Segment-Removal Based Stuttered Speech Remediation

Pierre Arbajian[1], Ayman Hajja[2], Zbigniew W. Raś[1,3,4], and Alicja A. Wieczorkowska[3]

[1] University of North Carolina, Dept. of Computer Science,
9201 University City Blvd., Charlotte, NC 28223, USA
[2] College of Charleston, Dept. of Computer Science,
66 George Street, Charleston, SC 29424 USA
[3] Polish-Japanese Academy of Information Technology,
Koszykowa 86, 02-008 Warsaw, Poland
[4] Warsaw University of Technology, Institute of Computer Science,
Nowowiejska 15/19, 00-665 Warsaw, Poland
parbajia@uncc.edu
hajjaa@cofc.edu
ras@uncc.edu
alicja@poljap.edu.pl

Abstract. Speech remediation by identifying those segments which take away from the substance of the speech content can be performed by correctly identifying portions of speech which can be deleted without diminishing from the speech quality, but rather improving the speech. Speech remediation is especially important when the speech is disfluent as in the case of stuttered speech. In this paper, we describe a stuttered speech remediation approach based on the identification of those segments of speech which when removed would enhance speech understandability in terms of both speech content and speech flow. The approach we adopted consists of first identifying and extracting speech segments that have weak significance due to their low relative intensity, then classifying the segments that should be removed. We trained several classifiers using a large set of inherent and derived features extracted from the audio segments for the purpose of automatically improving stuttered speech quality through providing a second layer of filtering stage. This second layer would discern the audio segments that need to be eliminated from the ones that do not. The resulting speech is then compared to the manually-labeled "gold standard" optimal speech. The comparisons of the resulting enhanced speeches and their manually-labeled optimal speech were favorable and the corresponding tabulated results are presented below.

Keywords: speech analysis, speech remediation, classification

1 Introduction and Background

Quality of life is negatively impacted when individuals are chronically unable to express themselves due to lack of speech fluency. Stuttering, also referred to as

stammering, is a condition that is exhibited in bad fluency of speech. The onset of stuttering generally occurs during childhood years, and in one fourth of cases this speech impediment persists throughout life [12, 9].

Automatic speech disfluency detection offers many advantages, ranging from time saving, constant speech monitoring, reducing the subjectivity of manual disfluency identification [10, 6] as well as a more effective automatic speech recognition. Therefore the identification of "episodes" of stutter will offer firm and actionable information [3, 6, 5]. Reliable identification of speech segments, from the beginning of episode to the end of episode, with pauses, blocks, interjections and hesitations, will allow speech cleanup by ridding the speech of the blocks and interjections, smoothing the hesitation segments and shortening the prolongations [8, 7, 11].

The state of the art is rich with papers which describe various prosodic- and semantic-based machine learning approaches and algorithms for the detection of disfluent speech [1, 10, 4, 13, 14]; however, research work addressing disfluent speech remediation is considerably less common. Our assumption in this research work is that speech intelligibility will be significantly improved by eliminating those segments.

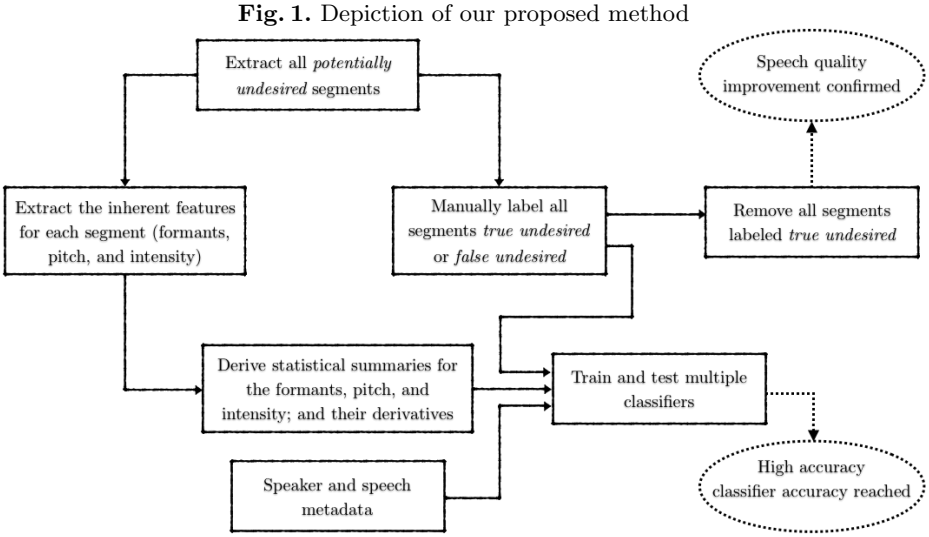
To confirm the assumption that undesired speech segment removal enhances speech quality, we ran a script that eliminates audio segments with low intensity levels using varying intensity thresholds and segment durations, which resulted in an indisputably better and more intelligible audio compared to the original speech. That being said, automatically detecting and removing those *potentially undesired* segments, resulted occasionally in discarding the wrong segments. For that reason, we introduced a second layer of filtering stage in which the *potentially undesired* segments are further classified into *true undesired*, which would need to be discarded; and *false undesired*, which would need to be retained.

