On the Benefits of Confidence Visualization in Speech Recognition

Keith Vertanen and Per Ola Kristensson
Cavendish Laboratory, University of Cambridge
JJ Thomson Avenue, Cambridge, CB3 0HE, UK
{kv227,pok21}@cam.ac.uk

ABSTRACT

In a typical speech dictation interface, the recognizer's best-guess is displayed as normal, unannotated text. This ignores potentially useful information about the recognizer's confidence in its recognition hypothesis. Using a confidence measure (which itself may sometimes be inaccurate), we investigated providing visual feedback about low-confidence portions of the recognition using shaded, red underlining. An evaluation showed, compared to a baseline without underlining, underlining low-confidence areas did not increase user's speed or accuracy in detecting errors. However, we found that when recognition errors were correctly underlined, they were discovered significantly more often than baseline. Conversely, when errors failed to be underlined, they were discovered less often. Our results indicate confidence visualization can be effective – but only if the confidence measure has high accuracy. Further, since our results show that users tend to trust confidence visualization, designers should be careful in its application if a high accuracy confidence measure is not available.

Author Keywords

Speech recognition, visualization, recognition interfaces.

ACM Classification Keywords

H.5.2. User Interfaces – Natural language. I.2.7. Natural Language Processing – speech recognition and synthesis.

INTRODUCTION

While speech recognition accuracy has improved substantially in recent years, users dictating text to their computer still face occasional recognition errors. These errors must first be detected by the user and then corrected in some manner. Much effort has been directed towards studying how users handle the combined detection and correction process in speech interfaces, e.g. [5].

CHI 2008, April 5–10, 2008, Florence, Italy.

Unlike previous research on confidence visualization (e.g. [1,8,9]), in this paper, we focus on the first part of the correction problem only: finding errors. Detection of errors can be tricky for users as errors made by a recognizer are all valid words in a language. In addition, recognizers may occasionally delete words which can be easy to overlook.

We investigate whether the detection process can be aided by using information available to the recognizer. This is achieved with a confidence score, which reflects the degree of belief a recognizer has in a particular word in the recognition result. Words with low confidence are typically at a higher risk of being an error. We have designed and implemented a speech interface which conveys information about low-confidence words to the user.

Our results show confidence visualization does not overall improve users' ability to detect recognition errors. However, unlike previous work, we found that it was not confidence visualization per se that caused the non-result. Rather, we found that when confidence visualization "did the right thing" and highlighted incorrect recognition results, participants detected significantly more errors when using visualization than without. However, participants also trusted the confidence visualization and tended to miss errors that (incorrectly) had a high confidence.

The rest of this paper is structured as follows. First we give the details of our speech recognizer and describe our confidence visualization method. Second, we describe the evaluation we conducted to assess if confidence visualization benefits users. Third, we present and discuss our results. Last, we discuss related work and conclude.

SYSTEM

Speech Recognizer

We used the CMU Sphinx speech recognizer. We trained a British-English acoustic model using 16 hours of WSJCAM0 data, cross-word triphones, 12 MFCCs plus deltas and delta-deltas, 8 continuous Gaussians/state, and 3K tied-states. We created gender-dependent models using MLLR- and MAP-adaptation with additional MLLR-adaptation performed on audio collected from each participant. Our software combined PortAudio for audio capture, Sphinx-3 for speech decoding, and SRILM for lattice pruning and word confusion network clustering. We trained a trigram language model using: newswire text from

the CSR-III corpus (222M words), Knesser-Ney smoothing, and a 5K-vocab with verbalized commas and periods.

We wanted our system to have a word error rate (WER) similar to what novices might encounter using a modern commercial recognizer. Past user studies reported WERs of 6-11% [6], 7-15% [2], and 15% [4]. As these studies used older recognizers, we did our own testing using Dragon v9. Using 24 novices who enrolled using the "Talking to your computer" text, we found a 7.9% WER on 84 sentences from the WSJ spoke 2 corpus. Thus, we targeted a WER of around 8% for our study. In pre-study testing, our recognizer had a 9.5% WER on a 5K-vocab WSJCAM0 test set, operating at 0.6×realtime. During our experiment to be described shortly, our participants had a WER of 8.5%.

Confidence Visualization

As a measure of confidence, we used the posterior probabilities given by a word confusion network (WCN) [7]. A WCN is a time-ordered sequence of clusters where each cluster contains competing words and their probabilities (Figure 1). The probabilities in a cluster sum to 1. A WCN is built using the time- and phonetic-overlap of a recognizer's word lattice output. WCNs can contain special "delete" words which represent the hypothesis that nothing was said. The best recognition result is found by taking the highest probability hop in each cluster.

