A Pipeline Consisting of Pattern Recognition and Finite Automata for Recognizing VCV Productions in the Study of Vocal Hyperfunction

Gbenga Omotara
ViGIR Lab - Dept. of EECS
University of Missouri
Columbia, USA
goowfd@mail.missouri.edu

Mark Berardi
Dept. of Psychiatry and Psychotherapy
University Hospital Bonn
Bonn, Germany
mark.berardi@ukbonn.de

Maria Dietrich
Dept. of Psychiatry and Psychotherapy
University Hospital Bonn
Bonn, Germany
maria.dietrich@ukbonn.de

G. N. DeSouza
ViGIR Lab - Dept. of EECS
University of Missouri
Columbia, USA
DeSouzaG@missouri.edu

***Abstract*—. Relative fundamental frequency (RFF) is an acoustic measure used to quantify vocal effort in voice science. Since it seeks to capture transitions between (i.e. to/from) steady-state vowels and unvoiced consonants, any machine learning approach to recognize patterns in these transitions should require time properties capable of identifying the sequence of phonemes. At the same time, Neural Networks (NN) have become a ubiquitous solution for data-driven problems, and Recursive NNs (RNN) provide a time-series schema to address time-dependent problems. Indeed, typical Neural Network solutions require either a time-series schema like in RNN or some spectral transformation to be able to handle time-dependent data. In this study, we decided to ignore – at least momentarily – any time-series dependency of the data and employed a simple NN to classify elements of the speech. Later, a State-Machine was used to identify their sequence with the purpose of localizing the transitions between voiced and unvoiced sounds in vowel-consonant-vowel (VCV) productions. The goal of this study was to demonstrate that a pipeline consisting of time-agnostic (Neural Network) and time-dependent (State Machine) components can be used to recognize time-dependent patterns in VCV productions.**

***Index Terms*—Neural Networks, Recursive NN, RFF, time-series, vocal hyperfunction**

I. Introduction

Relative fundamental frequency (RFF) was developed as an acoustic measure to quantify baseline laryngeal tensions in terms of short-term changes in fundamental frequency [7]. Research has shown that RFF differs between individuals with and without vocal hyperfunction [1], and while different individuals may produce vibrations at different frequencies, the relationship between fundamental frequencies (f_0) at different moments of the vowel-consonant-vowel (VCV) productions promotes the viability of RFF to capture information relating to muscle tension [7] – and to our knowledge, there has not been any work in the literature to date that identified regional accents or native language of the speaker of the VCV productions as factors in RFF calculation.

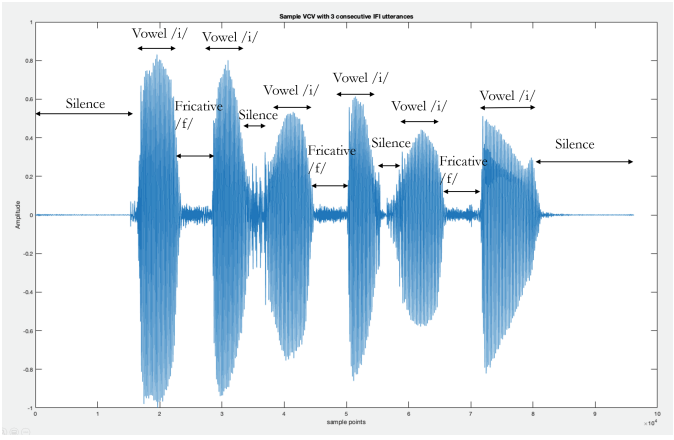

Fig. 1. Sample input data with three consecutive vowel-consonant-vowel productions.

So, in order to calculate RFF, a trained person assesses acoustic waveforms using an acoustic analysis software such as Praat [6], [17]. The waveforms, like in Fig. 1, must contain transitions between voiced sonorants (vowels) and voiceless consonants (fricatives), as the clinician examines them to locate 10 glottal cycles in the voiced sonorant immediately preceding the voiceless consonant (offset cycles), as well as 10 glottal cycles in a voiced sonorant immediately following the voiceless consonant (onset cycles) as shown in Fig. 2. Once the onset and offset cycles are identified, RFF is computed as the relative change in the frequency of the glottal cycles from the onset or offset with respect to their steady-states. In that sense, determining the exact voicing offset and onset is a crucial and yet challenging problem since research has shown that in order

to extract meaningful RFF at least six samples are required (six onsets and six offsets) from VCV productions such as in /afa afa afa ifi ifi ifi ufu ufu ufu/ [1]. Recently, researchers

