Using Keyword Spotting to Help Humans Correct Captioning Faster

Yashesh Gaur[1], *Florian Metze* [1], *Yajie Miao* [1], *and Jeffrey P. Bigham* [1,2]

[1] Language Technologies Institute, Carnegie Mellon University
[2]Human Computer Interaction Institute, Carnegie Mellon University

yashesh,fmetze,ymiao,jbigham@cs.cmu.edu

Abstract

Automatic real-time captioning provides immediate and on demand access to spoken content in lectures or talks, and is a crucial accommodation for deaf and hard of hearing (DHH) people. However, in the presence of specialized content, like in technical talks, automatic speech recognition (ASR) still makes mistakes which may render the output incomprehensible. In this paper, we introduce a new approach, which allows audience or crowd workers, to quickly correct errors that they spot in ASR output. Prior approaches required the crowd worker to manually "edit" the ASR hypothesis by selecting and replacing the text, which is not suitable for real-time scenarios. Our approach is faster and allows the worker to simply type corrections for misrecognized words as soon as he or she spots them. The system then finds the most likely position for the correction in the ASR output using keyword search (KWS) and stitches the word into the ASR output. Our work demonstrates the potential of computation to incorporate human input quickly enough to be usable in real-time scenarios, and may be a better method for providing this vital accommodation to DHH people.

Index Terms: speech recognition, human-computer interaction, spoken term detection, real-time crowd sourcing.

1. Introduction

Automatic speech recognition (ASR) technology has made great advances and it has evolved from desktop recognition in ideal recording conditions to mobile based recognition in real world settings. Most recently, the use of deep learning has led to impressive improvements [1]. However, even today's state-of-the-art ASR can easily produce errors that alter the meaning or obfuscate the contents of speech. In this paper, we consider a particular use case, which is still challenging today: real-time captioning of (technical) talks.

Real time captioning aims to generate visual text from aural speech with very low latency. It is very useful for the DHH population, as it provides access to "real-time subtitles", when taking a class at school or university, when attending conferences, or in any educational or social function. Current options for real time captioning are extremely limited and can be classified into 3 main varieties: (1) Communications Access Real-Time Translation (CART), (2) non-verbatim systems and (3) automatic speech recognition.

CART is the most reliable real-time captioning service. Extensively trained stenographers type on specialized keyboards that map key presses to phonemes, that are then expanded to verbatim text with accuracy generally over 95%. Stenography requires 2-3 years of training and enables the stenographer to type at natural speaking rates that average 141 words per minute (WPM) and can reach up to 231 WPM [2]. However, stenographers are expensive and require advanced booking, often in blocks of at-least an hour. Consequently, they cannot be used to caption any lecture or event at the last minute, or provide access to unpredictable learning opportunities such as discussions after the class. Moreover, stenographers do not have expertise based on the technical area covered in the lecture. This can result in distortions being introduced in the transcripts. Non-Verbatim Systems include computer-based macro expansion services like C-Print. C-Print captionists need less training and generally charge much less than stenographers. However, they cannot type fast enough to produce verbatim transcripts. Difficulty in paraphrasing technical talks and advanced booking makes non-verbatim systems a non-ideal choice for real-time captioning.

Automatic speech captioning has attracted significant attention from the research community [3, 4]. While services like CART and C-Print are expensive and required prior booking, ASR is relatively inexpensive and available on demand. However, although ASR works well in ideal conditions with suitably adapted models, its performance degrades when exposed to real-world settings. Reasons for this degradation are manifold: lectures and talks are usually held in rooms/ halls with echo, noise from the audience, music, and reverberation. Often the recording microphone is not optimal quality and is kept far away from the mouth. Use of a generic language model for a talk in a specialized domain further deteriorates the condition as it is difficult to create adapted language models for every class/ discipline [5, 6]. Technical jargon in the lectures is often missing from the ASR vocabulary and it is therefore mis-recognized every time it appears. Not only do these words contribute to recognition errors, they are words that carry important meaning and occur frequently throughout the lecture.

