DOI:0 0.0000/000

Scribe: Deep Integration of Human and Machine Intelligence to Caption Speech in Real Time

By Walter S. Lasecki, Christopher D. Miller, Iftekhar Naim, Raja Kushalnagar, Adam Sadilek, Daniel Gildea, and Jeffrey P. Bigham

Abstract

Quickly converting speech to text allows deaf and hard of hearing people to interactively follow along with live speech. Doing so reliably requires a combination of perception, understanding, and speed that neither humans nor machines possess alone. In this article, we discuss how our Scribe system combines human labor and machine intelligence in real time to reliably convert speech to text with less than 4s latency. To achieve this speed while maintaining high accuracy, Scribe integrates automated assistance in two ways. First, its user interface directs workers to different portions of the audio stream, slows down the portion they are asked to type, and adaptively determines segment length based on typing speed. Second, it automatically merges the partial input of multiple workers into a single transcript using a custom version of multiple-sequence alignment. Scribe illustrates the broad potential for deeply interleaving human labor and machine intelligence to provide intelligent interactive services that neither can currently achieve alone.

1. INTRODUCTION AND BACKGROUND

Real-time captioning converts speech to text in under 5s to provide access to live speech content for deaf and hard of hearing (DHH) people in classrooms, meetings, casual conversation, and other events. Current options are severely limited because they either require highly-skilled professional captionists whose services are expensive and not available on demand, or use automatic speech recognition (ASR) which produces unacceptable error rates in many real-world situations.[10] We present an approach that leverages groups of non-expert captionists (people who can hear and type, but are not specially trained stenographers) to collectively caption speech in real-time, and explore this new approach via *Scribe*, our end-to-end system allowing on-demand real-time captioning for live events.[19] Scribe integrates human and machine intelligence in real time to reliably caption speech at natural speaking rates.

The Word Health Organization (WHO) estimates that around 5% of the world population, that is, 360 million people, have disabling hearing loss.[32] They struggle to understand speech and benefit from visual input. Some combine lip-reading with listening, while others primarily watch visual translations of aural information, such as sign language interpreters or real-time typists. While visual access to spoken material can be achieved through sign language interpreters, many DHH people do not know sign language. This is particularly true of the large (and increasing) number of DHH people who lost their hearing later in life, which includes one third of people over 65.[12] Captioning may also be preferred by some to sign language interpreting for technical domains because it does not involve translating from the spoken language to the sign language, but rather transliterating an aural representation to a written one. Finally, like captionists, sign language interpreters are also expensive and difficult to schedule.

People learn to listen and speak at a natural rate of 120–180 words per minute (WPM).[17] They acquire this skill effortlessly without direct instruction while growing up or being immersed in daily linguistic interaction, unlike text generation, which is a trained skill that averages 60–80 WPM for both handwriting[29] and typing.[14] Professional captionists (stenographers) can keep up with most speakers and provide captions that are accurate (95%+) and real-time (within a few seconds). But they are not on-demand (need to be pre-booked for at least an hour), and are expensive ($120–200 per hour).[30] As a result, professionals usually cannot provide access for last minute lectures or other events, or for unpredictable and ephemeral learning opportunities, such as conversations with peers after class.

Automatic speech recognition (ASR) is inexpensive and available on-demand, but its low accuracy in many real settings makes it unusable. For example, ASR accuracy drops below 50% when it is not trained on the speaker, captioning multiple speakers, and/or when not using a high-quality microphone located close to the speaker.[3, 6] Both ASR and the software used to assist real-time captionists often make errors that can change the meaning of the original speech. As DHH people use context to compensate for errors, they often have trouble following the speaker.[6]

Our approach is to combine the efforts of multiple non-expert captionists. Because these non-expert captionists can be drawn from more diverse labor pools than professional captionists, they are more affordable and more easily available on demand. Recent work has shown, for instance, that workers on Mechanical Turk can be recruited within a few seconds,[1, 2, 11] and engaged in continuous tasks.[21, 24, 25, 28] Recruiting

Sign languages, such as American Sign Language (ASL) are not simply codes for an aural language, but rather entirely different languages with their own vocabulary, grammar, and syntax.

The original version of this paper is entitled "Real-Time Captioning by Groups of Non-Experts" and was published in (UIST, 10/2012, ACM.)

from a broader pool allows workers to be selectively chosen for their expertise not in captioning but in the technical areas covered in a lecture. While professional stenographers are able to type faster and more accurately than most crowd workers, they are not necessarily experts in the field they are captioning, which can lead to mistakes that distort the meaning of transcripts of technical talks.[30] Scribe allows student workers to serve as non-expert captionists for $8–12 per hour (a typical work-study pay rate). Therefore, we could hire several students for much less than the cost of one professional captionist.

Scribe makes it possible for non-experts to collaboratively caption speech in real time by providing automated assistance in two ways. First, it assists captionists by making the task easier for each individual. It directs each worker to type only part of the stream audio, it slows down the portion they are asked to type so they can more easily keep up, and it adaptively determines the segment length based on each individual's typing speed. Second, it solves the coordination problem for workers by automatically merging the partial input of multiple workers into a single transcript using a custom version of multiple-sequence alignment.