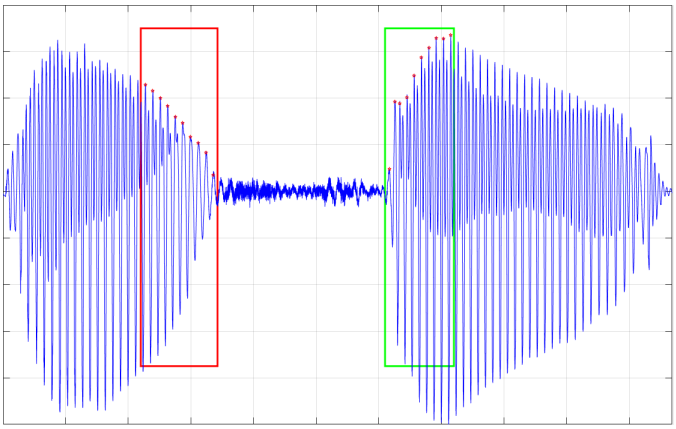

Fig. 2. Sample relative fundamental frequency utterance with one vowel-consonant-vowel production

in [2] have developed a tool for calculating RFF using an automated MATLAB program named aRFF-AP. Authors in [2] were interested in how fundamental frequency estimation and sample characteristics impacted the relationship between manual and semi-automated RFF estimates. When compared to the manual RFF method, their algorithm achieved high correspondence with root mean square error (RMSE) of 0.28 semitones (ST) and mean bias error (MBE) of 0.01 ST. While they achieved good results, after testing their method on our own data set, it was observed that 43% of the time their system required human intervention to correctly detect fricatives in VCV productions, even though it required only 4% intervention on another one of our data sets with continuous speech [9]. It is important to mention some limitations in our data set pertaining to the training of participants: to speak slowly with pauses in between sets (/afa afa afa/); and without vocal fry. We also did not use a pop screen during data collection, which we sought to amend by performing a 120 Hz highpass filtering on individual audio files. Nonetheless, this aspect of the aRFF-AP, while expected to be "semi-automated", caused too many human interventions and sparked our interest in developing a more reliable tool using Machine Learning and Finite Automata.

In typical time-series problems, the continuous nature of the signals challenges us to find an appropriate representation of the time-dependency of the data when setting up learning tasks. Using amplitude-domain, frequency-domain or hybrid-domain [20] features are the three possible choices when it comes to feature extraction and selection. When using Neural Networks (NNs) as the learning paradigm, another choice to be made is whether to use convolution, and then the dimension on which to perform it: e.g. 1-D and 2-D convolutions are typically done on the raw signal or on the spectrogram, and used as inputs to the NN-based classifier [13] [20]. Typically, Hidden Markov Models, Markov Chains, Monte Carlo Simulation, and Recurrent Neural Networks are useful paradigms to capture time-dependent patterns in input signals [16]. In this work, we got away with not using such paradigms and instead relied on the periodic nature of the phonemes in the speech for extracting time-independent windows which were classified as distinct elements of the same speech. Next, we used a finite automata to actually predict the time-dependent relationships between phonemes. This pipeline was applied to VCV productions (e.g /afa/) which, as explained earlier, are useful in speech science for estimation of vocal strain. In our approach, a neural network was used to analyze and identify each distinct phoneme in the VCV productions contained in a comprehensive dataset from a larger study [5]. The VCV productions were generated by ninety-two (92) female participants with and without symptoms of vocal fatigue and no phono-trauma. Data retrieved from four participants were unusable and thus discarded following collection, leaving only eighty-eight participants. Twelve out of whom came for repeat sessions. The subjects were native English speakers between ages 21 and 39 years – they repeated a series of three consectutive VCV's at a time e.g (/afa afa afa ifi ifi ifi ufu ufu ufu/) and three times throughout the course of the experiment (beginning, middle, end). These productions are typical utterances in voice analysis using RFF as they contain the necessary transitions between steady state vowels and fricatives [15]. In this sense, neural networks should be well suited for the task of recognizing phonemes given their ability to extract salient features that can aid in the individual, time-independent classification of vowels and consonants in any speech production. Our system also employs a state machine to process the neural network predictions and extract the temporal organization of these phonemes. That is, the emerging output labels from the NN were fed into the state machine and the output of the state machine is the location in time of the fricatives given the input signal corresponding to the VCV production. We tested our method by applying the proposed pipeline to audio signals containing three (3) consecutive VCV's. The goal was to "segment" the audio file into vowels, fricatives and regions of silence. The accuracy of the system was measured based on the classifier accuracy, as well as the accuracy of the state machine in fricative detection. The main contribution of this work is, a pipeline consisting of classifier and finite automata to detect voice productions and the location of their elements (phonemes).

