Spoken English Intelligibility Remediation with PocketSphinx Alignment and Feature Extraction Improves Substantially over the State of the Art

Conference Paper · January 2018

DOI: 10.1109/IMCEC.2018.8469649

3 authors:

Yuan Gao
Beijing University of Technology

Brij Mohan Lal Srivastava
National Institute for Research in Computer Science and Control

James Salsman
TalkNicer.com, LLC

SPOKEN ENGLISH INTELLIGIBILITY REMEDIATION WITH POCKETSPHINX ALIGNMENT AND FEATURE EXTRACTION IMPROVES SUBSTANTIALLY OVER THE STATE OF THE ART

Yuan Gao[2] *Brij Mohan Lal Srivastava*[1] *James Salsman*[34]

[1] International Institute of Information Technology, Hyderabad, India
[2] Beijing University of Technology, [3] 17zuoye.com, [4] TalkNicer.com, LLC

ABSTRACT

We use automatic speech recognition to assess spoken English learner pronunciation based on the authentic intelligibility of the learners' spoken responses determined from support vector machine (SVM) classifier or deep learning neural network model predictions of transcription correctness. Using numeric features produced by PocketSphinx alignment mode and many recognition passes searching for the substitution and deletion of each expected phoneme and insertion of unexpected phonemes in sequence, the SVM models achieve 82% agreement with the accuracy of Amazon Mechanical Turk crowdworker transcriptions, up from 75% reported by multiple independent researchers. Using such features with SVM classifier probability prediction models can help computer-aided pronunciation teaching (CAPT) systems provide intelligibility remediation.

Index Terms— phoneme alignment, pronunciation assessment, computer aided language learning, binary features

1. INTRODUCTION

Authentic intelligibility, the ability of listeners to correctly transcribe recorded utterances, initially used for CAPT by [1] and [2], is a better measure of pronunciation assessment for spoken language learners compared to mispronunciations identified by expert pronunciation judges or panels of experts, because such mispronunciations are associated with only 16% of intelligibility problems, according to [3], who state:

> We investigated ... which words are likely to be misrecognized and which words are likely to be marked as pronunciation errors. We found that only 16% of the variability in word-level intelligibility can be explained by the presence of obvious mispronunciations. Words perceived as mispronounced remain intelligible in about half of all cases. At the same time ... annotators were often unable to identify the word when listening to the audio but did not perceive it as mispronounced when presented with its transcription.

This substantial improvement is not yet well understood by most CAPT community. Currently, expert human pronunciation judges assess student performance, often with large inter-rater variability between experts scoring the same utterances. Since most formal mispronunciations do not substantially impede understanding of spoken language, automatic speech recognition CAPT systems trained to approximate the subjective assessments of judges do not perform as well as might be expected after intensive work on the issue by several hundred researchers spanning decades ([4], [5].) While there are many commercial CAPT applications, there is no consensus among speech language pathologists about which of them, if any, work well ([6]).

In high stakes situations, systems imitating subjective assessments of human judges have, for example, prevented native English speakers and trained English language radio announcers from immigrating to Australia ([7], [8]). A more technical related problem with traditional CAPT approaches is that popular pronunciation assessment metrics, primarily **goodness of pronunciation** (GOP) as defined by [9], are quotients with such vaguely specified denominators [10] that they tend to correlate weakly with authentic intelligibility. Earlier work suffers from similar problems.

We are offering remediation of authentic intelligibility for English CAPT to 17zuoye.com's 30 million K-6 English language students in China, and we are deploying the same technology in the Wikimedia Foundation's *Wiktionary* dictionaries along with their phonetics and pronunciation articles in Wikipedia to provide free CAPT assessment and remediation exercises. We are measuring which feedback choices perform the best for student proficiency outcomes, and studying the possibility of using students to provide transcriptions instead of paid crowdworkers.

[1] gyfreedom93@163.com (corresponding author for inquiries in Chinese)
[2] contactbrijmohan@gmail.com (corresponding author for inquiries in Indic languages)
[3] jim@talknicer.com (corresponding author for inquiries in other languages)

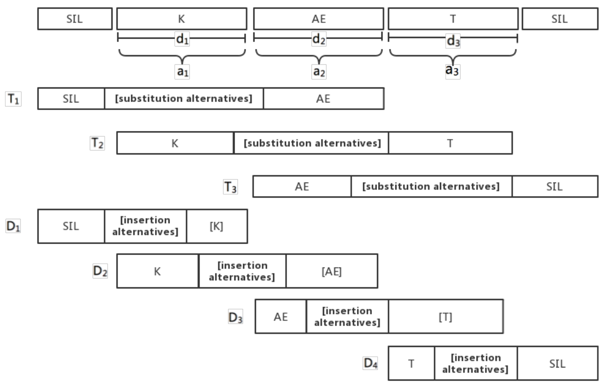

Fig. 1. Feature extraction: The three phonemes of the word 'cat' are aligned, producing durations d_n and acoustic scores a_n. Then several passes of recognition to the audio aligned to groups of three (T_n) and two (D_n) phonemes are used to measure phoneme substitutions, and insertions and deletions, respectively.