Because captions are dynamic, readers spend far more mental effort reading real-time captions compared to static text. Also, regardless of method, captions require users to absorb information that is otherwise consumed via two senses (vision and hearing) via only one (vision). In classroom settings, this can be particularly common, with content appearing on the board and being referenced in speech. The effort required to track both the captions and the material they pertain to simultaneously is one possible reason why deaf students often lag behind their hearing peers, even with the best accomodations.[26] To address these issues, we also explore how captions can be best presented to users,[16] and show that controlling bookmarks in caption playback can even increase comprehension.[22]

This paper outlines the following contributions:

- Scribe, an end-to-end system that has advantages over current state-of-the-art solutions in terms of availability, cost, and accuracy.
- Evidence that non-experts can collectively cover speech at rates similar to or above that of a professional.
- Methods for quickly merging multiple partial captions to create a single, accurate stream of final results.
- Evidence that Scribe can produce transcripts that both cover more of the input signal and are more accurate than either ASR or any single constituent worker.
- The idea of automatically combining the real-time efforts of dynamic groups of workers to outperform individuals on human performance tasks.

2. CURRENT APPROACHES

We first overview current approaches for real-time captioning, introduce our data set, and define the evaluation metrics used in this paper. Methods for producing real-time captioning services come in three main varieties:

Computer-Aided Real-time Transcription (CART): CART is the most reliable real-time captioning service, but is also the most expensive. Trained stenographers type in shorthand on a "steno" keyboard that maps multiple key presses to phonemes that are expanded to verbatim text. Stenography requires 2–3 years of training to consistently keep up with natural speaking rates that average 141 WPM and can reach 231 WPM.[13]

Non-Verbatim Captioning: In response to the cost of CART, computer-based macro expansion services like C-Print were introduced.[30] C-Print captionists need less training, and generally charge around $60 an hour. However, they normally cannot type as fast as the average speaker's pace, and cannot produce a verbatim transcript. Scribe employs captionists with no training and compensates for slower typing speeds and lower accuracy by combining the efforts of multiple parallel captionists.

Automated Speech Recognition: ASR works well in ideal situations with high-quality audio equipment, but degrades quickly in real-world settings. ASR is has difficulty recognizing domain-specific jargon, and adapts poorly to changes, such as when the speaker has a cold.[6] ASR systems can require substantial computing power and special audio equipment to work well, which lowers availability. In our experiments, we used Dragon Naturally Speaking 11.5 for Windows.

Re-speaking: In settings where trained typists are not common (such as in the U.K.), alternatives have arisen. In re-speaking, a person listens to the speech and enunciates clearly into a high-quality microphone, often in a special environment, so that ASR can produce captions with high accuracy. This approach is generally accurate, but cannot produce punctuation, and has considerable delay. Additionally, re-speaking still requires extensive training, since simultaneous speaking and listening is challenging.

3. LEGION: SCRIBE

Scribe gives users on-demand access to real-time captioning from groups of non-experts via their laptop or mobile devices (Figure 1). When a user starts Scribe, it immediately begins recruiting workers to the task from Mechanical Turk, or a pool of volunteer workers, using LegionTools.[11,20] When users want to begin captioning audio, they press the start button, which forwards audio to Flash Media Server (FMS) and signals the Scribe server to begin captioning.

Workers are presented with a text input interface designed to encourage real-time answers and increase global coverage (Figure 2). A display shows workers their rewards for contributing in the form of both money and points. In our experiments, we paid workers $0.005 for every word the system thought was correct. As workers type, their input is forwarded to an *input combiner* on the Scribe server. The input combiner is modular to accommodate different implementations without needing to modify Scribe. The combiner and interface are discussed in more detail later in this article.

The user interface for Scribe presents streaming text within a collaborative editing framework (see Figure 3). Scribe's interface masks the staggered and delayed format

Figure 1: Scribe allows users to caption audio on their mobile device. The audio is sent to multiple amateur captionists who use Scribe's Web-based interface to caption as much of the audio as they can in real-time. These partial captions are sent to our server to be merged into a final output stream, which is then forwarded back to the user's mobile device. Crowd workers are optionally recruited to edit the captions after they have been merged.