II. Background and Related Work

A. Vocal Hyperfunction (VH)

Vocal hyperfunction (VH) has been defined as "conditions of abuse and/or misuse of the vocal mechanism due to excessive and/or 'imbalanced' muscular forces" [18] [19]. It is associated with many instances of voice disorders [19] which affect populations such as school teachers [11]. Research shows that vocal hyperfunction is linked with laryngeal muscle tension, which establishes a relationship between VH and RFF [2]. Both acoustic data [9], [2] and surface electromyography (sEMG) data [5], [8] collected from the anterior neck surface

have been used in computationally oriented research for the study of vocal dysfunctions.

B. Time-Series Data for Studying Vocal Hyperfunction and Other Speech Related Tasks

Some studies have been conducted using sEMG data to classify signals emerging from vocally fatigued individuals and vocally healthy individuals [5] [8]. Both [5] and [8] used pattern recognition methods to deal with time-series data in the form of sEMG. In [8], a technique called Guided Under-determined Source Signal Separation or GUSSS was used to detect whether or not a previously learned, unique signature is present in the sEMG signal. This signature is injected into the signal and the GUSSS method returns a ratio. A small ratio indicates that the signature is likely present and a high ratio indicates its absence. In [5], features were extracted from the sEMG signal to train a Support Vector Machine (SVM) classifier which discriminates feature vectors arising from sEMG captured from fatigued and non-fatigued individuals using a leave-one out approach. Both [9] and [2] (the semi-automated algorithm, aRFF-AP) approached the problem of vocal hyperfunction using acoustic data, however, in [2], classical signal processing methods were used. In that case, the authors used high-to-low energy ratios in the acoustic waveform to locate fricatives. The fricative locations were used in latter steps of their algorithm to compute the RFFs. It is important to note that the work done in [2] served as the rationale for the development of [9],which uses a more traditional machine learning approach. The proposed pipeline utilized a Hidden Markov Model toolkit for Speech Recognition (HTK) which identified fricatives in the acoustic waveform. Similar to the other systems, once the fricatives are identified and located, they are used in subsequent steps to calculate RFF using the onset and offset cycles around them. The authors in [20] use a 1D CNN (1-Dimensional convolutional neural networks) trained on the TIMIT speech corpus for identifying fricatives. They compute the posterior probability for the event that the sample at the center of a given speech segment belongs to one of three phoneme classes. They achieve 92.79% unweighted average recall on the TIMIT core test set. The authors point out that methods using 1D CNNs or RNNs require some temporal context (both future and past) of the speech for the network to make a proper prediction. Here, we tackle the problem with a much simpler approach, given the limitations of our dataset in comparison to the speech examples in the TIMIT corpus; while also bypassing the nature of deep neural networks in terms of their stringent requirements in terms of large datasets, computation requirements for learning, etc. Indeed, our method uses a simple multi-layer perceptron, agnostic to context, leaving to the state machine the tasks of capturing the temporal dynamics and needed context.

III. Proposed Method

We propose a pipeline consisting of a pair of time-agnostic (Neural Network) and time-dependent (State Machine) components for classifying speech – i.e. time-series data. The first step in the construction of the classifier required some consideration on the best practice for sampling and labelling the data used for training and testing of the NN. In that sense, intervals in the waveforms (Figure 1) were manually labeled as: vowel, fricative, and silence. Next, multiple windows within those intervals were automatically sampled and assigned the same labels.