In this paper, we examine one explicit type of stuttered speech remediation; remediation in the form of removing undesired speech segments. In the next section, we present an approach to enhance the stuttered speech quality by eliminating the speech segments that contain stuttering blocks, which are defined as the segments of audio in which the person who stutters is unable to produce intelligible sounds. Furthermore, unwanted segment elimination provides the listener with yet another benefit; the speech is shortened (up to 60% reduction) without speeding up individual syllables.

Our algorithm first identifies those segments that have the distinct potential of being undesired in the speech through linguistic intensity and length analysis; detailed explanation will be provided in Section 2. Next, the set of *potentially undesired* audio segments are listened to in order to determine whether they a have semantic value or not, and to determine whether they must be deleted or retained accordingly. The final output of the speech audio file, after removing the segments that were manually labeled *true undesired*, represents the "gold standard", which will be later used to measure the performance of the trained classifier. There are two vital reasons for the phase in which the *potentially undesired* segments are manually labeled; the first reason is to generate the optimal

"gold standard" speech that will be used to evaluate our enhanced stuttered speeches, and the second reason is to aggregate the set of *true undesired* and *false undesired* segments used to build our classifier. Figure 1 below depicts the steps of our proposed method.

Fig. 1. Depiction of our proposed method



The dataset of speeches that we used was obtained from the UCLASS series of recordings. UCLASS, University College London Archive of Stuttered Speech, is a database of stuttered speech recordings with individual speech and speaker metadata. UCLASS contains a collection of over one hundred speeches from which we randomly chose fifteen for our research study.

Our proposed approach begins by scanning each speech for potentially stuttered segments according to varying intensity thresholds and lengths of audio segments. Every *potentially undesired* extracted segment is considered a candidate for removal; however, in numerous cases, the candidate segment may contain semantic speech and therefore should not be omitted from the original speech. Determining which candidate segments are truly undesired candidates and which segments are not, is done through a classification system trained using the previously manually-labeled *true undesired* and *false undesired* segments that were collected beforehand.

The features used in the classifier training include data-points extracted from both the frequency domain such as voice formants and pitch, and from the time domain such as intensity. Additionally, a combination of speaker and speech metadata are also used to improve the trained classifier; a more elaborate description of the list of features is presented in the next section.

The difference between the quality of the "gold standard" of a given speech compared to the quality of an enhanced speech after having every *potentially*

undesired segment being classified is used to evaluate the quality of our system. The results with respect to classifier performance are tabulated in the findings section according to the extraction parameters. Process efficiency improvements were implemented to minimize the manual labeling effort as well as segment deletion redundancy.

2 Proposed Method

The method we adopted for this research consisted of selecting a set of speeches to be examined from a group of disfluent speeches available from UCLASS. UCLASS speeches exhibit a wide variety of speakers and stuttering conditions, yet the vast majority of the speeches are distinctly stuttered and quite disfluent. Our goal is to develop a system capable of automatically analyzing stuttered speeches for the purpose of detecting undesired segments and enhancing the overall speech quality through eliminating the disfluent segments. In this work, we used a *Praat* script for the initial detection of potential candidate segments to be removed from the stuttered speech; the goal is to build a system able to distinguish *true undesired* from *false undesired* for all candidate segments. Next, we provide an elaborate description of the workings of the overall process.