This suggests that real-time captioning solutions are either expensive, not available on demand, or produce unacceptable errors. People, unlike machines, are very good at speech recognition and taking help from them to aid the ASR looks promising. Crowd-sourcing was found to be a cost-effective means of transcribing pod-casts [7], data from speech dialog systems [8], or conversational speech [9, 10]. It is also possible to identify likely error regions in automatic transcripts, and have the crowd review these [11]. The resulting transcripts enhance the usefulness and accessibility of otherwise automatically transcribed material [12]. Kolkhorst *et al.* have also specifically targeted editing transcriptions of classroom lectures [13].

However, all of these approaches take help from people in an offline setup, and therefore, they cannot be used in an application like real-time captioning. People who can hear can perform speech transcription, but most cannot type fast enough to keep up with natural speaking rates. Lasecki *et al.* [14] tried to circumvent this bottleneck by using input from a group of non-experts, like students in a classroom, to collectively caption speech with delays of less than 5 seconds. Their systems encourages each person to caption a certain portion of the au-

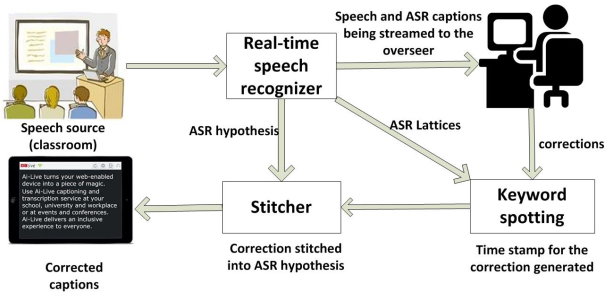

Speech and ASR captions being streamed to the overseer
Real-time speech recognizer
Speech source (classroom)
ASR hypothesis
ASR Lattices
corrections
Ai-Live turns your web-enabled device into a piece of magic. Use Ai-Live captioning and transcription service at your school, university and workplace or at events and conferences. Ai-Live delivers an inclusive experience to everyone.
Stitcher
Keyword spotting
Correction stitched into ASR hypothesis
Time stamp for the correction generated
Corrected captions

Figure 1: *Speech and ASR captions are streamed to the overseer in real-time. Corrections are entered by the overseer if mistakes are observed. KWS quickly finds the time at which correction word must have been spoken and then passes it to the stitcher, which stitches the correction into the ASR captions.*

dio and these segmented captions are stitched together using multiple sequence alignment to give continuous transcripts in real-time. However, this approach requires the combined effort of several people to work.

In this paper, we present a new approach for combining human intelligence with machine intelligence for the purpose of providing real-time captioning. Our approach can work with a single human overseer, who does not directly produce work, but rather oversees real-time ASR to quickly identify and correct its mistakes. The novelty of our approach is in the way in which the overseer interacts with the ASR: previous work has had humans "edit" the hypothesis of the ASR to generate correct output [15, 13]. These approaches are less suitable for a real-time ASR as editing hypothesis may take long and variable amounts of time. Our approach only requires the overseer to observe the ASR output for any mis-recognitions and enter the corresponding correct word[1] as soon the mistake is observed *i.e.* he or she simply types in corrections as the audio and the corresponding captions play along, without "placing" or "anchoring" them. The system, using keyword search (KWS) techniques [16], calculates where the entered word or phrase must have been spoken in the recent audio, and then stitches the correct word into the ASR hypothesis, yielding the corrected captions.

2. Combining ASR and KWS

Figure 1 shows the architecture of our approach. Audio from a speech source, say a classroom lecture, is fed to a real-time ASR. The ASR captions and the audio are streamed to the overseer. The overseer observes the captions for any mistakes in real-time and enters the corresponding correction if one is required. The correction is fed to the keyword search module, which locates the word in the ASR lattices corresponding to recently decoded speech. After the word is detected, the word along with its time-stamp is sent to the stitcher module, which intelligently stitches the correction into the ASR captions, to yield the corrected caption. This caption can then be streamed to the general audience. This section covers the 2 modules of our system, i.e, Keyword Spotting and Stitcher in detail.