In order to automatically sample the labeled intervals, a cycle detection algorithm was applied to the input signal to determine the beginning of each cycle of the signal. The starting points of the cycles were used as markers to set the beginning of the fixed-length windows (1700 sample points or approximately 38 ms) which were used as the actual sampled data for training and testing. The idea here was to generate windows of the signal derived from the same intervals of the speech (vowel, consonant or silence) and hence which retain a strong resemblance to each other. Since each subject's voice has approximately the same frequency (pitch), the fix-length windows should also contain approximately the same number of cycles. Performing the window extraction in this fashion was beneficial because it: 1) increased the number of training samples; 2) maintained the properties of intervals with same label (i.e. vowel, fricative, or silence); and 3) resulted in three corresponding sets of samples with low within-class scatter. Next, a Neural Network was trained to recognize the windows of speech independently of their temporal arrangement within the speech (time-agnostic). The state machine received the predicted class from the NN as input and performed their temporal analysis. Since some sequences of vowel, fricative and silence are not allowed (i.e. are not expected in the productions), the state machine can further reject and/or ignore false predictions provided by the classifier – increasing the accuracy in the detection of VCV productions and hence the better localization of fricatives in each one of the three consecutive VCV's in the waveform (e.g. /ifi ifi ifi/ in Figure 1). Figure 3 shows an overview of the proposed pipeline.

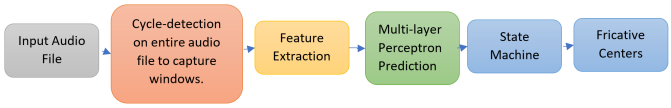

Fig. 3. Overview of the pipeline.

A. Relative Fundamental Frequency (RFF)

RFF is computed in semitones using the following equation:

$$RFF\,(ST) = 39.86 \times \log_{10}\left(\frac{f_o}{f_o^{ref}}\right) \tag{1}$$

For the voice offset, the 1^{st} cycle is the steady-state reference cycle f_o^{ref} in the RFF calculation [7]. It is used to normalize all offset RFF values. The 10^{th} cycle here is the cycle closest to the fricative. Ten offset RFF values calculated in semitones can be acquired using (1) [2]. For the voice

onsets, the 1^{st} cycle is the cycle closest to the fricative and the 10^{th} cycle is the steady-state reference cycle f_o^{ref} – similar to the case with the offsets. Also, ten onset RFF values can be acquired, just like for the offsets. The RFF of the reference will always be zero due to the property of logarithms.

B. Neural Network (NN)

The classifier used in this research was a simple multilayer perceptron with two hidden layers. The NN architecture was as follows: 42 neurons on the input layer followed by two hidden layers with 128 and 32 neurons respectively. Finally, the output layer consisted of 3 neurons for each one of our 3 classes – vowel, fricative and silence. The NN architecture is depicted in Figure 4. The network architecture can be represented in canonical form with the following equation:

$$y_n(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{k=1}^{N} \mathbf{w}_{nk}^{(3)}\, h\left(\sum_{j=1}^{M} \mathbf{w}_{kj}^{(2)}\, h\left(\sum_{i=1}^{D} \mathbf{w}_{ji}^{(1)} x_i + \mathbf{w}_{j0}^{(1)}\right) + \mathbf{w}_{k0}^{(2)}\right) + \mathbf{w}_{n0}^{(3)}\right) \quad (2)$$

where N=32, M=128 and D=42. The vector **w** contains the combined weight and bias parameters. The network uses a ReLU activation function on the hidden layers and a SoftMax on the output layer.

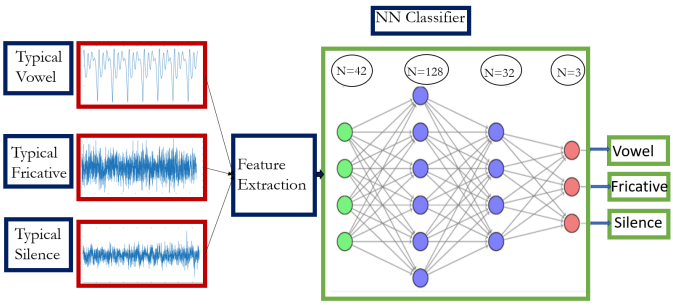

Fig. 4. Neural Network Architecture

C. State Machine (SM)

A finite automata or a state machine is a mathematical model of computation [12]. It processes a sequence of regular expressions as input and it changes its states as it recognizes the elements in the regular expression. So, a finite automata has a finite number of states and hence it can be in exactly one of its finite set of states at any given time. The state transition in a state machine is triggered as a response to an input.