2. ADAPTING POCKETSPHINX FOR FEATURE EXTRACTION

We chose to use PocketSphinx[11] system's alignment routines. We tried a two-pass alignment approach over a fixed grammar by using the time endpoints from recognizing the phonemes of the expected utterance in sequence, using a finite state grammar with no alternative or optional components other than silence, defined using a *JSpeech Grammar Format* file. The results for the first pass were discarded, because its purpose was solely to perform *cepstral mean normalization* for adapting to the audio characteristics of the microphone, channel, and noise. We found that grammar-based alignment, which is optimized for speed instead of accuracy, resulted in less correctly predictive features than using a single pass of the alignment API functions, which are only available from the PocketSphinx C API instead of command line invocations.

The results of the alignment are used to select audio sub-segments of the utterance to indicate substitutions of expected phonemes, insertions of unexpected phonemes, deletions of the expected phonemes, and five physiological measures of the vocal tract, in multiple subsequent recognizer passes of each three and two adjacent phonemes at a time. Figure 1 illustrates the non-physiological part of this feature extraction process.

After alignment, we run the recognizer on each sub-segment of the audio corresponding to each three aligned phonemes in sequence, and count how soon the expected phoneme occurs in the n-best recognition results. Then we run the recognizer on each sub-segment corresponding to each two adjacent phonemes in sequence, simultaneously counting how frequently the initial expected phoneme is omitted when searching for the insertion of all 39 phonemes and silence in between the two expected phones.

The substitution detection pass focuses on three adjacent phonemes at a time as located by the alignment routine. For the audio sub-segment of each three adjacent phonemes from the alignment, we use a grammar specifying the first and last of the three as the only options on the ends, with an alternative allowing for any one phoneme (including diphthongs) in the middle. The score, in the range $[0, 1]$, represents how high the expected middle phoneme ranks in the n-best results of all the possible phonemes in between the other two. We ask the recognizer for as many n-best results as possible, because sometimes a truncated grammar result (e.g., only two phonemes instead of three) result, but we often get at least 30 results from the 40 possible phonemes and silence, and sometimes get 70 results. The insertion and deletion pass operates on the audio sub-segments of two adjacent phonemes at a time, using a grammar to look for the first expected phoneme in the front as the only possibility, followed by an optional alternative of any phoneme other than the expected second phoneme counting as insertions, and then followed by the expected second phoneme specified as optional to account for deletion. Each time an insertion or deletion is returned in the n-best results before only the expected two phonemes are returned, the $[0, 1]$ score is reduced.

We also produce each phoneme's duration and the logarithm of its acoustic score from the alignment phase as features in our SVM or DNN classifier feature inputs. For each phoneme, we produce: (1) a duration; (2) an acoustic score from the alignment, corresponding to the numerator of the GOP score of [9]; (3) a $[0, 1]$ score measuring phoneme substitution, and (4) a $[0, 1]$ score measuring insertions and deletions. One final additional insertion and deletion measurement appears at the end of the feature vector for each word; in a multi-word phrase, that final score is shared as identical to the first insertion and deletion measurement of the next word.

As this article was going to press, we added five additional physiological features per phoneme, relating to **place, closedness, roundedness, voicing,** and the proportion of neighboring phonemes less likely. ([12])

We use some non-standard PocketSphinx parameters. We use a frame rate of 65 frames per second instead of 100, because learners are not likely to speak very quickly. We use a *-topn* value of 64 instead of 2. This provides more accurate recognition results at the expense of longer runtime, but our feature extraction system runs in better than real time in a single thread of a 2016 Apple MacBook Air, and on user's browsers as a *pocketsphinx.js* adaptation in JavaScript. We use a *-beam* parameter of 10^{-57}, a *-wbeam* parameter of 10^{-56}, and a *-maxhmmpf* value of -1 for the same reason. We set *-fsgusefiller* to "no" so that optional pauses are not assumed between every word, allowing us to define words comprised of a single CMUBET phoneme without slowdown.