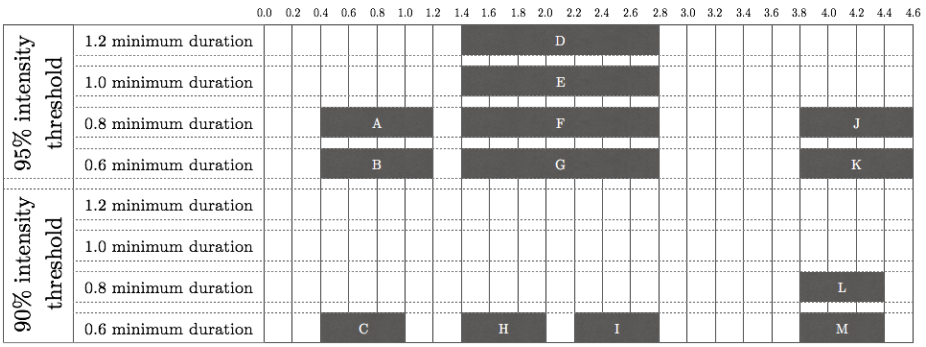
2.1 Extracting Potentially Undesired Candidate Segments

Fifteen speeches (roughly one hour of stuttered speech) from the UCLASS repository are randomly selected as the stuttered speeches of interest, each is scanned with a total of 8 segment extraction parameters. The extraction parameters determine what the sound intensity threshold and the minimum duration of those segments should be. The minimum durations that we used to iterate through our speeches are 0.6, 0.8, 1.0 and 1.2 seconds, and the speech intensity thresholds are at 95% and 90% of the entire speech intensity in decibels; for example, if the average intensity of a given speech is 50 decibels, then the intensity thresholds that correspond to 95% and 90% are 47.5 db and 45 db, respectively. The minimum duration thresholds were chosen by trial and error; low intensity time durations under 0.6 seconds elimination tended to delete intended and legitimate silences and distorted the speech, while minimum durations over 1.2 seconds were too few and statistically no more useful than the 1.2 seconds upper limit we chose.

For each possible pair combination of segment duration and intensity threshold (total of 8 unique combinations), the entire speech is scanned and the *potentially undesired* candidates are detected according to the corresponding pair combination. The minimum segment duration ensures that no excessively short segments are extracted as potential candidates for deletion, which would disrupt the flow of normal speech and unfavorably affect the speech cadence. Note that the extracted segments from each unique pair will not overlap, and that only such overlapping is possible, and rather likely to happen, when examining segments extracted from different pairs. We provide a demonstration of how the

segments may look like in Figure 2. For example, according to Figure 2, extracting potentially undesired segments from the fourth row (0.6 minimum duration with 95% intensity threshold) will result in three different segments (B, G, and K); since segment B's length is 0.8 seconds, which satisfies the minimum duration threshold for the third row, then the same segment will also be identified as a *potentially undesired* segment on the third row; however, since the length of the segment is not 1.0 second (or more), that segment will not be identified on the second (or first) row in Figure 2.

Fig. 2. Extracting potentially undesired candidate segments



Each extracted candidate segment results in a waveform audio file accompanied by five additional vectors that serve as the segment inherent features, three out of which contain the values of formant 1, formant 2, and formant 3, and the remaining two vectors contain the pitch values and the intensity values. All formants were computed with a 50 ms Gaussian analysis window with a 12.5 ms offset. The pitch values were taken at 10 ms interval and computed with the *Praat* software algorithm which performs an acoustic periodicity detection on the basis of an accurate autocorrelation method as described in [2]. The intensity values were computed by squaring then convolving the sound values within a Gaussian analysis window of 50 ms length. The waveform audio files will be later used for manually labeling the candidate segment, and the set of inherent features (in addition to a set of derived features presented in Subsection 2.3) will be used to build our classifier.

2.2 Labeling Potentially Undesired Segments

Following the generation of the segment files, each segment candidate must be manually labeled as: 1) delete and analyze; referred to by Class G, 2) leave in the speech and analyze; referred to by Class B, and 3) no analysis, but delete from the speech, these segments of speech represent external sounds such as when the

interviewer was attempting to speak softly; we chose to exclude these segments from the speech during cleanup because they showed no speech disfluency, so we refrained from including them in the training dataset.