The proposed state machine consisted of a total of eight (8) states with three (3) of those states being "terminal states" and five (5) "internal states". As the machine parses a sequence of labels extracted by the NN from the VCV productions, the terminal states recognize the end of a sequence of windows with the same label (e.g. a sequence of vowel windows in the speech), at which point, they output the label of that entire sequence. On the other hand, internal states are used to accept a certain level of false-predictions from the NN. For example, if a single window is identified as fricative by the NN during a sequence of vowel windows, the SM can reject such fricative window as "spurious noise". These internal states behave as "tolerance" states to make the state machine robust to spurious, erroneous predictions by the NN in the sequence of windows – and hence, correcting misclassifications from the NN classifier. The number of acceptable spurious errors in classification by the NN and consequently the required number of internal states were determined after an analysis of the performance of the NN and the probability of errors in classification to happen in sequence. It was observed that the NN very rarely makes two mistakes in a row, and even more rarely makes three mistakes. Also, those same misclassifications happened more frequently, and only affected the parsing of the VCV productions during vowels. So, the internal states were set accordingly, as explained further below. Figure 5 depicts the structure of the proposed state machine. The arrows in the figure represent the state transitions while the boxes represent the states of the machine. The terminal states are indicated by "VOWEL1","FRICATIVE", and "VOWEL2", corresponding to the offset vowel, fricative, and onset vowel, respectively – 'silence', even though an output class from the NN, did not need to be a terminal state of the SM since it is only used to reset the same SM to its initial state. Due to the way in which the SM was constructed, the same group of four 'tolerance' states were able to capture spurious errors in classification during the parsing of both offset and onset vowels. These internal states are marked as "EV1", "EV2", EV3", and "EV4" (which stands for "expecting vowel 1" etc.) in Figure 5. A fifth internal state, "WAITV2" (wait vowel 2), was used for the same purpose during the parsing of fricatives. The vertices of the SM – i.e. the inputs of the SM that cause it to transition between states – are the three output classes of the NN, in Figure 5: VOW, SIL, and FRIC. Once the machine reaches a terminal state, it returns to the initial state where the process of recognizing the next VCV in the speech is retaken.

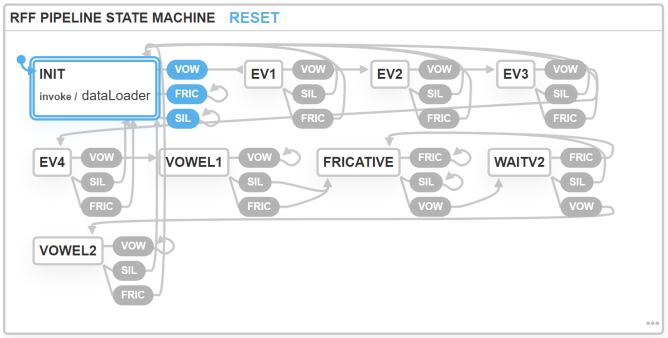

Fig. 5. State Machine Structure

IV. EXPERIMENTAL EVALUATION

We used the nonsense VCV production (/afa/, /ifi/ and, /ufu/) [14] for this research. As mentioned earlier, these VCV's capture the natural transitions between steady state vowels and fricatives which are necessary for measuring RFF. The sampling frequency of the audio recordings was set to 44.1kHz. The Neural Network was trained using the following features: zero-crossing, median frequency, difference autoregressive coefficients, autoregressive coefficients, histogram values, cepstral coefficients, mean absolute value, modified mean absolute value 1, modified mean absolute value 2, root mean square value, standard deviation, sum of squared integral, temporal moment, variance, waveform length and Mel-frequency cepstral coefficients.