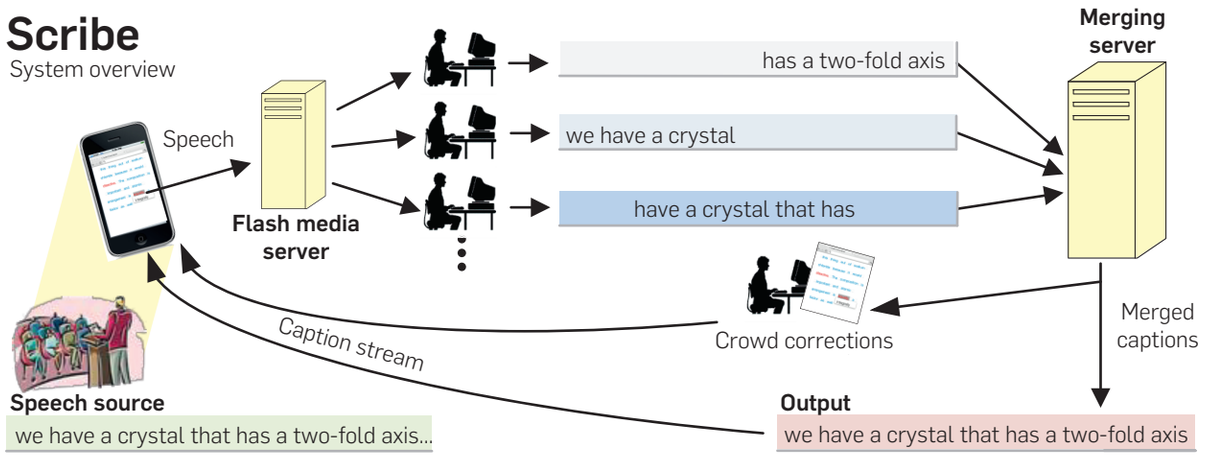


Figure 2: The original worker interface encourages captionists to type quickly by locking in words soon after they are typed. To encourage coverage of specific segments, visual and audio cues are presented, the volume is reduced during off periods, and rewards are increased during these periods.

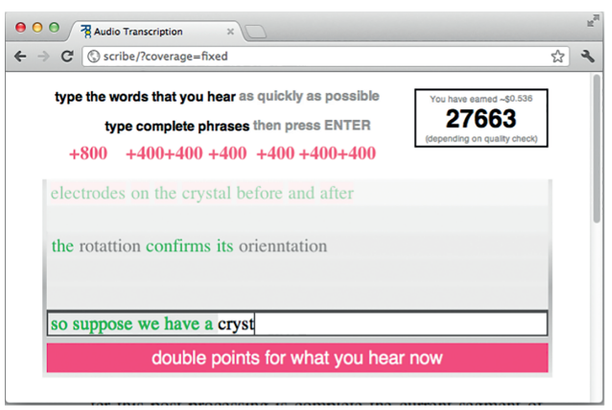
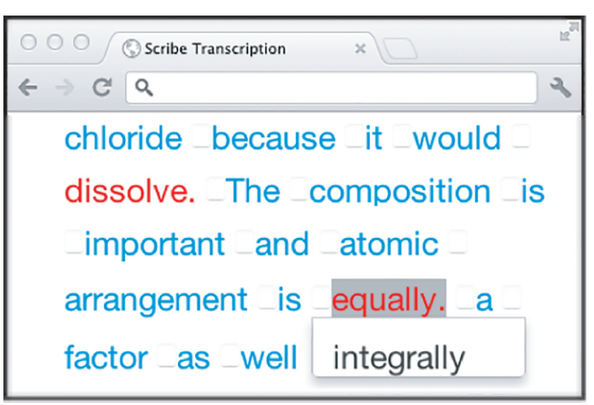


Figure 3: The Web-based interface that shows users the live caption stream returned by Scribe.



of real-time captions with a more natural flow that mimics writing. In doing this, the interface presents the merged inputs from the crowd workers via a dynamically updating Web page, and allows users to focus on reading, instead of tracking changes. We have also developed methods for letting users have more control over their own caption playback, which can improve comprehension.[22] When users are done, pressing stop will end the audio stream, but lets workers complete their current transcription task. Workers are asked to continue working on other audio for a time to keep them active so that response time is reduced if users need to resume captioning.

Though this article focuses on captioning speech from a single person, Scribe can handle dialogues using automated speaker segmentation techniques. We use a standard convolution-based kernel method to first identify distinct segments in a waveform. We then use a one-class support vector machine (SVM) to classify each segment and assign a speaker ID.[15] Prior work has shown such segmentation techniques to be accurate even in the presence of severe noise, such as when talking on a cell phone while driving.[12] The segmentation allows us to *decompose* a dialogue in real-time, then caption each part individually, without burdening workers with the need to determine and annotate which person is currently speaking.

Our solution to the transcription problem is two-fold. First, we designed an interface that facilitates real-time captioning by non-experts and encourages covering the entire audio signal. Second, we developed algorithms for merging partial captions to form one final output stream. The interface and algorithm have been developed to address these problems jointly. For instance, because determining where each word in a partial caption fits into the final transcript is difficult, we designed the interface to encourage workers to type continuous segments during specified periods. In the following sections, we detail the co-evolution of the worker interface and algorithm for merging partial captions in order to form a final transcript.

4. COORDINATING CAPTIONISTS

Scribe's non-expert captioning interface allows contributors to hear an audio stream of the speaker(s), and provide captions with a simple user interface (UI) (Figure 2). Captionists are instructed to type as much as they can, but are under no pressure to type everything they hear. If they are able, workers are asked to separate contiguous sequences of words by pressing enter. Knowing which word sequences are likely to be contiguous can help later when recombining the partial captions from multiple captionists.

To encourage real-time entry of captions, the interface "locks in" words a short time after they are typed (500ms). New words are identified when the captionist types a space after the word, and are sent to the server. The delay is added to allow workers to correct their input while adding as little additional latency as possible to it. When the captionist presses enter (or following a 2s timeout during which they have not typed anything), the line is confirmed and animates upward. During the 10–15s trip to the top of the display (depending on settings), words that Scribe determines were entered correctly (based on either spell-checking or overlap with another worker) are colored green. When the line reaches the top, a point score is calculated for each word based on its length and whether it has been determined to be correct.