2.1. Keyword Search

Different from ASR's speech to text approach, keyword search (KWS), also known as keyword spotting or spoken term detection (STD, [17]), determines *when* a given word (or search term) was spoken, rather than providing a dense transcription. Popular approaches implement a search through word, syllable, or phone lattice produced by an ASR system [18]. Searching through lattices allows us to go beyond the 1st-best hypothesis, and consider during KWS possible alternatives that were also being considered by the ASR, but ultimately discarded. Importantly, KWS is not constrained by the ASR's vocabulary, and can locate arbitrary words [19, 20].

Keyword search can be used for searching spoken terms over huge collection of audio data [21]. In our approach, we take help of this technique to know where the correction word (or phrase) provided by the overseer has been spoken in the recent audio. The hope is that even though the correction was not present in the 1-best hypothesis, the information from the lattice will allow us to determine which part of the transcription should be modified to accomodate the correction, and how.

Performing keyword search for the correction word in the lattices corresponding to recent audio will result in a list of timestamps where the word could have been spoken. Each timestamp is also associated with a confidence score. This list is passed to the stitcher module, which decides how to stitch the correction word into the hypothesis to correct the caption.

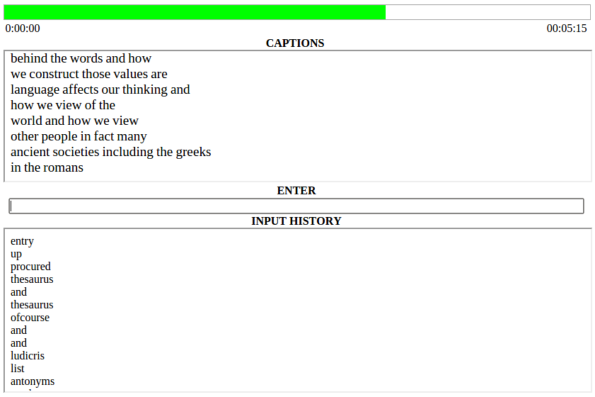
2.2. Stitcher

Once the KWS module gives a list of possible detections for the correction word, the stitcher module selects one detection out of them and replaces one or more words in the ASR hypothesis with the correction word. The stitcher module looks for a detection within a time window stretching back into history from when the correction was fired. The width of this search window is crucial and should be modelled based on how long it takes the overseer to enter the correction once the mis-recognized word was displayed. If this window is too small, we risk not detecting the correction word. If the window is too large, the stitcher is presented with many detections (both false alarms and genuine detections from the correction word being recognized correctly in another part of the sentence) and might choose the wrong one. From the search window, the stitcher picks the detection with the highest confidence score, provided that the correction word is not already present in the hypothesis at detection's timestamp. This can happen often because, the correction word can be present more than once in the search space but it is incorrectly recognized only once. To stitch the word into the ASR hypothesis at the determined timestamp, the stitcher considers factors like the duration of the detection and phonetic similarity between the correction word and the word(s) present in the hypothesis, at the timestamp of the detection.

3. Experimental setup

To see if our approach could result in better captions, we prepared an interface that would allow the overseer to monitor the captions and enter the corrections (Figure 2). The experiment was conducted with workers on Amazon's Mechanical Turk. We used the TEDLIUM corpus [22], and trained an ASR system with Kaldi's [23] TEDLIUM recipe, using PDNN [24]. Kaldi keyword search system [16] was used for keyword spotting. TEDLIUM is a collection of TED talks and resembles classroom lectures to some degree. Each talk in the test set is about

[1] For clarity of presentation, we will simply use the term "word" in this work, however a correction can be, and often will be a "phrase", *i.e.* a sequence of words.