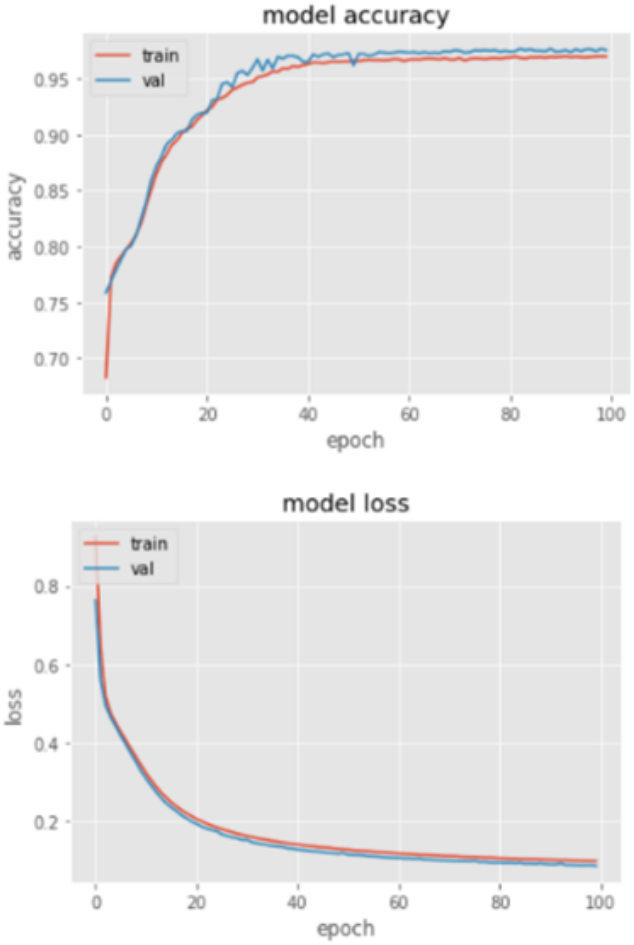

Fig. 6. Training curve showing the model accuracy and loss for both training (red) and validation (blue)

A total of 14958 samples were generated after applying the window sampling algorithm described above. A total of 4986 samples belonged to class Vowel; 4986 to class Fricative; and 4986 to class Silence. The 14958 samples were partitioned into training (80%), and testing (20%). The training set was further split into training (80%) and validation (20%) – using the same proportion between the classes, i.e. 1/3 for each.

During training, the Neural Network achieved a high accuracy (97% after about 40 epochs – Figure 6-a) and low loss (less than 0.1 after 60 epochs – Figure 6-b) across all three phonetic classes. The network also performed very well (97.16% accuracy) in testing with never-encountered data samples, as depicted by the confusion matrix in Figure 7.

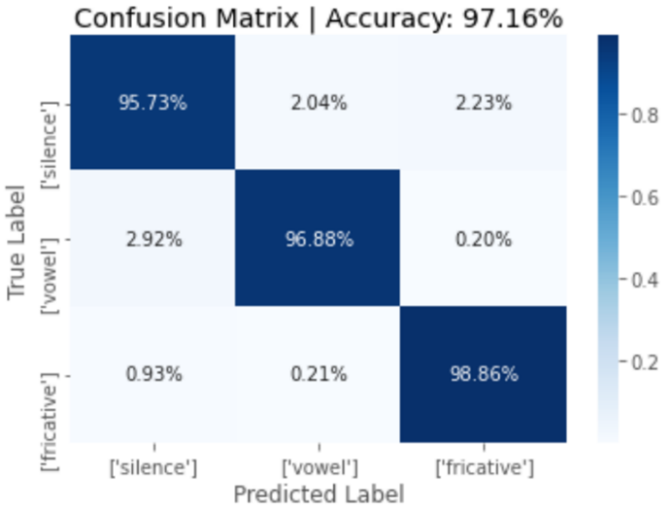

Fig. 7. Confusion Matrix : Model performance in testing

As we mentioned at the beginning, we also compared the results of our pipeline to the method in [2] – a signal-processing based method for RFF calculation, named aRFF-AP, and that uses an (semi-)automated MATLAB program. Our first metric for comparison was based on the percentage of correct detection of fricatives in the VCV – i.e. vis-a-vis the number of required human interventions. In that sense, we tested both systems on 875 audio files each containing the same 3 VCV productions. As Table I shows, our pipeline performed clearly better than the aRFF-AP algorithm with a 40% improvement with respect to automatic detection (i.e. reduction of human interventions). This is presented in the first row of Table I where one or more misdetections of the three fricatives in an individual audio file would require human intervention of the entire audio file. Now, in terms of percentage of misdetection of fricative instances, second row in Table I, we observed an 6.2% improvement with respect to the aRFF-AP.