To recover the true speech, non-expert captions must *cover* all of the words spoken. A primary reason why the partial transcriptions may not fully cover the true signal relates to *saliency*, which is defined in a linguistic context as "that quality which determines how semantic material is distributed within a sentence or discourse, in terms of the relative emphasis which is placed on its various parts".[7] Numerous factors influence what is salient, and so it is likely to be difficult to detect automatically. Instead, we inject artificial saliency adjustments by systematically varying the volume of the audio signal that captionists hear. Scribe's captionist interface is able to vary the volume over a given a period with an assigned offset. It also displays visual reminders of the period to further reinforce this notion.

Initially, we tried dividing the audio signal into segments that we gave to individual workers. We found several problems with this approach. First, workers tended to take longer to provide their transcriptions as it took them some time to get into the flow of the audio. A continuous stream avoids this problem. Second, the interface seemed to encourage workers to favor quality over speed, whereas streaming content reminds workers of the real-time nature of the task. The continuous interface was designed in an iterative process involving tests with 57 remote and local users with a range of backgrounds and typing abilities. These tests showed that workers tended to provide chains of words rather than disjoint words, and needed to be informed of the motivations behind aspects of the interface to use them properly.

A non-obvious question is what the period of the volume changes should be. In our experiments, we chose to play the audio at regular volume for 4s and then at a lower volume for 6s. This seems to work well in practice, but it is likely that it is not ideal for everyone (discussed below). Our experience suggests that keeping the *in* period short is preferable even when a particular worker was able to type more than the period because the latency of a worker's input tended to go up as they typed more consecutive words.

5. IMPROVING HUMAN PERFORMANCE

Even when workers are directed to small, specific portions of the audio, the resulting partial captions are not perfect. This is due to several factors, including bursts of increased speaking rates being common, and workers mis-hearing some content due to a particular accent or audio disruption. To make the task easier for workers, we created *TimeWarp,*[23] which allows each worker to type what they hear in clips with a lower playback rate, while still keeping up with real time *and* maintaining context from content they are not responsible for.

5.1. Warping time

TimeWarp manages this by balancing the play speed during *in* periods, where workers are expected to caption the audio and the playback speed is reduced, and *out* periods, where workers listen to the audio and the playback speed is increased. A *cycle* is one *in* period followed by an *out* period. At the beginning of each cycle, the worker's position in the audio is aligned with the real-time stream. To do this, we first need to select the number of different sets of workers N that will be used in order to partition the stream. We call the length of the *in* period P_i, the length of the *out* period P_o and the play speed reduction factor r. Therefore, the playback rate during *in* periods is $\frac{1}{r}$. The amount of the real-time stream that gets buffered while playing at the reduced speed is compensated for by an increased playback speed of $\frac{N-1}{N-r}$ during *out* periods. The result is that the cycle time of the modified stream equals the cycle time of the unmodified stream.

To set the length of P_i for our experiments, we conducted preliminary studies with 17 workers drawn from Mechanical Turk. We found that their mean typing speed was 42.8 WPM on a similar real-time captioning task. We also found that a worker could type *at most* 8 words in a row on average before the per-word latency exceeded 8s (our upper bound on acceptable latency). Since the mean speaking rate is around 150 WPM,[13] workers will hear 8 words in roughly 3.2s, with an entry time of roughly 8s from the last word spoken. We used this to set $P_i = 3.25s$, $P_o = 9.75s$, and $N = 4$. We chose $r = 2$ in our tests so that the playback speed would be $\frac{1}{2} = 0.5$ times for *in* periods, and the play speed for *out* periods is $\frac{N-1}{N-r} = \frac{3}{2} = 1.5$ times.

To speed up and slow down the play speed of content being provided to workers without changing the pitch (which would make the content more difficult to understand for the worker), we use the Waveform Similarity Based Overlap and Add (WSOLA) algorithm.[4] WSOLA works by dividing the signal into small segments, then either skipping (to increase play speed) or adding (to decrease play speed) content, and finally stitching these segments back together. To reduce the number of sound artifacts, WSOLA finds overlap points with similar wave forms, then gradually transitions between sequences during these overlap periods.

5.2. Integrating ASR into crowd captioning

Combining ASR into human captioning workflows can also help improve captioning performance. By using the suggestions from an ASR system to provide an initial "baseline"

answer that crowd workers can correct, we can reduce latency. However, above an error rate of $\geq \sim 30\%$ error, the ASR input actually increases latency because of the cost of finding and repairing mistakes.[9] The opposite integration is also possible: by using sparse human input to provide corrections to the word lattice of an ASR system, it is possible to reduce the error rate.[8]

6. AGGREGATING PARTIAL CAPTIONS

The problem of aligning and aggregating multiple partial transcripts can be mapped to the well-studied Multiple Sequence Alignment (MSA) problem. The basic formulation of the problem involves some number of ordered sequences that include at least some similar elements (coming from the same "dictionary" of possible terms plus a "gap" term). Finding the alignment that minimizes total distance between all pairs of sequences is a non-trivial problem because, in the worst case, all possible alignments of the content of each sequence—including all possible spaces containing a gap term—may need to be explored. This optimization problem has been shown to be NP-complete,[31] and exact algorithms have time complexity that is exponential in the number of sequences. As a result, it is often necessary to apply heuristic approximations to perform MSA with in a reasonable amount of time.