0:00:00
00:05:15

CAPTIONS

behind the words and how
we construct those values are
language affects our thinking and
how we view of the
world and how we view
other people in fact many
ancient societies including the greeks
in the romans

ENTER

INPUT HISTORY

entry
up
procured
thesaurus
and
thesaurus
ofcourse
and
and
ludicris
list
antonyms

Figure 2: *Interface enabling the overseer to enter corrections as the audio and the ASR captions play. The transcription appears in the top part of the window as the audio plays along. The overseer enters corrections in the central "ENTER" field by entering a word (or phrase) and then pressing "return". A history of entered corrections is shown on bottom.*

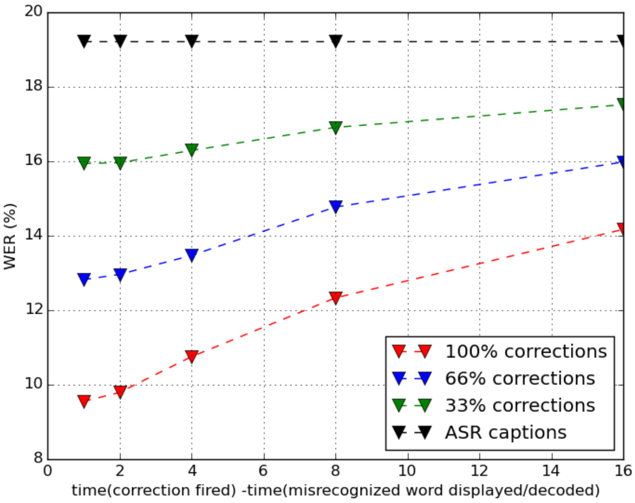


Figure 3: *System performance in terms of WER w.r.t. percentage of corrections entered and search window size.*

15 minutes long on average. There are 11 talks and we assigned one talk per Mechanical Turk worker. The resulting captions had a word error rate of 19.2%. The experiment involved the overseer to supervise the streaming captions while listening to the audio, and enter corrections if mistakes were spotted. The participants were asked to complete the supervision of a talk in one sitting, without any breaks. They were paid according to a standard hourly wage of $6/hour.

4. Results and Discussion

To see the efficacy of our approach, we started with a simulation experiment. Figure 3 depicts the system's performance by simulating corrections instead of getting them from the crowd. We simulate "correction" words whenever the original ASR output contains a substitution or deletion error. Insertions are thus ignored. Two system parameters were varied: the percentage of total corrections entered, and the time difference (lag) between when the misrecognized word appeared in the transcript, and when the correction was "entered", so that the KWS system could process it. A search window a little greater than this time difference was then given to the stitcher, in order to be able to process this correction.

It can be observed that WER does not go to zero, even if 100% of the corrections are provided within just 1 second of when the mis-recognized word was made available ("lower bound"). This is because *(i)* insertion errors cannot be corrected, *(ii)* time alignments for words (in particular, short words) are often imprecise within a few 100s of milli-seconds, and *(iii)* because KWS sometimes does not return detections for the globally optimized parameters which we used in this work. Longer window sizes obviously result in worse output, although window sizes of up to two seconds seem to give good results.

Our current stitching scheme works by replacing entire words in the ASR hypothesis, and is inherently a substitution based solution. As a result, although our approach is very good at reducing substitutions, it is not as effective with insertion and deletion errors. However, insertions are not a significant source of errors here, and are often less harmful than deletions or substitutions. Figure 3 also shows the dependence of WER on search window size: larger search window sizes invite more detections, which consequently increase the chance of an incorrect stitching. Window sizes up to two seconds however do not result in significant degradation, while the amount of corrections entered has a linear influence on the final word error rate; there is no saturation effect or so when many corrections are being provided in rapid succession.

Table 1 shows the WER results for the ASR captions, the best possible performance (*i.e.*, assuming the user was able to enter all the corrections) and a single human overseer from Mechanical Turk supervising the ASR captions.

Table 1: *WER results for ASR captions, simulated lower bound performance, and a single MTurk observer correcting the ASR captions.*

Error	ASR Captions	Lower bound	MTurk observer
WER	19.2 %	9.6 %	16.6 %
Substitution	12.8 %	3.1 %	10.3 %
Insertion	2.3 %	1.7 %	2.1 %
Deletion	4.1 %	4.8 %	4.2 %

The lower bound performance shows that, even when using our current primitive stitching algorithm, human correction has the potential to cut the WER of the ASR captions to half. Our trials with workers on Amazon's Mechanical Turk however showed that it is difficult for a single worker to get close to this lower bound (see Table 1). In a typical Mechanical Turk worker input, 3.9 % of the entered words were misspelled and 21.4% words were not misrecognized words but words that had been recognized correctly. The remaining 74.8% crowd input (valid corrections) constituted about 29.5% of the overall "possible" corrections.