TABLE I
COMPARISON BETWEEN TECHNIQUES BASED ON % OF ACCEPTED SAMPLES.

	Proposed Pipeline	***Automated RFF (aRFF-AP)***
Individual Audio File	79%	57%
Fricative Instances	86%	81%

Since the ultimate goal of a system for automatic calculation of RFF could be equated to the problem of locating the center of the fricatives – so that the onset and offset boundaries can be derived from those same detected centers – we also

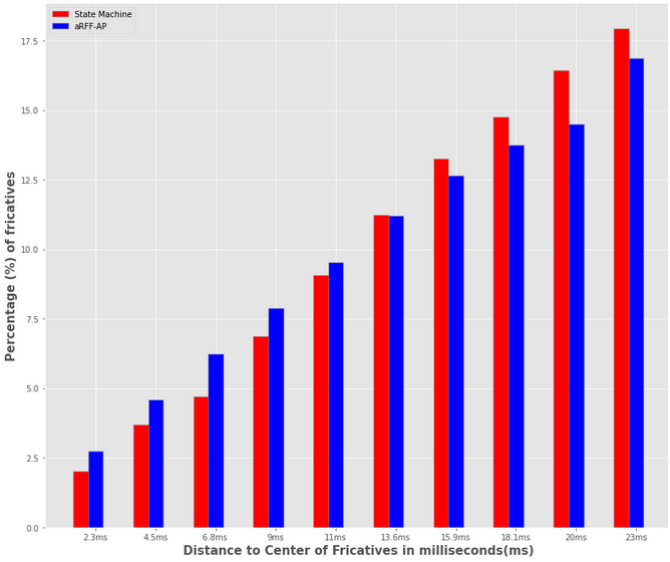

Fig. 8. Comparison between techniques based on closeness to the fricative center.

examined the performance of each method based on how close the detected center of the fricatives were to the actual center of the fricative intervals in approximately 300 of the 875 waveforms. In Figure 8, we summarize the results from this measurement using a histogram with the percentage of samples (Y axis) that fell within a given number of samples away from the actual center of the fricative (X axis). As the Figure shows, we observed that the aRFF-AP algorithm (blue), though comparable to the proposed pipeline (red), localized the fricatives closer to the center of the fricative interval than our method. However, it is important to notice two things: 1) the two approaches differed by a very small number of samples – i.e. about 15 samples, or 2.5% of the samples, in the worst case scenario found in the last column of the graph in Figure 8; and 2) the typical length of a fricative is 130 ms, so the distances between predicted and actual centers of the fricatives on the Y axis of Figure 8 are equivalent (also in the worst case scenario) to only about 1/4 to 1/3 of the length of the fricative – which still falls well enough within the fricative interval for any reliable calculation of RFFs.

V. CONCLUSIONS

A completely automated algorithm for RFF calculation should not require any human intervention. Also, whether such algorithm directly returns the boundaries of the offset and onset cycles, or the center of the fricatives, its performance is also determined by how accurate and reliable those detections are, so they lead to equally successful RFF calculations. In this paper, we compared our proposed approach to the aRFF-AP method [2] for the cases of VCV productions. Our proposed pipeline clearly outperforms the aRFF-AP algorithm for both individual audio files and individual fricative instances needing manual intervention, and presented comparable performance in terms of distance of the center of the fricative. We have also demonstrated that a pipeline consisting of time-agnostic and time-dependent components can successfully recognize patterns of VCV productions. In the future, we will expand the application of the proposed methods to the detection of VCV productions in more natural, every-day sentences.

ACKNOWLEDGMENT

Research in this publication was supported by the National Institute on Deafness and Other Communication Disorders of the National Institute of Health under Award Numbers R15DC015335 and R01DC018026. The content is solely the responsibility of the authors and does not necessarily represent the official view of the National Institute of Health. We thank Erin Tippit for manual RFF analyses.

REFERENCES

[1] T. Eadie and C. Stepp, "Acoustic correlate of vocal effort in spasmodic dysphonia," Annals of Otology, Rhinology & Laryngology 122(3):169-176. 2013.

[2] J. Vojtech, et al., "Refining algorithmic estimation of relative fundamental frequency: Accounting for sample characteristics and fundamental frequency estimation method," Journal of the Acoustical Society of America; 146, 3184-3202 2019.