The label "delete and analyze" (G) indicates that the sound is void of semantic meaning and that we would like for our classifier to mark such segments as "delete", or *true undesired*. The "leave in the speech and analyze" label is used for sounds that were long enough and with low intensity, yet the segment sound contained semantic meaning and must not be removed from the original speech; our classifier should label such segment as "retain", or *false undesired*. The "delete and do not analyze" label indicates that a segment is best deleted from the speech but does not represent the type of sound that our classifier must learn to recognize (or take any action about); a common example would be a segment containing the soft voice of the interviewer.

The number of candidate segments generated using our eight unique pairs is substantially large (on average, one segment per second). In an attempt to avoid redundant labeling efforts and minimize the number of segments which must be manually reviewed, we have sorted the list of segments to be reviewed such that if a segment s is marked "delete", then all other segments that are contained within s (start with, or after, s; and end when s ends, or anytime before that) will also be marked as "delete". The reasoning behind our approach is rather clear; if some segment s is void of semantic meaning, then any other segment contained in s must also be void of semantic meaning. Note here that it is not the case that the opposite is true; given that some segment s contains some semantic meaning, we cannot conclude that all other segments contained in s must also contain semantic meaning. This approach has allowed us to reduce review time by a factor of five.

We will use Figure 2 to demonstrate the process explained above. The first step is to sort all *potentially undesired* segments according to their start time, from top to bottom. This means that based on Figure 2, the list of candidate segments will be sorted as follows: A, B, C, D, E, F, G, H, I, J, K, L, and M. The first candidate segment to be listened to is A; if the segment A is marked as *true undesired* (no semantic value), then all other subsets that are contained in A are also marked *true undesired*, which are B and C in our case. Similarly, if segment D is labeled as *true undesired*, then segments E, F, G, H, and I are also labeled *true undesired*. If however, segment D was marked as *false undesired* (contains semantic value), then this only implies that segments E, F, and G are also *false undesired* (since they are essentially the same as segment D), but we cannot conclude that segment H nor segment I are also *false undesired*; therefore, we would need to listen to the two segments H and I to determine what the label of each of them should be.

The dataset used in our classification exhibits two possible class values: G (positive) and and B (negative). The "positive" class indicates that the candidate speech segment must be removed, whereas "negative" class indicates a speech segment that should not be deleted and must remain in the speech.

2.3 Building the Classifiers

In addition to the label feature, the dataset used to train our classifiers will contain two additional types of features: 1) a set of derived attributes extrapolated from the set inherent features, and 2) speaker and speech metadata. The inherent features that are associated with each candidate segment are formant 1, formant 2, and formant 3 (sampled at 50 ms intervals with the time step being 12.5 ms), pitch (sampled at 10 ms), and intensity (sampled at 8 ms). The derived features that are used in our classifier training are all extrapolated from the inherent features and will serve as a set of data-points that provide information about the entire segment as opposed to an exact point in time (due to sampling). We start by calculating the derivatives of each one of the five inherent features; the reason for calculating the derivatives is to assess the variance of our inherent features, which is vital for detecting stutter. Then, for each one of the inherent features and their derivatives, we calculate the average, median, standard deviation, percentiles (25, 50, and 75), minimum value, maximum value, peak-to-peak amplitude measurement, and variance.

The speaker and speech metadata we included in our dataset consisted of those data features provided with the UCLASS dataset; they consisted of the following:

Speaker category metadata: Gender (M/F), Handedness (L, R, not known), Past history of stuttering in the family, Age of stuttering onset, Age at the time of recording, Location of recording was made (clinic, UCL, or home), Recording conditions (quiet room or sound-treated room), Type of therapy received (family based treatment or holistic treatment), Time between therapy and recording time, Speaker had any history of hearing problems (Y/N), Speaker had a history of language problems (Y/N), and Special educational needs (Y/N).

Speech category metadata: Background acoustic noise level (numeric), environmental noise level (numeric), speaker clarity (numeric), interlocutor intrusiveness (numeric).

