Warping Time for More Effective Real-Time Crowdsourcing

Walter S. Lasecki, Christopher D. Miller, and Jeffrey P. Bigham
University of Rochester Computer Science
{wlasecki, jbigham}@cs.rochester.edu, c.miller@rochester.edu

ABSTRACT

In this paper, we introduce the idea of "warping time" to improve crowd performance on the difficult task of captioning speech in real-time. Prior work has shown that the crowd can collectively caption speech in real-time by merging the partial results of multiple workers. Because non-expert workers cannot keep up with natural speaking rates, the task is frustrating and prone to errors as workers buffer what they hear to type later. The *TimeWarp* approach automatically increases and decreases the speed of speech playback systematically across individual workers who caption only the periods played at reduced speed. Studies with 139 remote crowd workers and 24 local participants show that this approach improves median coverage (14.8%), precision (11.2%), and per-word latency (19.1%). Warping time may also help crowds outperform individuals on other difficult real-time performance tasks.

Author Keywords

Real-Time Crowdsourcing; Human Computation; Captioning

ACM Classification Keywords

H.5.m. Information Interfaces and Presentation: Misc.

General Terms

Human Factors; Design; Measurement

INTRODUCTION

Real-time captioning provides deaf and hard of hearing people access to the aural speech around them. Past approaches to real-time captions used either *(i)* costly professional captionists (stenographers) who are not available on demand, or *(ii)* automatic speech recognition that often produces unusable results in real-world settings. Legion:Scribe allows the *crowd* to caption speech in real-time by having workers type part of the speech they hear, then automatically merging the pieces together [7]. The captions produced are better than ASR and approach the quality of those by stenographers [5].

Captioning speech in real-time is difficult for crowd workers because they cannot keep up with natural speaking rates, which routinely reach 150 to 225 words per minute (WPM) [4]. In this paper, we introduce the *TimeWarp* approach for better real-time crowdsourcing and apply it to the problem of real-time captioning. The idea is to *warp* (slow down) time relative to individual workers in order to make each worker's task easier to complete online. Workers are asked to caption only specific portions of the speech, which are coordinated between workers (Figure 1). Thus, by slowing each worker's audio at different times in the stream and speeding up content workers are not responsible for, the crowd is able to collectively complete their task in real-time while keeping up with the speech during the faster periods.

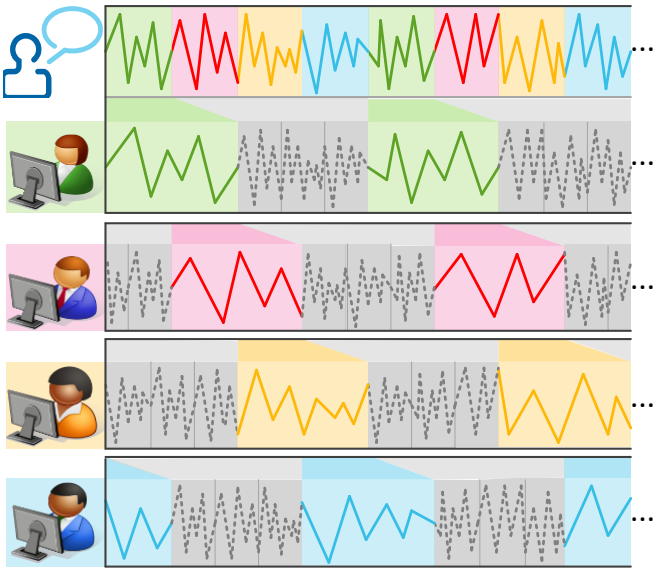

Figure 1. ***TimeWarp*** **forwards different "warped" versions of the audio to each worker. Workers hear a slowed down version of the content they are supposed to caption, while they hear the rest sped up. This increases the quality of workers' output by asking them to complete an easier task without losing context. The start of each slowed segment is aligned with the original audio, allowing the crowd to collectively caption in real time.**

We evaluated *TimeWarp* with 139 unique workers recruited from Amazon's Mechanical Turk over 257 trials, and show that with *TimeWarp* workers were able to caption an average of 11.4% more of the speech and do so 12.6% more accurately. Most interestingly, workers also reduced per-word latency by 16.8%, which suggests that the elimination of the cognitive buffering necessary when captioning at speeds higher than one's typing rate more than compensated for the introduced delay. Additional studies with 24 local participants showed similar latency improvement. Post-trial interviews indicated workers felt less pressured and altered their captioning style when the content played slower.