[3] G. Dorffner, "Neural networks for time series forecasting," Neural Network World 1996.

[4] M. Dörfler, R. Bammer and T. Grill, "Inside the spectrogram: convolutional neural networks in audio processing," International Conference on Sampling Theory and Applications (SampTA); 152-155, 10.1109/SAMPTA.2017.8024472. 2017

[5] Y. Gao, M. Dietrich, and G. N. DeSouza, "Classification of vocal fatigue using sEMG: Data imbalance, normalization, and the role of Vocal Fatigue Index scores," Applied Sciences 2021;11(10):4335

[6] P. Boersma and D. Weenink, "Praat, A system for doing phonetics by computer," Glot Int. 5(9-10), 341–345. 2018

[7] Y. Lien, et al., "Validation of an algorithm for semi-automated estimation of voice relative fundamental frequency," Annals of Otology, Rhinology and Laryngology. 126(10), 712—716. 2017

[8] N. R. Smith, T. Klongtruagrok, G. N. DeSouza, C. R. Shyu, M. Dietrich and M. P. Page, "Non-invasive ambulatory monitoring of complex sEMG patterns and its potential application in the detection of vocal dysfunctions," 2014 IEEE 16th International Conference on e-Health Networking, Applications and Services (Healthcom), pp. 447-452, doi: 10.1109/HealthCom.2014.7001884. 2014.

[9] M. Berardi, E. Tippit, G. N. DeSouza, and M. Dietrich, "Automatic segmentation of relative fundamental frequency from continuous speech," The 14th International Conference on Advances in Quantitative Laryngology, Voice and Speech Research (AQL). pp. 64-65, 2021.

[10] N. P. Solomon, "Vocal fatigue and its relation to vocal hyperfunction," International Journal of Speech-Language Pathology, vol. 10, pp. 254-266, 2008.

[11] N. Roy, R. M. Merrill, S. Thibeault, S. D. Gray and E. M. Smith, "Voice disorders in teachers and the general population: Effects on work performance attendance and future career choices," Journal of Speech Language and Hearing Research, vol. 47, pp. 1092-4388, 2004.

[12] J. A. Anderson and T. J. Head, "Automata theory with modern applications," Cambridge University Press. ISBN 978-0-521-84887-9. 2006.

[13] H. Meng, T. Yan, F. Yuan and H. Wei, "Speech emotion recognition from 3D log-Mel spectrograms with deep learning network," in IEEE Access, vol. 7, pp. 125868-125881, doi: 10.1109/ACCESS.2019.2938007. 2019.

[14] Y. Gao, M. Dietrich, M. Pfeiffer and G. N. DeSouza, "Classification of sEMG signals for the detection of vocal fatigue based on VFI Scores," 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 5014-5017, doi: 10.1109/EMBC.2018.8513224. 2018.

[15] Y. Lien, C. Gattuccio and C. Stepp, "Effects of phonetic context on relative fundamental frequency", Journal of Speech, Language, and Hearing Research, vol. 57, pp. 1259-1267, 2014.

[16] O. Almqvist, "A comparative study between algorithms for time series forecasting on customer prediction: An investigation into the performance of ARIMA, RNN, LSTM, TCN and HMM,". 2019.

[17] V. S. McKenna and C. Stepp, "The relationship between acoustical and perceptual measures of vocal effort," Journal of the Acoustical Society of America. 2018;144(3):1643. doi:10.1121/1.5055234

[18] R. E. Hillman, E. B. Holmberg, J. S. Perkell ,M. Walsh, and C. Vaughan, "Objective assessment of vocal hyperfunction: an experimental framework and initial results," Journal of Speech, Language, and Hearing Research. 1989 Jun;32(2):373-92. doi: 10.1044/jshr.3202.373. PMID: 2739390.

[19] C. E. Stepp, R. E. Hillman and J. T. Heaton, "The impact of vocal hyperfunction on relative fundamental frequency during voicing offset and onset," Journal of Speech, Language, and Hearing Research. 2010 Oct;53(5):1220-6. doi: 10.1044/1092-4388(2010/09-0234). Epub 2010 Jul 19. PMID: 20643798.

[20] M. Yurt, et al. (2021) "Fricative phoneme detection using deep neural networks and its comparison to traditional methods," Proc. Interspeech 2021, 51-55, doi: 10.21437/Interspeech.2021-645