We solicitated feedback from all workers, and most of them mentioned that still they found it difficult to read, listen and type simultaneously. On average, a crowd worker was able to enter one word (possible correction) every 6.7 seconds. Assuming an average speaking rate of 141 WPM, and ASR captions with a

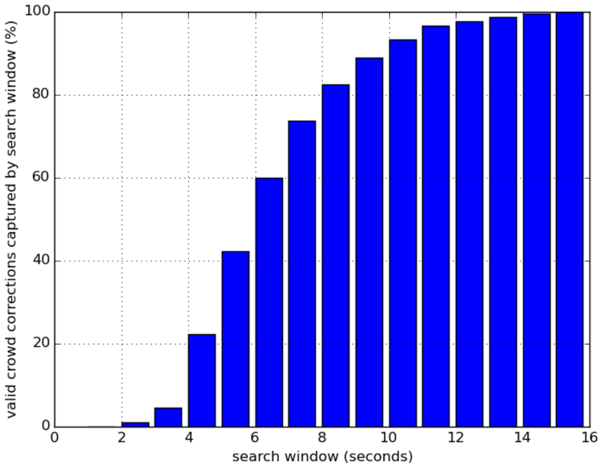


Figure 4: *Cumulative percentage of captured crowd corrections as a function of the search window based on a average Mechanical Turk worker response*

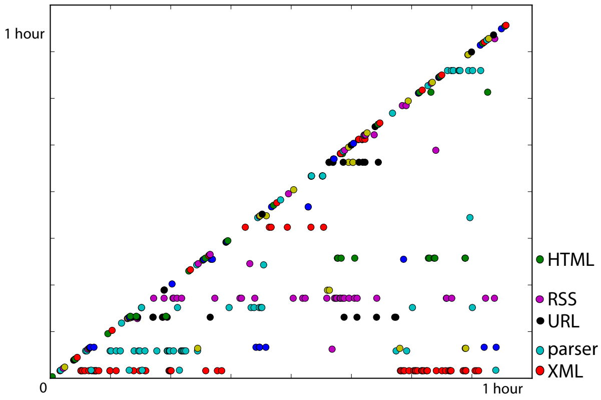


Figure 5: *Pattern of out-of-vocabulary (OOV) word occurrences and repetitions in a typical lecture from [26]: about 60% of OOV tokens repeat, showing up as horizontal lines in this plot. The 5 most repeated OOV words are labeled, and correspond to the primary content of the lecture*

WER of 19.2%, to cover all the errors, a worker would need to enter one correction every 2.2 seconds. The workers also mentioned difficulty in catching up with the audio and captions once they finished entering the correction. We are currently using this feedback to improve the interface and the task.

Figure 4 shows a cumulative histogram for number of valid corrections captured as a function of search window. We see that nearly all the corrections (93.0%) can be captured by a search window of 10 seconds. Our experiments have shown that 10 seconds is the optimal search window for actual crowd input. A window shorter than 10 seconds results in suboptimal performance due to loss of correction words, while with a window greater than 10 seconds, the disadvantage of inviting more detections by using a larger window overpowers the benefit we get from capturing the few correction words that were entered with a latency greater than 10 seconds.

The Kaldi TEDLIUM setup does not contain Out-Of-Vocabulary (OOV) words, while real-world classroom lectures almost certainly would. Still, getting technical jargon correct is often quite important to understanding a lecture properly. KWS techniques have been successfully used for spotting OOV words [19, 20]. To look for an OOV word, one can either do a phonetic (syllabic) search, *i.e.*, look for the phonetic sequence of the OOV word in the lattice or, search the lattice for words that are phonetically similar to the OOV word, also called proxy words. These techniques have proven to be pretty effective[25] and our approach can therefore leverage keyword spotting to handle OOVs as well.