The resulting dataset consisting of signal statistics, metadata, and labels is used to train eight different classifiers, each for a unique pair combination of minimum duration and intensity threshold. Recall that the minimum duration threshold is the minimum length of *potentially undesired* segments, and that the intensity (percentage) threshold is measured by examining the averaged speech intensity. We used the R Caret package in order to streamline the classifier testing process and cover a wide range of classifiers.

Algorithm 1: Summary of our proposed algorithm

```
initial phase;
for every segment duration and intensity threshold pair do
    extract potentially undesired candidate segments from a given speech;
    manually label each candidate as true undesired or false undesired;
end
training and testing phase;
for every segment duration and intensity threshold pair do
    train a classifier using a portion of the labeled candidate segments;
    test the remaining segments using the classifier built above;
    evaluate the resulting labels by comparing them to the "gold
     standard";
end
```

There are three different courses of action for segments removal that are examined in this research work, each producing a different level of stutter remediation:

1. Remove all candidate segments without human intervention (this approach does not require manual labeling).
2. Remove only the candidate segments that are manually labeled *true undesired* (resulting speech is referred to by the "gold standard").
3. Classify all candidate segments using our trained classifier, and only remove the candidates that are classified *true undesired*.

Each removal course of action yields a certain "enhanced" speech file; the enhanced files are then examined and compared. Our goal is to build a classifier that generates an audio file (course of action number 3) that is as close as possible to the golden standard audio file (generated from course of action number 2).

It is also worth mentioning that in the case of speech remediation, it is more important to avoid deleting semantically meaningful segments than to miss deleting semantically meaningless segments. In other terms, when evaluating our classifier performance, false negatives (deleting what should not be deleted) should be considered more undesirable than false positives (missing to delete segments which must be deleted). There is a large selection of classifiers on could choose from, we leaned towards trying classifiers from several categories: Decision tree classifiers, SVM classifiers, neural network classifiers, and meta algorithms. After trying multiple algorithms from each category we chose the better performing, caret supported, classifiers; those classifiers formed the basis of our subsequent research work. The R classifiers that we continued to train and test are C5.0, Neural Networks, Recursive Partitioning, Random Forests, Polynomial SVM, and AdaBoost.

3 Experiments and Results

Our approach proved effective in eliminating the vast majority of voice blocks when applied to stuttered speech. The results of disfluent speech remediation

were a net elimination of anomalous speech and a reduction in speech length between 60 percent for the most severe stuttered speech to about 25 percent for mildly stuttered speeches.

The effectiveness of our remediation process is highly dependent on the threshold of the speech intensity tolerance during the potentially undesired segments identification. As we have stated earlier, we used two different thresholds for the sound intensity for the initial phase. On the one hand, the lower threshold (90% of average) resulted in a considerably less potentially undesired candidates as shown in Table 1; although most of the undesired candidates extracted using the lower threshold were actually true undesired, there were many instances where stutter segments were not detected (due to the low threshold). On the other hand, the higher threshold (95%) resulted in a much higher number of potentially undesired segments, some of which were false undesired; however, most of the stuttered segments were detected in the first phase. Therefore, the value of the second phase (training and testing phase), which identifies true undesired and false undesired from potentially undesired segments, is most useful when the threshold is high, and when most of the actual stutter segments are detected, even if that means marking some segments as potentially undesired during the first phase while in reality they are false undesired.

Table 1. Segment count

	90% of speech avg sound volume threshold	95% of speech avg sound volume threshold
0.6 seconds	195	211
0.8 seconds	149	159
1.0 seconds	114	130
1.2 seconds	83	93

The second threshold value that we used during the extraction of the potentially undesired sound segments was the minimum time duration, starting with 0.6 seconds and ending with 1.2 seconds. It is worth noting here that we tried shorter duration time frames with little success, causing the comprehensibility of the speech to be diminished; we found that 0.6 second minimum silence duration is the lowest value that can be used in our experiments.

We used two different approaches for treating our data with each classifier. The first approach is to build one classifier for our data after balancing the labels regardless of the minimum duration and intensity threshold values; in other words, we combined all the segments extracted from all eight different unique pairs of minimum duration and intensity threshold, and build our classifier accordingly. The balancing approach we chose to apply for balancing the data is to randomly exclude data tuples of the over-represented class G.