In practice, MSA is a well-studied problem in the bioinformatics literature that has long been used in aligning genome sequences, but also has applications in approximate text matching for information retrieval, and in many other domains. Tools like MUSCLE Edgar[5] provide extremely powerful solvers for MSA problems. Accordingly, our approach is to formulate our text-matching problem as MSA.

6.1. Progressive alignment algorithms

Most MSA algorithms for biological sequences follow a progressive alignment strategy that first performs pairwise alignment among the sequences, and then merges sequences progressively according to a decreasing order of pairwise similarity. Due to the sequential merging strategy, progressive alignment algorithms cannot recover from the errors made in the earlier iterations, and typically do not work well for the caption alignment task.

6.2. Graph-based alignment

We first explored a graph-based incremental algorithm to combine partial captions on the fly.[19] The aggregation algorithm incrementally builds a chain graph, where each node represents a set of equivalent words entered by the workers, and the links between nodes are adjusted according to the order of the input words. A greedy search is performed to identify the path with the highest confidence, based on worker input and an n-gram language model. The algorithm is designed to be used online, and hence has high speed and low latency. However, due to the incremental nature of the algorithm and the lack of a principled objective function, it is not guaranteed to find the globally optimal alignment for the captions.

6.3. Weighted A* search algorithm

We next developed a weighted A* search based MSA algorithm to efficiently align the partial captions.[27] To do this, we formulate MSA as graph-traversal over a specialized lattice. Our search algorithm then takes each node as a state, allowing us to estimates the cost function $g(n)$ and the heuristic function $h(n)$ for each state.

At each step of the A* search algorithm, the node with the smallest evaluation function is extracted from the priority queue Q and expanded by one edge. This is repeated until a full alignment is produced (the goal state). While weighted A* significantly speeds the search for the best alignment, it is still too slow for very long sequences. To counteract this, we use fixed-size time windows to scope the exploration to the most-likely paths.

7. EXPERIMENTAL RESULTS

We have tested our system with non-expert captionists drawn from both local and remote crowds. As a data set, we used lectures freely available from MIT OpenCourseWare. These lectures were chosen because one of the main goals of Scribe is to provide captions for classroom activities, and because the recording of the lectures roughly matches our target as well—there is a microphone in the room that often captures multiple speakers, for example, students asking questions. We chose four 5 min segments that contained speech from courses in electrical engineering and chemistry, and had them professionally transcribed at a cost of $1.75 per minute. Despite the high cost, we found a number of errors and omissions. We corrected these to obtain a completely accurate baseline.

7.1. Core system study results

Our study used 20 local participants. Each participant captioned 23 min of aural speech over a period of approximately 30 min. Participants first took a standard typing test and averaged a typing rate of 77.0 WPM (*SD*=15.8) with 2.05% average error (*SD*=2.31%). We then introduced participants to the real-time captioning interface, and had them caption a 3 min clip using it. Participants were then asked to caption the four 5 min clips, two of which were selected to contain saliency adjustments. We measure coverage (recall within a 10s per-word time bound), precision, and WER.

We found that saliency adjustment made a significant difference on coverage ranges. For the electrical engineering clip, the difference was 54.7% (*SD*=9.4%) for words in the selected periods as compared to only 23.3% (*SD*=6.8%) for words outside of those periods. For the chemistry clips, the difference was 50.4% (*SD*=9.2%) of words appearing inside the highlighted period as compared to 15.4% (*SD*=4.3%) of words outside of the period.

To see if workers on Mechanical Turk could complete this task effectively—which would open up a large new set of workers who are available on-demand—we recruited a crowd to caption the four clips (20 min of speech). Our tasks paid $0.05 and workers could make an additional $0.002 bonus per word. We provided workers with a 40s instructional video to beign with. In total, 18 workers participated, collectively achieving 78.0% coverage. The average coverage over just three workers was 59.7% (*SD*=10.9%), suggesting we could be conservative in recruiting workers and cover much of the input signal.

http://ocw.mit.edu/courses/.

In our tests, workers achieved an average of 29.0% coverage, ASR achieved 32.3% coverage, CART achieved 88.5% coverage and Scribe reached 74% out of a possible 93.2% coverage using 10 workers (Figure 4). Collectively, workers had an average latency of 2.89 significantly improving on CART's latency of 4.38s. For this example, we tuned our combiner to balance coverage and precision (Figure 5), getting an average of 66% and 80.3% respectively. As expected, CART outperforms the other approaches. However, our combiner presents a clear improvement over both ASR and a single worker.

7.2. Improved combiner results

We further improved alignment accuracy by applying a novel weighted-A* MSA algorithm.[27] To test this, we used the same four 5 min long audio clips as before. We tested three configurations of our algorithm: (1) no agreement needed with a 15s sliding window, (2) two-person agreement needed with a 10s window, and (3) two-person agreement needed with a 15s window. We compare the results from these three configurations to our original graph-based method, and to the MUSCLE package (Figure 6).

The with agreement and a 15s window (the best performing setting), our algorithm achieves 57.4% average (1-WER) accuracy, providing 29.6% improvement with respect to the graph-based system (average accuracy 42.6%), and 35.4% improvement with respect to the MUSCLE-based MSA system (average accuracy 41.9%). On the same set of audio clips, we obtained 36.6% accuracy using ASR (Dragon Naturally Speaking, version 11.5 for Windows), which is worse than all the crowd-powered approaches. We intentionally did not optimize the ASR for the speaker or acoustics, since DHH students would also not be able to do this in realistic settings.