To establish the importance of OOV words in technical talks, we analyzed a set of 23 lectures from a public corpus of Stanford computer science courses [26]. Across these lectures, 43% of the 10 words with the highest tf-idf score (computed with lectures as documents) were OOV with respect to a standard switchboard lexicon (which led to a reasonable overall OOV rate of 3.4%), yet they were typically important content words, e.g., *semaphore, RSS*, etc. Importantly, 60% of these words repeat, and many repeat often (Figure 5). This makes intuitive sense because a lecture will often introduce a concept with a new technical jargon and then repeat that multiple times over the course of the lecture. Our approach would be particularly fruitful in this scenario because once the overseer has entered an OOV word as a required correction, we can ask the system to keep searching for the OOV word and automatically correct it if it is being spotted again, potentially even before the overseer sees it. More importantly, this can be done transparently, without re-estimating the language model and regenerating a decoder's static search graph, which would take too long to be useful over a single lecture. Over time this would lead to less dependence on the overseer.

Another benefit of the proposed technique is that it can transparently handle text normalization conventions in cases such as "F. ZERO" (a typical recognizer output), which in a speech lecture note should probably be written as "F0".

5. Conclusion and Future Work

This paper presents a novel approach to captioning real-world lectures, that combines speech-to-text and keyword search techniques. We show how real-time corrections from a single human overseer can be used to reduce the word error rate of a transcript, without the overseer having to deal with a complex interface, or having to "edit" a transcript.

In future work, we will explore lattice-based "stitching" algorithms, in order to further optimize the replacement process, and better handle cases in which the crowd only provides part of the correction. Parsing techniques from natural language processing could also be integrated, in order to ensure corrections get stitched in in the best possible way. We will check the efficacy of our approach for detecting technical jargon/ OOV words on a dataset which models classroom technical lectures more closely. We will also work on incorporating input from multiple crowd workers into the ASR. Our initial experiments with multiple crowd workers' input have not shown significant benefits, and we believe that changes in the stitching algorithm and task design will have to be made to effectively combine input from multiple crowd workers. Will it for example be possible to encourage users to provide corrections for areas in which other users have not yet provide corrections, to ensure they will be complementary? In addition, it may be interesting to explore presentation slides or similar materials as sources of "candidate corrections", and compare our stitching approach to vocabulary expansion and langauge model adaptation.

6. Acknowledgements

This work was supported by National Science Foundation Award #IIS-1218209.

7. References

[1] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, Nov 2012.

[2] C. Jensema, R. McCann, and S. Ramsey, "Closed-captioned television presentation speed and vocabulary," *American Annals of the deaf*, vol. 141, no. 4, pp. 284–292, 1996.

[3] M. Cettolo, J. Niehues, S. Stüker, L. Bentivogli, and M. Federico, "Report on the 10th iwslt evaluation campaign," in *Proc. IWSLT*, Heidelberg; Germany, 2013, http://www.eu-bridge.eu/87_282.php.

[4] J. Glass, T. J. Hazen, S. Cyphers, I. Malioutov, D. Huynh, and R. Barzilay, "Recent Progress in the MIT Spoken Lecture Processing Project," in *Proc. Interspeech*, 2007. [Online]. Available: http://groups.csail.mit.edu/sls/publications/2007/Interspeech07-glass-lecture.pdf

[5] H. Yamazaki, K. Iwano, K. Shinoda, S. Furui, and H. Yokota, "Dynamic language model adaptation using presentation slides for lecture speech recognition," in *In Proc. INTERSPEECH*, 2007, pp. 2349–2352.

[6] C. Munteanu, G. Penn, and R. Baecker, "Web-based language modelling for automatic lecture transcription," in *Proc. INTERSPEECH*, 2007.

[7] J. Ogata, M. Goto, and K. Eto, "Automatic transcription for a web 2.0 service to search podcasts," in *INTERSPEECH 2007, 8th Annual Conference of the International Speech Communication Association, Antwerp, Belgium, August 27-31, 2007*, 2007, pp. 2617–2620.

[8] G. Parent and M. Eskenazi, "Toward better crowdsourced transcription: Transcription of a year of the let's go bus information system data," in *Spoken Language Technology Workshop (SLT), 2010 IEEE*. IEEE, 2010, pp. 312–317.