The second approach is to build a single classifier for each unique pair, then average the accuracy and confusion matrix results. Table 2 shows the results obtained using the first approach, while Table 3 shows the results obtained from

using the second approach for the two classifiers that performed the best using separate classifiers for every unique pair; Random Forests and C5.0.

Using 10-fold cross validation, we trained the following six different classifiers: C5.0, Neural Networks, Recursive Partitioning, Random Forests, Polynomial SVM, and AdaBoost. During the training of our classifiers, we used the Caret R package because of its built-in tuning functionality. Next, we list the parameters chosen for each classifier used.

Neural Network (nnet): Tuned model parameters when training on the balanced dataset: size = 5 and decay = 0.1. Tuned model parameters when training on the entire dataset: size = 1 and decay = 0.1. Size represents the number of units in the hidden layer and can be zero if there are skip-layer units. The decay parameter represents weight decay.

Random Forest (rf): Tuned model parameters when training on the balanced dataset: mtry = 2. Tuned model parameters when training on the entire dataset: mtry = 56. The parameter mtry represents the number of variables randomly sampled as candidates at each split.

SVM Polynomial (svmPoly): Tuned model parameters when training on the balanced dataset: degree = 3, scale = 0.01 and C = 1. Tuned model parameters when training on the entire dataset: degree = 3, scale = 0.01 and C = 1. The degree parameter represents the polynomial degree of the kernel function. The scale is the scaling parameter of the polynomial and tangent kernel. C is the cost regularization parameter.

Adaptive Boosting (AdaBag): Tuned model parameters when training on the balanced dataset: mfinal = 100 and maxdepth = 3. Tuned model parameters when training on the entire dataset: mfinal = 150 and maxdepth = 3. The mfinal parameters represents the number of iterations for which boosting is run or the number of trees to use. Maxdepth is the maximum depth of any node of the final tree.

Recursive Partitioning (rpart): Tuned model parameters when training on the balanced dataset: cp = 0.03797468. Tuned model parameters when training on the entire dataset: cp = 0.05970149. Cp is the complexity parameter; any split that does not decrease the overall lack of fit by a factor of cp is not attempted. The main role of this parameter is to save computing time by pruning off splits that are evidently not worthwhile.

As can be seen in Table 2, when we balance our data we reduce the number of samples in the dataset, this big reduction in the number of tuples seems to have a detrimental impact on our neural network classifier model.

4 Conclusion and Future Work

In summary, we described a system which can be successfully used to reduce stuttered speech disfluency by removing certain potentially undesired speech segments. The dataset consisted of speech and speaker metadata, and speech signal statistics. The remedied speeches showed a marked improvement over the original speeches.

Table 2. Results obtained from building a single classifier using all of segments

	True Positive Rate	True Negative Rate	Accuracy
C5.0	100%	94%	97.4%
Neural Networks	75.6%	55.9%	67.7%
Recursive Partitioning	92.6%	94.2%	93.2%
Random Forests	98.4%	91.7%	95.8%
Polynomial SVM	89.9%	75.6%	84.4%
AdaBoost	98.3%	93.8%	96.4%

Table 3. Results obtained from using a single classifier for each pair of minimum duration and intensity threshold, and averaging the results

	Random Forest	C5.0 Classifier
True Positive Rate (Averaged)	95.3 %	94.8%
True Negative Rate (Averaged)	82.3 %	76.6%
Classifier with Highest True Positive Rate	1.2 minimum duration, 90% intensity threshold	0.8 minimum duration, 90% intensity threshold
Classifier with Highest True Negative Rate	1.2 minimum duration, 90% intensity threshold	1.2 minimum duration, 90% intensity threshold
Classifier with Lowest True Positive Rate	1.2 minimum duration, 95% intensity threshold	1.0 minimum duration, 95% intensity threshold
Classifier with Lowest True Negative Rate	1.0 minimum duration, 95% intensity threshold	1.0 minimum duration, 95% intensity threshold

The scope of data collection and the classification is predetermined and designed with the singular objective of determining whether a possible action must be performed or not; in our case, we only collected potentially undesired candidate segments that might be removed, and extracted features which are then used to classify the segments with the sole purpose of deciding whether a segment needs be removed or not.