2.1. Compiling featex.c with PocketSphinx

The C source code to perform the feature extraction, **featex.c**, and instructions for compiling and using it are available under the MIT open source code license at:
`https://github.com/jsalsman/featex`

3. USING POCKETSPHINX.JS IN WEB BROWSERS

Feature extraction can take place in web browsers' JavaScript code using the **Emscripten** system of compiling C to JavaScript, and audio recorded in web browsers supporting microphone input. During the initialization process, the browser is checked for microphone availability and the sampling frequency at which it operates. A media source stream is requested to record audio from the microphone, and connected to a recorder thread which listens or stops listening based on browser user interface events. The pocketsphinx.js module is initialized inside a web worker to asynchronously call the alignment and feature extraction modules.

Algorithm 1 Web client algorithm

1: The user presses the 'Record' button.
2: The recorder thread starts listening.
3: The user presses the 'Stop' button.
4: The recorded audio is converted and downsampled if necessary.
5: The extracted feature vector and word is sent to the intelligibility prediction service (see sections 5.1 and 7.)
6: Assessment feedback is provided to the user.

The integrated code and detailed compilation instructions can be found at [13]. For more information and an example of an integrated web browser system, please see [14]. For an example of how such a system might be integrated into Wiktionary, please see [15].

4. OBTAINING TRANSCRIPTIONS OF STUDENT UTTERANCES

We consistently obtained faster responses from Amazon Mechanical Turk when paying \$0.03 per transcript compared to \$0.15. We believe crowdworkers prefer to do low-paying tasks because they are likely to be easier and will cause fewer problems if the work is rejected. We are studying the possibility of using our English learners to provide transcriptions instead of paying crowdworkers, as *bona fide* listening comprehension and typing exercises suitable for assessments in their own right.

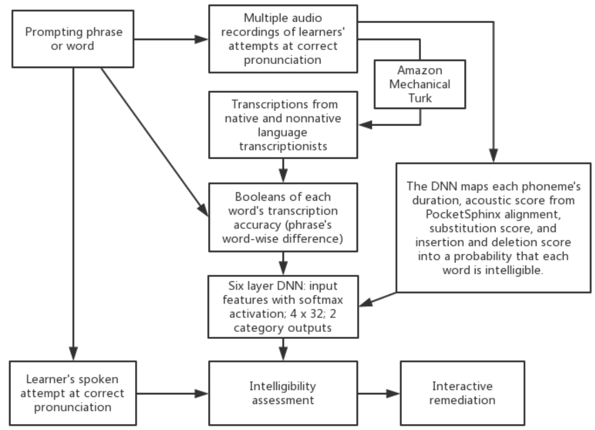

Fig. 2. Predicting intelligibility.

5. PREDICTING INTELLIGIBILITY

Using nine features per phoneme as described above (but not depicted) with support vector machine classification routines from the Python **Scikit-learn SVC** library configured with a radial basis function kernel and probability prediction, we obtain 82% accuracy in predicting the intelligibility of about 700 basic English words in agreement with Amazon Mechanical Turk workers, using about 30 recordings per words and four transcripts per recording. We have measured strong evidence that increasing the number of recordings per word and transcripts per recording can result in very substantial accuracy improvements. We have obtained similar results on longer phrases. Using the four features per phoneme to train a *linear logistic regression* model, we only get 75% accuracy, which was reported by [1] and [2] and the ETS ([3]).

For a client-server system to predict word intelligibility from feature vectors, please see [16].

6. MEASURING THE ACCURACY OF INTELLIGIBILITY ASSESSMENT

When different transcripts of the same utterance of a word show both intelligible and unintelligible results, we measure accuracy as a fraction of the best possible result. For example, if the same utterance was transcribed correctly by three transcriptionists but incorrectly by a fourth, the maximum unadjusted accuracy achievable from predicting that utterance's intelligibility is 75%, so an unadjusted accuracy of 50% is adjusted to be 67%, representing the proportion of the maximum possible accuracy. In practice, the probability of intelligibility is a floating point value in [0, 1], which is typically compared to a threshold, the estimated intelligibility of other words in the same phrase, or both, so the accuracy with which we can

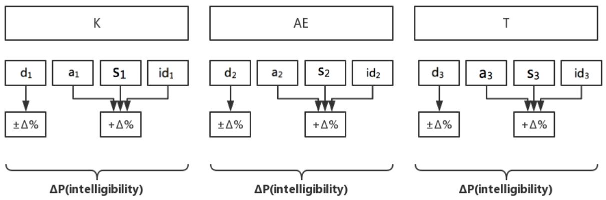

Fig. 3. Determining feedback: Adjusting the feature scores for each phoneme changes the probability of intelligibility of the whole word. The adjustments which make the best changes signal which phoneme(s) need improvement the most.

predict intelligibility by transcriptionists is used as a benchmark by which we can measure the relative utility of different prediction methods.