[9] M. Marge, S. Banerjee, and A. I. Rudnicky, "Using the amazon mechanical turk for transcription of spoken language," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*. IEEE, 2010, pp. 5270–5273.

[10] S. Novotney and C. Callison-Burch, "Cheap, fast and good enough: Automatic speech recognition with non-expert transcription," in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Los Angeles, California: Association for Computational Linguistics, June 2010, pp. 207–215. [Online]. Available: http://www.aclweb.org/anthology/N10-1024

[11] C.-Y. Lee and J. R. Glass, "A transcription task for crowdsourcing with automatic quality control," in *INTERSPEECH'11*, 2011, pp. 3041–3044.

[12] C. Munteanu, R. Baecker, and G. Penn, "Collaborative editing for improved usefulness and usability of transcript-enhanced webcasts," in *Proceedings of the 2008 Conference on Human Factors in Computing Systems, CHI 2008, 2008, Florence, Italy, April 5-10, 2008*, 2008, pp. 373–382. [Online]. Available: http://doi.acm.org/10.1145/1357054.1357117

[13] H. Kolkhorst, K. Kilgour, S. Stüker, and A. Waibel, "Evaluation of interactive user corrections for lecture transcription," in *2012 International Workshop on Spoken Language Translation, IWSLT 2012, Hong Kong, December 6-7, 2012*, 2012, pp. 217–221.

[14] W. Lasecki, C. Miller, A. Sadilek, A. Abumoussa, D. Borrello, R. S. Kushalnagar, and J. Bigham, "Real-time captioning by groups of non-experts," in *Proceedings of the 25th annual ACM symposium on User interface software and technology - UIST '12*. ACM Press, 2012, pp. 23–34.

[15] M. Wald, "Crowdsourcing correction of speech recognition captioning errors," 2011.

[16] J. Trmal, G. Chen, D. Povey, S. Khudanpur, P. Ghahremani, X. Zhang, V. Manohar, C. Liu, A. Jansen, D. Klakow, D. Yarowsky, and F. Metze, "A keyword search system using open source software," in *Proc. IEEE Workshop on Spoken Language Technology*. South Lake Tahoe, NV; USA: IEEE, Dec. 2014, to appear.

[17] J. G. Fiscus, J. Ajot, J. S. Garofolo, and G. Doddingtion, "Results of the 2006 spoken term detection evaluation," in *Proc. SIGIR*, vol. 7, 2007, pp. 51–57.

[18] D. Can and M. Saraclar, "Lattice indexing for spoken term detection," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 19, no. 8, pp. 2338–2347, Nov 2011.

[19] F. Seide, P. Yu, C. Ma, and E. Chang, "Vocabulary-independent search in spontaneous speech," in *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on*, vol. 1. IEEE, 2004, pp. I–253.

[20] J. Mamou, B. Ramabhadran, and O. Siohan, "Vocabulary independent spoken term detection," in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2007, pp. 615–622.

[21] D. R. Miller, M. Kleber, C.-L. Kao, O. Kimball, T. Colthurst, S. A. Lowe, R. M. Schwartz, and H. Gish, "Rapid and accurate spoken term detection," in *Eighth Annual Conference of the International Speech Communication Association*, 2007.

[22] A. Rousseau, P. Deléglise, and Y. Estève, "Ted-lium: an automatic speech recognition dedicated corpus." in *LREC*, 2012, pp. 125–129.

[23] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, Dec. 2011.

[24] Y. Miao, "Kaldi+pdnn: Building dnn-based ASR systems with kaldi and PDNN," *CoRR*, vol. abs/1401.6984, 2014. [Online]. Available: http://arxiv.org/abs/1401.6984

[25] National Institute of Standards and Technology, "NIST open keyword search 2014 evaluation (OpenKWS14)," http://www.nist.gov/itl/iad/mig/openkws14.cfm.

[26] J. Cain, "introduction to computer science – programming paradigms," http://see.stanford.edu/see/lecturelist.aspx?coll=2d712634-2bf1-4b55-9a3a-ca9d470755ee.