A practical application of a system such as the one presented in this work is to remedy a disfluent speech with episodes of blocks for a speaker who wishes to transform his (or her) speech to one that is easy to listen to, hence better understood. Another potential application would be to provide speech therapist with a tool to aid them in their treatment plan for patients with speech disorder.

The concept of single action based intelligent solutions can help systems utilize compartmentalized machine learning and classification solutions. Such scope-specific machine-learned actions can provide lightweight and fast independent action prediction tool that can be chained together for more complex tasks. The process of utilizing the use of multiple perspectives can be employed to optimize and devise an adaptive approach to data gathering, where the data collection method and scope are optimized according to the metadata and condition of the subject on hand.

In future work, we plan on utilizing various extraction thresholds to fine-tune the feature collection to the speech condition, which would better streamline the remediation process. Another extension to disfluent speech remediation would be to eliminate speech interjections by identifying the voiced and unvoiced segments of speech and singling out those voiced sound segments with interjection episode characteristics for removal.

References

1. Ai, O. C., Hariharan, M., Yaacob, S., Chee, L. S.: Classification of speech dysfluencies with MFCC and LPCC features. In: Expert Systems with Applications, 39(2), 2157-2165 (2012).
2. Boersma, Paul.: Accurate short-term analysis of the fundamental frequency and the harmonics-to-noise ratio of a sampled sound. In: Proceedings of the institute of phonetic sciences. Vol. 17. No. 1193. (1993).
3. Chee, L. S., Ai, O. C., Yaacob, S.: Overview of automatic stuttering recognition system. In: Proc. International Conference on Man-Machine Systems, no. October, Batu Ferringhi, Penang Malaysia, 1-6 (2009).
4. Craig, A., Hancock, K., Tran, Y., Craig, M., Peters, K.: Epidemiology of stuttering in the community across the entire life span. In: Journal of Speech, Language, and Hearing Research, 45(6), 1097-1105 (2002).
5. Fook, C. Y., Muthusamy, H., Chee, L. S., Yaacob, S. B., Adom, A. H. B.: Comparison of speech parameterization techniques for the classification of speech disfluencies. In: Turkish Journal of Electrical Engineering & Computer Sciences, 21, no. Sup. 1 (2013).
6. Hariharan, M., Chee, L. S., Ai, O. C., Yaacob, S.: Classification of speech dysfluencies using LPC based parameterization techniques. In: Journal of medical systems, 36(3), 1821-1830 (2012).
7. Honal, M., Schultz, T.: Automatic Disfluency Removal on Recognized Spontaneous Speech-Rapid Adaptation to Speaker Dependent Disfluencies. In: ICASSP (1), 969-972 (2005).
8. Honal, M., Schultz, T.: Correction of disfluencies in spontaneous speech using a noisy-channel approach. In: Interspeech (2003).
9. Howell, P., Davis, S., Bartrip, J.: The UCLASS archive of stuttered speech. In: Journal of Speech, Language, and Hearing Research, 52(2), 556-569 (2009).
10. Ravi Kumar K. M., Ganesan, S.: Comparison of multidimensional MFCC feature vectors for objective assessment of stuttered disfluencies. In: International Journal of Advanced Networking Applications, 2(05), 854-860 (2011).
11. Lease, M., Johnson, M., Charniak, E.: Recognizing disfluencies in conversational speech. In: IEEE Transactions on Audio, Speech, and Language Processing, 14(5), 1566-1573 (2006).
12. Liu, Y., Shriberg, E., Stolcke, A., Harper, M. P.: Comparing HMM, maximum entropy, and conditional random fields for disfluency detection. In: Interspeech, 3313-3316 (2005).
13. Ravikumar, K. M., Rajagopal, R., Nagaraj, H. C.: An Approach for Objective Assessment of Stuttered Speech Using MFCC. In: The International Congress for global Science and Technology, p. 19 (2009).
14. Swietlicka, I., Kuniszyk-Jóźkowiak, W., Smołka, E.: Hierarchical ANN system for stuttering identification. In: Computer Speech & Language, 27(1), 228-242 (2013).