Real-time captioning is a task which requires a significant cognitive load and coordination on the part of the user to complete correctly. This makes it well-suited for *TimeWarp*. We conclude with a description of task properties that suggest

CHI'13, May 2, 2013, Paris, France.

TimeWarp would be exceptionally useful for workers completing that task, and discuss how the approach may generalize to other real-time crowdsourcing tasks.

In this paper we make the following contributions:

- We introduce the idea of warping time to make real-time crowdsourcing tasks easier to solve.
- We introduce the *TimeWarp* approach for allowing individual workers to systematically receive some content that is slowed down while the rest is sped up to compensate.
- We present studies showing that *TimeWarp* can help workers to improve their captioning performance in terms of coverage, precision, and latency.
- We discuss participant feedback on *TimeWarp* and explore how it can be applied to other problems.

RELATED WORK

Real-time captioning is supported by professional captionists, automatic speech recognition (ASR), or more recently, crowd captioning systems such as Legion: Scribe [7]. Each of these approaches has limitations. Professional captionists are the most reliable option, but are not available on demand, can cost over $150/hr, and can only be scheduled in blocks of an hour. ASR is relatively cheap and available on-demand, but often provides extremely low-quality results in realistic settings. Crowd captioning uses multiple non-expert human captionists each submitting partial input which is then automatically recombined to generate a caption in real-time. This approach is more robust in real-world situations, but requires that workers contribute useful partial captions. Using time warps, we are able to improve individual worker captioning performance by giving them a more manageable task, while still providing answers in real-time.

Crowd captioning is a type of real-time human computation. Real-time human computation has been explored in systems like VizWiz [2], which was one of the first applications to target nearly real-time responses from the crowd, and Adrenaline [1], which uses a retainer model to reduce response time to less than two seconds. Legion introduced the idea of engaging a synchronous crowd in a continuous real-time task, using the crowd to collectively control existing user interfaces as if they were a single individual [6]. Each workers submits input independently of other workers, then the system uses an *input mediator* to combine the input into a single control stream.

While prior work has investigated offline captioning using the crowd [8, 9], Legion:Scribe is the only system and worker interface for real-time online captioning [7]. Scribe extends the idea of using continuous stream of input from workers to real-time captioning, generating transcripts by combining multiple workers' partial captions into a single final caption stream. Legion:Scribe allows deaf users to stream content from the mobile devices to a server, which forwards it to multiple workers. Multiple partial captions from workers are sent to the server where they are merged and then forwarded back to the user. The *TimeWarp* approach represents a way to modify the underlying problem so that it is easier for workers to complete. One of the ways we measure worker's performance is using the *coverage* metric used in [7]. Coverage is similar to recall, but with the added constraint that the word be entered within a fixed time window (in our tests, 10 seconds).

CAPTIONING SYSTEM

The goal of *TimeWarp* is to allow each worker to type slowed clips played as close to real-time as possible while still maintaining the context acquired by hearing all of the audio. It does 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 did a preliminary study 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 8 seconds (our upper bound on acceptable latency). Since the mean speaking rate is around 150 WPM [4], workers will hear 8 words in roughly 3.2 seconds, with an entry time of roughly 8 seconds 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 $\frac{N-1}{N-r} = \frac{3}{2} = 1.5\times$.

System Architecture

Our system architecture is similar to Legion:Scribe [7]. Audio is forwarded from a laptop or mobile device to a server running Flash Media Server (FMS). Since FMS does not allow access to the underlying waveform for live streams, we connect to FMS using N instances of FFmpeg (ffmpeg.org) – one for each offset – then use FFmpeg to modify the stream to play it faster or slower. The N streams are then forwarded to worker pages that present workers recruited from either Mechanical Turk or volunteers with the appropriate version of the audio. Worker input is then forwarded back to the server where it is recorded and scored for accuracy.

In order 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 [3]. 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.

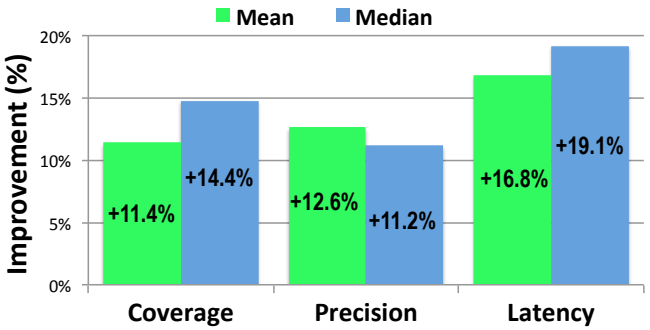

Figure 2. Relative improvement from no warp to warp conditions in terms of mean and median values of coverage, precision, and latency.