7. DETERMINING OPTIMAL FEEDBACK

We use the modeled probability of intelligibility of each word in a prompt word or phrase to help students improve their pronunciation by providing audiovisual feedback indicating which word(s) were pronounced the worst. How many words to indicate were not pronounced well after each utterance is an open question.

For words which are not considered sufficiently intelligible, we can use the SVM classifier probability prediction models to determine which identical numerical improvement to each phoneme's non-duration features improves the probability of word intelligibility the most. We can also see how increasing and decreasing each phoneme's duration improves the intelligibility of the word. Such adjustments to the features derived from automatic speech recognition may be more useful as products than sums to identify the specific phoneme(s) most in need of improvement in the less unintelligible word(s). Figure 3 shows how we determine the phoneme-level feedback for each word.

8. CONCLUSION

Using PocketSphinx automatic speech recognition with improved phonetic accuracy features training SVM prediction models can help CAPT systems provide better intelligibility remediation. Researchers and commercial software publishers should try to understand the reasons this technique is superior to the state of the art, and adopt it for improved CAPT outcomes.

9. ACKNOWLEDGMENTS

We thank 17zuoye.com (China), Zzish.com (UK), Prof. Seiichi Nakagawa, the Google Open Source Programs Office, and the Wikimedia Foundation for their kind financial support, suggestions, personnel resources, and educational infrastructure.

10. REFERENCES

[1] Hiroshi Kibishi and Seiichi Nakagawa, "New feature parameters for pronunciation evaluation in english presentations at international conferences," in *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[2] Hiroshi Kibishi, Kuniaki Hirabayashi, and Seiichi Nakagawa, "A statistical method of evaluating the pronunciation proficiency/intelligibility of english presentations by japanese speakers," *ReCALL*, vol. 27, no. 1, pp. 58–83, 2015.

[3] Anastassia Loukina, Melissa Lopez, Keelan Evanini, David Suendermann-Oeft, Alexei V Ivanov, and Klaus Zechner, "Pronunciation accuracy and intelligibility of non-native speech," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[4] Nancy F Chen and Haizhou Li, "Computer-assisted pronunciation training: From pronunciation scoring towards spoken language learning," in *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2016 Asia-Pacific*. IEEE, 2016, pp. 1–7.

[5] J. 2016. Llisterri, "Computer-assisted pronunciation teaching bibliography.," October 2016, [Online; posted 25-October-2016].

[6] Shiran Dudy, Meysam Asgari, and Alexander Kain, "Pronunciation analysis for children with speech sound disorders," in *Engineering in Medicine and Biology Society (EMBC), 2015 37th Annual International Conference of the IEEE*. IEEE, 2015, pp. 5573–5576.

[7] T. 2017. Ferrier, "Australian ex-news reader with English degree fails robot's English test.," August 2017, [Online; posted 9-August-2017].

[8] Australian Associated Press. 2017., "Computer says no: Irish vet fails oral English test needed to stay in Australia," August 2017, [Online; posted 8-August-2017].

[9] Silke M Witt and Steve J Young, "Phone-level pronunciation scoring and assessment for interactive language learning," *Speech communication*, vol. 30, no. 2, pp. 95–108, 2000.

[10] Xiaojun Qian, Helen Meng, and Frank Soong, "A two-pass framework of mispronunciation detection and diagnosis for computer-aided pronunciation training," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 24, no. 6, pp. 1020–1028, 2016.

[11] David Huggins-Daines, Mohit Kumar, Arthur Chan, Alan W Black, Mosur Ravishankar, and Alexander I Rudnicky, "Pocketsphinx: A free, real-time continuous speech recognition system for hand-held devices," in *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*. IEEE, 2006, vol. 1, pp. I–I.

[12] Srikanth Ronanki, James Salsman, and Li Bo, "Pronunciation evaluation and mispronunciation detection using cmu sphinx," in *Proceedings of the 24th International Conference on Computational Linguistics (Mumbai, India) Workshop on Speech and Language Processing Tools in Education*. COLING, 2012, pp. 61–68.

[13] "Pocketsphinx.js," `https://github.com/brijmohan/pocketsphinx.js`, 2017.

[14] "Iremedy," `https://github.com/brijmohan/iremedy`, 2017.

[15] "Live demo," `https://brijmohan.github.io/iremedy/single_line.html`, 2017.

[16] "Pronunciation evaluation service," `https://github.com/brijmohan/proneval-service`, 2017.