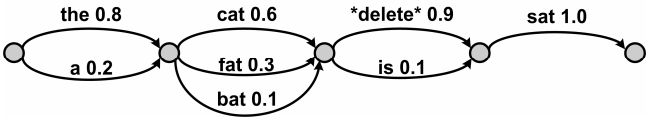

Figure 1. Example word confusion network with 4 clusters and a best recognition result of "the cat sat". Confidence scores for the word / delete-word hypotheses are shown on the arcs.

Visualization of Substitution and Insertion Errors

A *substitution error* is a word mistakenly recognized as another word. An *insertion error* is an extra word which was mistakenly added. Both errors result in visible words in the user's display that should either be deleted (for insertion errors) or replaced (for substitution errors).

Insertion and substitution errors were underlined using a 3-pixel wide red brush. The color was made more intense for words more likely to be an error. The color was calculated as a red-green-blue color triplet using linear interpolation:

$$color = (1, c, c)$$

where c is a word's confidence (between 0.0 and 1.0).

Visualization of Deletion Errors

A *deletion error* occurs when a word is missing from the recognition. Unlike some other confidence measures, WCNs provide confidence scores regarding deletion errors. We developed a novel technique to visualize these deletion errors (see Table 1 for an example).

Deletion errors were visualized as empty rectangular areas at the position where the deleted word would have been in the text. Fitts' law [3] tells us that larger targets are faster to click than smaller (assuming same amplitude). Also, an error with a low confidence is more likely to be a true error. Thus, to make it easier to click on low confidence deletion errors, the width (in pixels) of the space denoting a deletion error was computed as a function of the confidence c:

$$w = 1 + (15(1 - c))$$

The space indicating a deletion error was also underlined in the same manner as substitution and insertion errors.

Error type	Before correction	After correction
Substitution	The fat sat	The cat sat
Insertion	The cat is sat	The cat sat
Deletion	The sat	The cat sat

Table 1. Example errors and their visualization in our system.

EVALUATION

Method

We used a within-subjects experimental design with two conditions: confidence visualization versus a baseline that presented words without any indication of confidence.

Participants and Apparatus

16 volunteer participants were recruited from the university campus (13 men, 3 women). Their ages ranged between 22-33 (mean = 26.6, sd = 2.7). They were paid £5 for participating in a single 45-minute study session. Participants used a Dell laptop with a 15" 1400x1280 screen and wore a Plantronics DSP-400 headset mic.

Material and Procedure

Participants spoke a short paragraph consisting of 1-2 sentences from the set-aside directory of the CSR-III corpus. These sentences were excluded from language model training. The average paragraph length was 20 words, in line with a previous study by Suhm *et al.* [8]. The experiment consisted of two conditions:

1. *Baseline*. Words were presented without confidence visualization.
2. *Visualization*. Confidence scores from the recognizer were visualized to the user.

The starting condition and paragraph order were counterbalanced across participants. Participants first trained the speech recognizer (about 10 mins). Participants then proceeded to their first condition (either *Visualization* or *Baseline* depending on their order). In each condition, participants did 1 practice and 20 trial paragraphs. During the first practice paragraph, an experimenter described how to use the interface. The practice paragraph was excluded from all analysis. After each condition, participants filled in a brief questionnaire. Between the two conditions, participants were given a break (about 5 mins). After the last condition, participants filled in a final questionnaire.

In both conditions, participants were presented with the stimuli paragraph in a textbox. To encourage reading the paragraph before speaking, the paragraph was displayed in teleprompter-style (each character was added to the textbox after a delay). After the entire paragraph was displayed, participants pressed a BEGIN SPEAKING button, which started streaming audio to the recognizer. After finishing speaking, the participant pressed the STOP SPEAKING button.

After a small recognition delay (mean = 2.3s, sd = 1.9s), a beep alerted the participant that recognition was complete and a BEGIN CORRECTING button appeared. If the participant had misspoken, a TRY AGAIN button allowed the paragraph to be dictated again. When the participant pressed the BEGIN CORRECTING button, the stimuli paragraph was hidden and the recognition results were displayed. In the visualization condition, red underlining denoted possible recognition errors (Figure 2). In all other aspects, the interactions in both conditions were identical.