Worker Interface

Once the audio is streaming, workers are shown a captioning interface consisting of a text box to enter captions in, a score box which tracks the points workers have earned, and visual and audio alerts telling them when they should or should not be captioning. Visual alerts include a status message that changes between a green "type what you hear now" alert and a red "do not type" alert. Workers are able to see an animation of the points they earn flying from the word they input to the score box and being added to their total. Audio cues consist of tones played when we want users to start and stop captioning, and volume adjustments that reduce the volume of content we do not want workers to caption. We lower the volume instead of mute it in order to help workers maintain context even when they are not actively captioning.

EVALUATION

To evaluate *TimeWarp*, we ran two studies that asked participants to caption a 2.5 minute (12 captioning cycles) lecture clip from MIT's Open CourseWare project (www.ocw.mit.edu). In the first, we recruited workers from Amazon's Mechanical Turk and, in the second, we recruited 24 local participants. In the Mechanical Turk study, we ran 257 trials with 139 unique workers using a between subjects design. 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. Workers were allowed to complete at most two tasks and were randomly routed to each condition. Since Mechanical Turk often contains low quality (or even malicious workers), we first removed input which got less than 10% coverage or precision. A total of 206 tasks were approved by this quick check. Workers were paid a base rate of 5 cents if their input was accepted by the automated check, and were paid a bonus of roughly 1 cent for every 5 words they got correct. After the automated check, we calculated the F_1 score, the harmonic mean (a common metric from information retrieval) of the coverage and precision, to get a single representative score for each pair of values. We then calculated the mean and standard deviation (σ) of these scores, and removed any inputs with a score more than 2σ from the mean as outliers.

Figure 2 and Table 1 show the results from the remaining 196 trials (98 from the warp and no-warp conditions). Worker's mean coverage 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% ($t(df) = 5.41$, $p < .001$).

Local Workers

We evaluated *TimeWarp* with local participants who were generally more skilled 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. The ordering of the trial conditions was randomized, and the segment was picked randomly.

Unlike the Mechanical Turk workers, our students worker all rated themselves as proficient typists and 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. Thus, these participants did not seem as overwhelmed by the original task, and seemed to benefit less from *TimeWarp* helping them to keep up. For these workers, 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 ($t(df) = 2.78$, $p < .01$).

DISCUSSION

Our results showed that *TimeWarp* improved worker's coverage, precision, and latency on the real-time captioning task. While workers are still short of being able to reliably caption the requested content entirely on their own, Scribe is designed to leverage multiple workers in order to reach coverage rates exceeding 90%. This means that the collective is still capable of rates competitive with professional captionists.

The effect of TimeWarp was particularly positive for workers on Mechanical Turk, who struggled most with the original task. Observations, surveys, and follow-up interviews conducted with the local workers indicated that worker skill level was the most indicative of their preference of warping the signal over regular playback, with less skilled workers rating the time warps higher while more skilled workers found it unnecessary. The most significant complaint about the system, regardless of skill level, was the quality of the warped audio since our warping approach reduced the quality of the audio and added a slight echo. We discuss how this may be improved in future versions of the system in the next section.

The observed reduction in latency seems counterintuitive because slowing playback in *TimeWarp* reduces best-case latency. The likely cause of this improvement is that the slower playback of speech allowed workers to alter their approach to captioning. At regular playback speeds, workers cannot typically match the speed of the speaker, leading them to a behavior in which they first listen to the clip and memorize the content, then type what they heard. This adds a delay that can begin at over 3.25 seconds and grows as the worker types (this effect was also observed in [5]).

When the audio playback was slowed, workers begin typing words as soon as they were spoken because they generally

Median	Baseline	TimeWarp	Improvement
Coverage	48.18%	55.30%	14.78%
Precision	59.98%	66.67%	11.15%
Latency	4.80s	3.99s	19.11%

Mean	Baseline	TimeWarp	Improvement
Coverage	48.09%	53.56%	11.39%
Precision	58.53%	65.92%	12.61%
Latency	4.80s	3.99s	16.77%

Table 1. Results from experiments on Mechnical Turk showed significant improvement in coverage, precision, and latency when using *TimeWarp*.

had enough time to type a word before they heard the next one. The forced delay of the slower playback, which hits a maximum of 3.25 seconds at the end of a warped clip, was still less than the delay incurred by the practice of storing and then recalling and typing an entire sequence from working memory. Interviews with local workers confirmed the existence of this observed behavior: workers who were interviewed indicated they followed these two different patterns when presented with normal and slowed segments of audio.