7.3. TimeWarp results

To evaluate *TimeWarp*, we ran two studies that asked participants to caption a 2.5 min (12 captioning cycles) lecture clip. Again, we ran our experiments with both local participants and workers recruited from Mechanical Turk. Tests were divided into two conditions: time warping on or off, and were randomized across four possible time offsets: 0s, 3.25s, 6.5s, 9.75s.

Local participants were again generally proficient (but non-expert) typists and had time to acquaint themselves with the system, which may better approximate student employees captioning a classroom lecture. We recruited 24 volunteers (mostly students) and had them practice with our baseline interface before using the time warp interface. Each worker was asked to complete two trials, one with *TimeWarp* and one without, in a random order.

We also recruited 139 Mechanical Turk workers, who were allowed to complete at most two tasks and were randomly routed to each condition (providing 257 total responses). Since Mechanical Turk often contains low quality (or even malicious workers),[18] we first removed inputs which got less than 10% coverage or precision or were outliers more than 2σ from the mean. A total of 206 tasks were approved by this quick check. Task payment amounts were the same as for our studies described above.

Our student captionists were able to caption a majority of the content well even without *TimeWarp*. The mean coverage from all 48 trials was 70.23% and the mean precision was 70.71%, compared to the 50.83% coverage and 62.23% precision for workers drawn from Mechanical Turk. For student captionists, total coverage went up 2.02%, from 69.54% to 70.95%, and precision went up by 2.56% from 69.84% to 71.63%, but neither of these differences were detectably significant. However, there was a significant improvement in mean latency per word, which improved 22.46% from 4.34s to 3.36s ($t(df) = 2.78$, $p < .01$). Mechanical Turk workers' mean coverage (Figure 7) increased 11.39% ($t(df) = 2.19$, $p < .05$), precision increased 12.61% ($t(df) = 3.90$, $p < .001$), and latency was reduced by 16.77% ($t(df) = 5.41$, $p < .001$).

Figure 4: Optimal coverage reaches nearly 80% when combining the input of four workers, and nearly 95% with all 10 workers, showing captioning audio in real-time with non-experts is feasible.

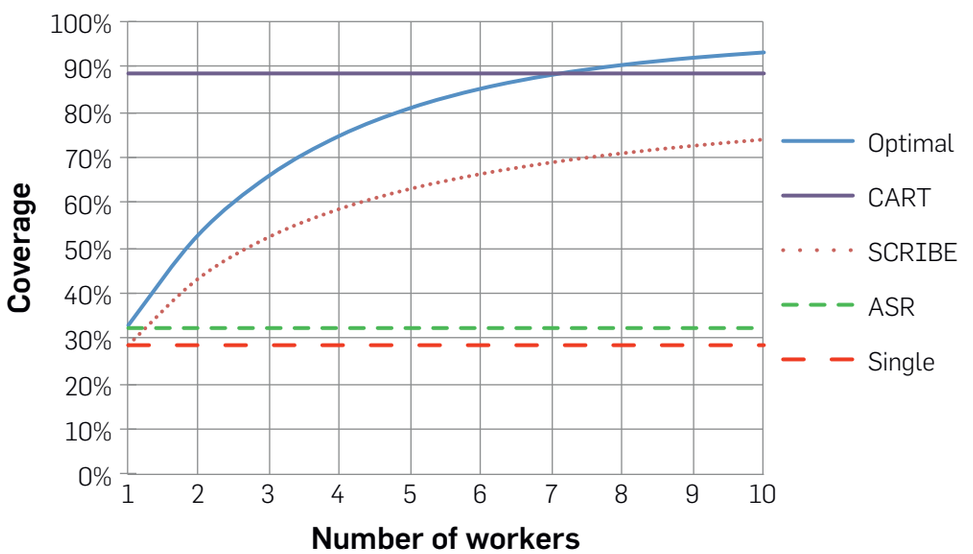


Figure 5: Precision-coverage curves for the electrical engineering (EE) and chemistry (Chem) lectures using different combiner parameters with 10 workers. In general, increasing coverage reduces accuracy.

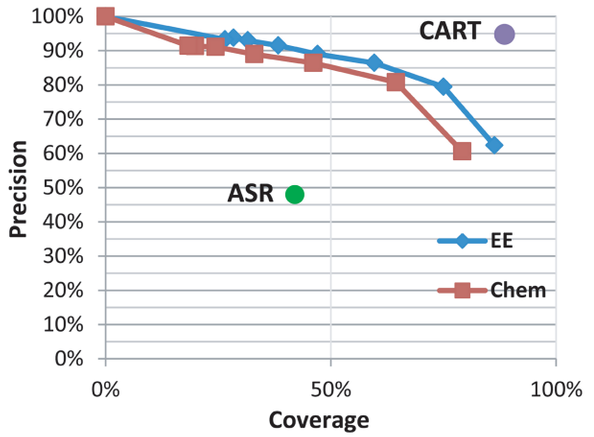


Figure 6: Evaluation of different systems on using (1-WER) as an accuracy measure (higher is better).