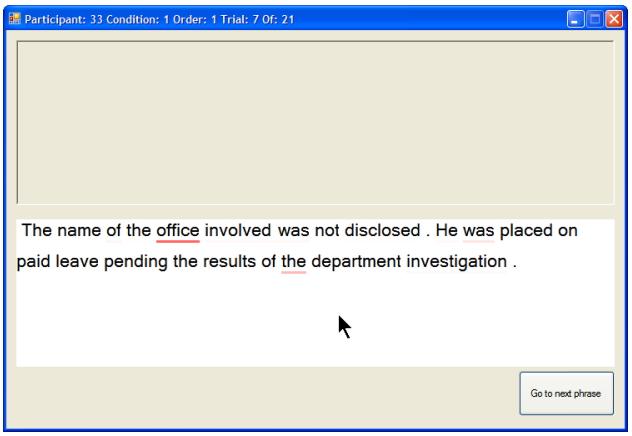

Figure 2. Words with low-confidence are underlined with a shade of red. The more intense the red color, the higher the chance that the word was a recognition error.

The participant was told to quickly and accurately indicate all errors in the recognized text by clicking on them. When an error was clicked, it was automatically corrected. This was possible because the software knew the correct text. If something was clicked that was already correct, it was left unchanged (but the click was logged as a mistake). We emphasize that we used "oracle" knowledge (knowledge of the stimuli paragraph) *only* to perform automatic correction. The recognized text and confidence scores were obtained from real recognition results from the participant's audio. Further, confidence visualization did not use any oracle knowledge and relied solely on the recognizer's output.

We did not allow manual correction of errors since we were interested in whether visualization enabled faster error detection. In order to reliably measure "detection time", we wanted to avoid the possible confound that while the participant manually corrected an error (e.g. by re-positioning the text caret and erasing the incorrect word) the participant may discover other errors. This extra manual editing time would have been impossible to separate out from the time it took to actually discover all the errors.

After the participant corrected a paragraph to the best of his or her ability, the participant pressed the FINISHED CORRECTING button and the next paragraph was initiated.

Results

Detection Time

Detection time was defined as the duration between the user pressing BEGIN CORRECTING and FINISHED CORRECTING. The mean response time was 9704 ms in the visualization condition and 9010 ms in the baseline condition. Repeated measures analysis of variance showed that the result was not significant ($F_{1, 15} = .846$, $p = .372$).

Incorrect Correction Attempts

Before the evaluation, we were concerned that participants might react to the colorfully underlined words by clicking on all of them to remove the underlining. Such behavior would have caused an inflated number of incorrect corrections in the visualization condition. Surprisingly, users committed more incorrect clicks in the baseline condition (mean = 5.4, sd = 6.6) than in the visualization condition (mean = 4.1, sd = 4.8). This difference was not significant ($F_{1, 16} = 1.176$, $p = .295$). Hence, participants were careful and did not click on all underlined items.

Error Reduction Rate

Error reduction rate was measured as the number of recognition errors the user corrected, divided by the number of total recognition errors. For example, if the recognizer made 40 errors and the participant corrected 30, the error reduction rate was 75%. The significances of mean error reduction rates between the conditions were determined by repeated measures analysis of variance.

For all errors, in the visualization condition, participants reduced errors by 84% (sd = 10.3%), in comparison to 81% (sd = 7.1%) in the baseline condition. The difference in error reduction was not significant ($F_{1, 15} = .792$, $p = .388$). Overall, confidence visualization did not improve participants' ability to discover errors.

However, we were curious why visualization did not help. A possible problem is how well we detected real recognizer mistakes while avoiding falsely flagging correct words. In our system, a confidence score > 0.9 resulted in so pale an underlining as to be almost imperceptible. We therefore split our errors into two sets: visibly underlined errors (confidence ≤ 0.9), and not visibly underlined errors (confidence > 0.9). At this threshold, our false accept rate was 3.4% (non-underlined words that were errors) and our false reject rate was 7.7% (underlined words that were correct).

For errors with confidence ≤ 0.9 (errors that were actually visible to participants in condition 2), participants' mean error reduction rate was 81% (sd = 9.6%) in the baseline and 92% (sd = 8.9%) with visualization. The 11% increase in participants' ability to reduce errors in the visualization condition was statistically significant ($F_{1,15} = 15.384$, $p =$

.001). This means that confidence visualization helped participants detect more low-confidence errors than in the baseline. However, this win must be a loss somewhere else, since we found, overall, visualization did not improve participants' error reduction rate. Indeed, for errors with confidence > 0.9, participants' error reduction rate was 82% in the baseline but only 71% in the visualization condition. Although this result was not significant ($F_{1, 15} = 3.404$, $p = .085$), the dramatic difference does explain why confidence visualization did not improve overall error reduction rates.

