomly chosen. Then a carrier phrase is constructed by walking a random path through a grammar designed to express the challenge in natural language terms for the user. The next step is to determine the parameters needed for rendering the text as speech (*e.g.*, durations, pitch contours); again, these values are randomly set within certain ranges. In addition, the acoustic inventory can be selected from among a number of different speakers. Finally, noise (*e.g.*, static, background music, "cocktail party" murmuring) is injected into the signal. The net effect of all of these stages is to introduce an enormous degree of variability that is intended to defeat ASR, but which humans should be able to take in stride.

2 Making the Speech Signal Difficult for Machines

Analogous to the OCR example cited earlier, ASR performance degrades quickly in the presense of noise, music, or any other distortion. We may use a TTS system to generate

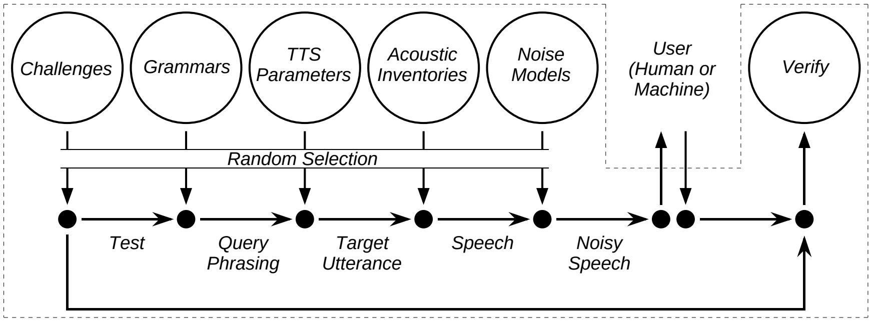

Figure 2: System for generating and verifying a spoken language Human Interactive Proof.

speech and mix it with noise, music, or background speech. The added signal can be tuned so that it is not difficult for humans, but sufficient to cause problems for ASR systems.

Human perception of speech in noisy environments is fairly robust. Normal-hearing listeners need a signal-to-noise ratio (SNR) of approximately 1.5 dB to recognize speech sounds. ASR, on the other hand, typically requires a much more favorable SNR, in the range of 5 to 15 dB. While the results just quoted depend heavily on the testing conditions, there is unquestionably a large gap in the SNR thresholds between humans and machines, suggesting that this is a feature we can use to distinguish one group from the other.

When we speak, we move articulators (tongue, lips, jaw) to change the shape of our vocal tract. This filters the glottal waves and creates the distinctive patterns of resonant frequencies, or formants, of different speech sounds. ASR systems use features derived from formants, such as spectral envelope and cepstrum, to identify speech. Anything that affects this frequency pattern is likely to create difficulties for ASR. For example, adding constant narrow-band noise or a sine wave at the frequency of a formant distorts cepstrum coefficients and will cause problems. Severe-but-constant band-pass filtering yields speech that is intelligible to humans, but considerably harder for machines.

Likewise, humans can bridge missing segments of up to 200 msec when the gaps are filled with white noise. Such gaps are difficult for ASR systems to handle, however. We could, for example, create TTS speech, randomly excise a short chunk and replace it with white noise, and present it to the user for identification. We would expect humans to pass this test and machines to fail.

On another note, speech carries two parallel channels: segmental information (the sounds shaped by the vocal track) and prosody, which includes duration, intonation, and amplitude. The former determines what is being said, while prosody conveys how it is said, including emotion and emphasis. Current ASR systems lack the ability to process prosody, a flaw that could be exploited in designing potential challenges. For example, we might synthesize speech using digit or word strings with randomly placed emphasis, such as “three, four, NINE, seven” or “paper, PENCIL, ruler, eraser,” and ask the listener to identify the emphasized word. ASR is trained to recognize speech sounds, but not prosody.

3 Exploiting Idiosyncrasies in Human Perception

The obvious approach when designing a test to distinguish humans from machines is to identify a task for which machines are known to perform poorly. This is the premise underlying both the graphical test developed by Coates, *et al.*, as well as the spoken language tests we have just described. There is another option, however. It is only important that there be a way to discriminate reliably between humans and machines, not that the latter always fail the test. We can also consider building HIP's where machines are "fooled" into performing better than a human can.

It is well known that vowel quality is primarily determined by spectral information. However, human perception of vowel height is also affected by the distance between $f1$ (the first formant) and $f0$ (pitch, or fundamental frequency). ASR systems train acoustic models primarily by using various representations of the spectral information, but do not incorporate information on $f1-f0$. One effective way to discriminate between humans and machines, then, is to synthesize words with different $f0$ values while keeping the formant structure constant. Under drastically altered $f0$ conditions, humans perceive different vowels, but a machine sees the same vowel. For example, we can synthesize the word "met" with $f0$ values of 100 Hz and 400 Hz. ASR will recognize both stimuli as "met" based on the constant formant relationship. Humans, whose perception is affected by the $f1-f0$ space, will regard the word "met" synthesized with $f0 = 400$ Hz as "mit," while they understand the speech signal with $f0 = 100$ Hz as "met."

4 Conclusions

In this brief overview, we have described several novel approaches for building Human Interactive Proofs for spoken language interfaces. This application raises some interesting questions for TTS research. The goal of intentionally generating speech in a way that makes it unintelligible to ASR while keeping it understandable to humans (or vice versa) is, to our knowledge, a completely new way of framing the text-to-speech problem. We have also suggested that such tests, whether spoken or graphical, do not necessarily have to be hard for the machine and easy for the human; the opposite standpoint is equally valid and perhaps even more intriguing. We are now in the process of beginning to evaluate some of our ideas, and hope to have preliminary experimental results to present at the workshop.

References

[1] A. L. Coates, H. S. Baird, and R. J. Fateman. Pessimal print: A reverse Turing Test. In *Proceedings of the Sixth International Conference on Document Analysis and Recognition*, pages 1154–1158, Seattle, WA, September 2001.

[2] L. von Ahn, M. Blum, J. Langford, and U. Manber. The CAPTCHA project: Telling humans and computers apart (automatically). http://www.captcha.net/.