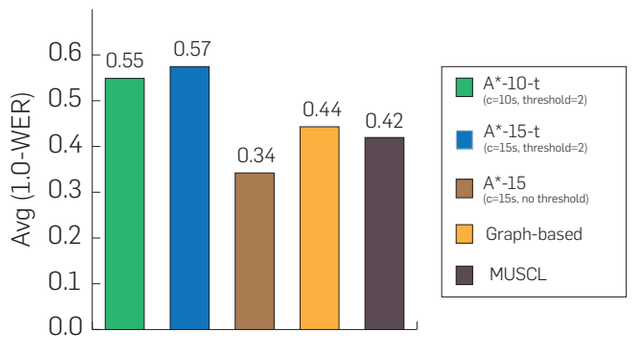
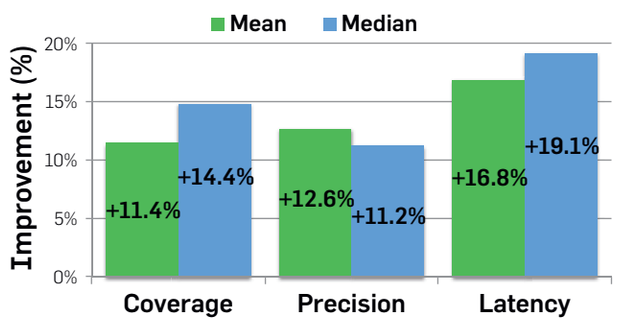


Figure 7: Relative improvement from no warp to warp conditions in terms of mean and median values of coverage, precision, and latency. We expected coverage and precision to improve. Shorter latency was unexpected, but resulted from workers being able to consistently type along with the audio instead of having to remember and go back as the speech outpaced their typing.



8. CONCLUSION AND FUTURE WORK

Scribe is the first system capable of making reliable, affordable captions available on-demand to deaf and hard of hearing users. Scribe has allowed us to explore further issues related to how real-time captions can be made more useful to end users. For example, when captions are used, we have shown that students' comprehension of instructional material significantly improves when they have the ability to control when the captions play, and track their position so that they are not overwhelmed by using one sensory channel to absorb content that is designed to be split between both vision and hearing. To help address this problem, we built a tool that lets students highlight or pause at the last position they read before looking away from the captions to view other visual content.[22]

While we have discussed how automation can be used to effectively mediate human caption generation, advances in ASR technologies can aid Scribe as well. By including ASR systems as workers, we can take advantage of the affordable, highly-scalable nature of ASR in settings where it works, while using human workers to ensure that DHH users always have access to accurate captions. ASR can eventually use Scribe as an in situ training tool, resulting in systems that are able to provide reliable captions right out of the box using human intelligence, and scale to fully automated solutions quicker than would otherwise be possible.

More generally, Scribe is an example of an interactive system that deeply integrates human and machine intelligence in order to provide a service that is still beyond what computers can do alone. We believe it may serve as a model for interactive systems that solve other problems of this type.

Acknowledgments

This work was supported by the National Science Foundation under awards #IIS-1149709 and #IIS-1218209, the University of Michigan, Google, an Alfred P. Sloan Foundation Fellowship, and a Microsoft Research Ph.D. Fellowship.

References

1. Bernstein, M.S., Brandt, J.R., Miller, R.C., Karger, D.R. Crowds in two seconds: Enabling realtime crowd-powered interfaces. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, page to appear (New York, NY, USA, 2011). ACM, 33–42.
2. Bigham, J.P., Jayant, C., Ji, H., Little, G., Miller, A., Miller, R.C., Miller, R., Tatarowicz, A., White, B., White, S., Yeh, T. Vizwiz: Nearly real-time answers to visual questions. In *Proceedings of the 23nd Annual ACM Symposium on User Interface Software and Technology*, UIST '10, (New York, NY, USA, 2010). ACM, 333–342.
3. Cooke, M., Green, P., Josifovski, L., Vizinho, A. Robust automatic speech recognition with missing and unreliable acoustic data. *Speech commun. 34*, 3 (2001), 267–285.
4. Driedger, J. Time-scale modification algorithms for music audio signals. Master's thesis, Saarland University, 2011.
5. Edgar, R. Muscle: multiple sequence alignment with high accuracy and high throughput. *Nucleic acids research 32*, 5 (2004), 1792–1797.
6. Elliot, L.B., Stinson, M.S., Easton, D., Bourgeois, J. College students learning with C-print's education software and automatic speech recognition. In *American Educational Research Association Annual Meeting* (New York, NY, 2008), AERA.
7. Flowerdew, J.L. Salience in the performance of one speech act:the case of definitions. *Discource Processes 15*, 2 (Apr–June 1992), 165–181.
8. Metze, F., Gaur, Y., Bigham, J. P. Manipulating word lattices to incorporate human corrections. In *Proceedings of INTERSPEECH*, (2016).
9. Gaur, Y., Lasecki, W.S., Metze, F., Bigham, J.P. The effects of automatic speech recognition quality on human transcription latency. In *Proceedings of the 13th Web for All Conference* (2016) ACM.
10. Glass, J.R., Hazen, T.J., Cyphers, D.S., Malioutov, I., Huynh, D., Barzilay, R. Recent progress in the MIT spoken lecture processing project. In *Interspeech* (2007), 2553–2556.
11. Gordon, M., Bigham, J.P., Lasecki, W.S. Legiontools: A toolkit+ UI for recruiting and routing crowds to synchronous real-time tasks. In *Adjunct Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (2015) ACM, 81–82.
12. Gordon-Salant, S. Aging, hearing loss, and speech recognition: stop shouting, i can't understand you. In *Perspectives on Auditory Research*, volume 50 of *Springer Handbook of Auditory Research*. A.N. Popper and R.R. Fay, eds. Springer New York, 2014, 211–228.