We draw three conclusions. First, confidence visualization did work in the sense that participants took advantage of underlining to detect recognition errors. Second, it is plausible participants trusted confidence visualization and stopped actively verifying non-highlighted words, inflating the number of undetected errors that had too high a confidence to be underlined. Third, it is likely that visualization using highly accurate confidence scores will significantly help users detect errors overall.

RELATED WORK

In Suhm *et al.* [8], they found no difference in the corrected words-per-minute achieved using an interface with confidence visualization and one without. Similarly, our results showed no difference in the time it took to detect errors with and without visualization. However, there are several differences between their study and ours. First, their measured times included speech or pen correction. In contrast, our study used an oracle to instantly correct errors users detected. This allowed us to measure error detection time – the time it takes to proofread recognized text and indicate errors. Second, their 25% WER on their participants' original speech was much higher than our WER of 8.5%. We suspect that confidence visualization becomes increasingly useless as WER increases due to the large number of things highlighted. With such a high WER it is likely that users would ignore confidence visualization and carefully check the entire recognition result for errors.

In Burke *et al.* [1], recognition was performed on voice mail messages with the results shaded using a WCN-based confidence measure. Their objective was to enable users to quickly read a voice mail summary, ignoring low-confidence words. In contrast, our goal was to focus users' attention on potential errors. Additionally, they never treated confidence visualization as an independent variable.

In Vermuri *et al.* [9], an audio playback interface was tested using recognition results with and without confidence visualization. No difference in users' comprehension rate was found. Again, their goal of aiding comprehension was different from our goal of facilitating error detection.

CONCLUSIONS

We have presented a system capable of visualizing all recognition error types: deletion, insertion and substitution. We used our system to investigate if confidence visualization helps users find errors in a dictation task. An evaluation of our system shows that confidence-based underlining did not improve users' overall speed or accuracy at finding errors. However, we found that when errors were correctly underlined in the visualization condition, they were found significantly more often than in the baseline condition. Conversely, when errors failed to be underlined, they were detected less often than baseline. This suggests that confidence visualization must be used cautiously. A poor confidence measure may distract attention away from legitimate errors, leading to an actual increase in user errors. Further, a good confidence measure focuses attention on errors that would otherwise go unnoticed, leading to a decrease in user errors.

ACKNOWLEDGEMENTS

We thank Alan Blackwell and David MacKay for helpful discussions. Keith Vertanen was supported by a scholarship from the Cavendish Laboratory. Per Ola Kristensson was supported by Nokia, and Ericsson Research Foundation.

REFERENCES

1. Burke, M., Amento, B. and Isenhour, P. 2006. Error correction of voice mail transcripts in SCANMail. *CHI 2006,* ACM Press: 339-348.
2. Devine, E., Gaehde, S. and Curtis, A. 2000. Comparative Evaluation of Three Continuous Speech Recognition Software Packages in the Generation of Medical Reports. *Journal of the American Medical Informatics Assoc*., 7:462-468.
3. Fitts, P. 1954. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology* 47(6): 381-391.
4. Horstmann H. 2004. Usage, performance, and satisfaction outcomes for experienced users of automatic speech recognition. *Journal of Rehabilitation Research and Development*, 41(5): 739-754.
5. Karat, C., Halverson, C., Horn, D. and Karat, J. 1999. Patterns of entry and correction in large vocabulary continuous speech recognition systems. *CHI 1999*, ACM Press: 568-575.
6. Karat, J., Horn, D., Halverson, C., and Karat, C. 2000. Overcoming unusability: developing efficient strategies in speech recognition systems. *CHI 2000*, ACM Press:141-142.
7. Hakkani-Tür, D., Béchet, F., Riccardi, G., Tur, G. 2006. Beyond ASR 1-best: Using word confusion networks in spoken language understanding. *Journal of Computer Speech and Language* 20(4): 495-514.
8. Suhm, B., Myers, B. and Waibel, A. 2001. Multimodal error correction for speech user interfaces. *ACM Trans. on Computer-Human Interaction* 8(1): 60-98.
9. Vermuri, S., DeCamp, P., Bender, W., and Schmandt, C. Improving speech playback using time-compression and speech recognition. *CHI 2004*, ACM Press: 295-302.