Interviews also showed that workers (especially less experienced typists) felt as if they were under less pressure when the audio was played slowly when they were expected to be captioning. We expect that this mirrors the sentiments of web workers, where stress is likely to play a key role in how likely a worker is to return to a task in the future, effecting the size and cost of the pool of workers available on-demand.

FUTURE WORK

TimeWarp illustrates one of the most interesting qualities of real-time crowdsourcing - by cleverly partitioning work to different workers, performance demands can be reduced per worker, even while collective performance increases. Because of the importance of time pressure on performance tasks, the *TimeWarp* approach may be useful in helping the collective perform better than constituent workers on demanding tasks. Our findings also suggest improvements that can be made to the *TimeWarp* system for real-time captioning.

Improved Tempo-Shift Algorithms

One complaint that was common among participants was that "echo" present in the slowed down audio. This echo is a result of WSOLA not being the optimal algorithm for transforming speech [11]. In the future, we will implement a version of this system that uses the Phase Vocoder algorithm (which works in the frequency domain), or a specialized version of WSOLA [11] or Time Domain Pitch Synchronous Overlap and Add (TD-PSOLA) which are designed for pitch consistency and are more well suited to transforming speech with fewer artifacts. Libraries such as libsonic (dev.vinux-project.org/sonic) include implementations of these algorithms, but they must be adapted to work on streaming content.

Other Applications and Generalization

Our studies show that *TimeWarp* makes real-time tasks easier by mitigating the effects of high cognitive load and human motor limitations. Existing real-time crowdsourcing systems that enable crowd control of interfaces [6] and real-time segmentation and labeling of video [10] may benefit immediately. The general approach of making tasks easier for individual workers in order to improve collective performance likely applies across many applications.

CONCLUSIONS

In this paper we have presented the idea of systematically "warping time" to increase performance on continuous real-time crowdsourcing tasks by slowing the relative speed of the task for each individual worker. Our experiments with Mechanical Turk workers performing a real-time captioning task demonstrated that coverage and accuracy can be significantly improved using this system. Interestingly, mean latency also improves despite the reduced playback speed. A second study with local users showed similar results in terms of latency, and revealed that the work flow used by workers was altered by the lower play speed. This change resulted in a less stressful task that workers could truly complete in real-time instead of buffering what they hear. Our results demonstrate the promise that the time warp model holds as a means of supporting complex continuous real-time tasks with the crowd without requiring highly skilled workers, which may reduce cost and increase the availability of the service.

ACKNOWLEDGMENTS

We would like to thank Craig Harman for his contributions. This work has been supported by Google and National Science Foundation Awards #IIS-1149709 and #IIS-1016486.

REFERENCES

1. Bernstein, M. S., Brandt, J. R., Miller, R. C., and Karger, D. R. Crowds in two seconds: Enabling realtime crowd-powered interfaces. In *UIST 2011*.
2. Bigham, J. P., Jayant, C., Ji, H., Little, G., Miller, A., Miller, R. C., Miller, R., Tatarowicz, A., White, B., White, S., and Yeh, T. Vizwiz: nearly real-time answers to visual questions. In *UIST 2010*.
3. Driedger, J. Time-scale modication algorithms for music audio signals. Master's thesis, Saarland University, 2011.
4. C. Jensema, R. McCann, S. Ramsey. Closed-captioned TV presentation speed and vocabulary. In *Am Ann Deaf*. 141(4):284–92. 1996.
5. Lasecki, W., and Bigham, J. Online quality control for real-time crowd captioning. In *ASSETS 2012*.
6. Lasecki, W., Murray, K., White, S., Miller, R. C., and Bigham, J. P. Real-time crowd control of existing interfaces. In *UIST 2011*.
7. Lasecki, W. S., Miller, C. D., Sadilek, A., Abumoussa, A., Borrello, D., Kushalnagar, R., and Bigham, J. P. Real-time captioning by groups of non-experts. In *UIST 2012*.
8. Y. C. Beatrice Liem, H. Zhang. An iterative dual pathway structure for speech-to-text transcription. In *HCOMP 2011*.
9. Luz, S. and Masoodian, M. and Rogers, B. Supporting collaborative transcription of recorded speech with a 3D game interface. In *KES 2010.*.
10. Lasecki, W. S., Song, Y. C., Kautz, H., and Bigham, J. P. Real-time crowd labeling for deployable activity recognition. In *CSCW 2013*.
11. Verhelst, W., and Roelands, M. An overlap-add technique based on waveform similarity (wsola) for high quality time-scale modification of speech. In *ICASSP 1993*.