13. Jensema, C., McCann, R., Ramsey, S. Closed-captioned television presentation speed and vocabulary. In *Am Ann Deaf 140*, 4 (October 1996), 284–292.
14. John, B.E. Newell, A. Cumulating the science of HCI: from s-R compatibility to transcription typing. *ACM SIGCHI Bulletin 20*, SI (Mar. 1989), 109–114.
15. Kadri, H., Davy, M., Rabaoui, A., Lachiri, Z., Ellouze, N., et al. Robust audio speaker segmentation using one class SVMs. In *IEEE European Signal Processing Conference* (Lausanne, Switzerland, 2008) ISSN: 2219-5491.
16. Kushalnagar, R.S., Lasecki, W.S., Bigham, J.P. Captions versus transcripts for online video content. In *Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility*, W4A '13, (New York, NY, USA, 2013), ACM, 32:1–32:4.
17. Kushalnagar, R.S., Lasecki, W.S., Bigham, J.P. Accessibility evaluation of classroom captions. *ACM Trans Access Comput. 5*, 3 (Jan. 2014), 1–24.
18. Lasecki, W. Bigham, J. Online quality control for real-time crowd captioning. In *International ACM SIGACCESS Conference on Computers & Accessibility*, ASSETS 2012, 2012.
19. Lasecki, W., Miller, C., Sadilek, A., Abumoussa, A., Borrello, D., Kushalnagar, R., Bigham, J. Real-time captioning by groups of non-experts. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, UIST '12, (2012), 23–34.
20. Lasecki, W.S., Gordon, M., Koutra, D., Jung, M.F., Dow, S.P., Bigham, J.P. Glance: rapidly coding behavioral video with the crowd. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, (New York, NY, USA, 2014). ACM, 1.
21. Lasecki, W.S., Homan, C., Bigham, J.P. Architecting real-time crowd-powered systems. *Human Computation 1*, 1 (2014).
22. Lasecki, W.S., Kushalnagar, R., Bigham, J.P. Helping students keep up with real-time captions by pausing and highlighting. In *Proceedings of the 11th Web for All Conference*, W4A '14 (New York, NY, USA, 2014). ACM, 39:1–39:8.
23. Lasecki, W.S., Miller, C.D., Bigham, J.P. Warping time for more effective real-time crowdsourcing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13 (New York, NY, USA, 2013). ACM, 2033–2036.
24. Lasecki, W.S., Murray, K., White, S., Miller, R.C., Bigham, J.P. Real-time crowd control of existing interfaces. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, page To Appear (New York, NY, USA, 2011). ACM, 23–32.
25. Lasecki, W.S., Song, Y.C., Kautz, H., Bigham, J.P. Real-time crowd labeling for deployable activity recognition. In *Proceedings of the 2013 conference on Computer supported cooperative work* (2013) ACM, 1203–1212.
26. Marschark, M., Sapere, P., Convertino, C., Seewagen, R. Access to postsecondary education through sign language interpreting. *J Deaf Stud Deaf Educ. 10*, 1 (Jan. 2005), 38–50.
27. Naim, I., Gildea, D., Lasecki, W.S., Bigham, J.P. Text alignment for real-time crowd captioning. In *Proceedings North American Chapter of the Association for Computational Linguistics (NAACL)* (2013), 201–210.
28. Salisbury, E., Stein, S., Ramchurn, S. Real-time opinion aggregation methods for crowd robotics. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems (2015), 841–849.
29. Turner, O.G. The comparative legibility and speed of manuscript and cursive handwriting. *The Elementary School Journal* (1930), 780–786.
30. Wald, M. Creating accessible educational multimedia through editing automatic speech recognition captioning in real time. *Interactive Technology and Smart Education 3*, 2 (2006), 131–141.
31. Wang, L., Jiang, T. On the complexity of multiple sequence alignment. *J Comput Biol. 1*, 4 (1994), 337–348.
32. World Health Organization. Deafness and hearing loss, fact sheet N300. http://www.who.int/mediacentre/factsheets/fs300/en/, February 2014.

Walter S. Lasecki (wlasecki@umich.edu), Computer Science & Engineering, University of Michigan.

Christopher D. Miller, Iftekhar Naim, Adam Sadilek, and Daniel Gildea (c.miller@rochester.edu) ({inaim,sadilek,gildea}@cs.rochester.edu), Computer Science Department, University of Rochester.

Raja Kushalnagar (raja.kushalnagar@gallaudet.edu), Information Technology Program, Gallaudet University.

Jeffrey P. Bigham (jbigham@cmu.edu), HCI and LT Institutes, Carnegie Mellon University.