Fundamental Frequency Tracking In Noisy Environments Using Deep Learning

Eric E Hamke ehamke@unm.edu
Amir Nafchi raeisi@ieee.org
Manel Martínez-Ramón manel@unm.edu
Balasubramaniam Santhanam bsanthan@unm.edu
Ramiro Jordan rjordan@unm.edu

2021 55th Asilomar Conference on Signals, Systems, and Computers | 978-1-6654-5828-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/IEEECONF53345.2021.9723185

***Abstract*—This work introduces a novel approach to determine the fundamental frequency or pitch of a person's voice. Using two Deep Long Short-Term Memory Recurrent Neural Networks. The first network is used to perform voiced/unvoiced classifications in a noisy environment. The noisy environments include other speakers in typical settings such as restaurants and large crowds. The second network is used as a time series prediction of the fundamental frequency to enforce the continuity constraint between speech frames. This method will be compared with existing algorithms (Pitch Estimation Filter with Amplitude Compression) and a similar Deep Belief Network trained using the same noisy environment data. We found that the new approach is more robust in -10dB environments than either of the other methods used and had comparable performance in low noise environments.**

***Index Terms*—Speech Processing, Recurrent Neural Networks, Machine Learning, Fundamental Frequency Tracking.**

I. Introduction

Fundamental frequency tracking has been explored as a tool for discerning emotion from speech features in psychology especially related to anxiety and fear. Others have explored using it to examine when a person is experiencing physical stress.

The process of tracking the fundamental frequency is composed of three elements: voice activity/segmentation function; feature extraction (spectral analysis); and a continuity tracking function. The continuity tracking function compensates for errors in the feature extraction processes by imposing a continuity constraint on the estimated frequency values.

The three main novelties of this paper are as follows.

The first one addresses segmenting the speech recording into voiced/unvoiced segments. It is the voiced segments that exhibit resonance that reflect the fundamental frequency or the pitch. Most pitch tracking algorithms compute a probability that the Short Time Fourier Transform (STFT) of a sampled signal is speech or back ground using various robust measures. Hughes and Mierle [1] and Graves, Abdel-rahman and Hinton [2], [3] proposed using deep Bidirectional Recurrent Neural Network's with for phoneme/word recognition. Dissen [4] proposed and implemented a similar approach in tracking formants. Sak et. al. [5] has shown that Long Short-Term Memory (LSTM) [6] RNNs are more effective than Deep Neural Networks and conventional RNNs for acoustic modeling. They concluded that a two-layer deep LSTM RNN can exceed state-of-the-art speech recognition performance. They attributed this to the LSTM being able to not only consider the current feature set but also learn trends across previous speech frames. We use a deep Bidirectional Long Short-Term Memory (BLSTM) RNN to identify voiced and unvoiced segments of recording. Our use of the LSTM is tested with noise sources that contain other speech signals. Ee are using a longer frame interval of 40ms to enhance the autocorrelation between frames. With formant tracking the frames are usually around 20ms in length to promote stationarity.

The second novelty involves using Expectation Maximization to fit a multimodal distribution to a Relative Power Spectrum. The distributions are composed of kernels composed of normal and students-t distributions. With noise present the Short Time Spectral Transform (STFT) has many lesser peaks than the primary resonant peaks. These peaks tend to show up in a peak finding algorithms making it hard to separate them out. The expectation maximization algorithm fits the more prominent peaks.

The fitting process is performed using a STFT of 40ms window. This makes resonant features easier to "see" and work with. The larger window reinforces the correlation between frames and tends to average out the impact of the noise. A set of standardized noise environments are used to reflect the types of environments (airports, train/bus stations, checkout lines, restaurants) where the processing will occur in.

The last novelty involves using a second deep LSTM RNN to impose a continuity constraint on the observed pitch values. This constraint is a result of the natural processes of speech where the transitions between formants is a smooth process as the larynx and articulators' transition between sounds. The observation process does not always track this due to the noise environments, so it is necessary to constrain an observation in a reasonable manner. In this manner, the speech estimates can viewed as a time series and we wish to predict the next value based on the previous observations and the current estimate.

The key advantage to using an RNN for both the voiced/unvoiced and the continuity constraint is that networks can be retrained or updated with each new use, which makes them adaptive, i.e. an application will self-adjust to a user over time. [7].

In the following discussion we, quickly present the details

Department Of Electrical and Computer Engineering, University of New Mexico, Albuquerque, New Mexico, USA

of the methodologies and provide the results of a study comparing the results with existing approaches Pitch Estimation Filter with Amplitude Compression (PEFAC) [8] and Deep Belief Networks. The use of a Deep Belief Network for Voice Activity Detection [9], [10] to make the unvoiced/voicing decision. All these methods are used in an off-line manner as is our method.

II. Methodology

1) Relative Power Spectrums: In this study, each resonant peak was modeled as a composite kernel [11], [12] containing a Normal, a Laplace, and a Students t-Location-Scale distribution. A Students t-Location-Scale distribution is a students-t distribution centered on the data's mean. Each distribution exhibited varying degrees of ability to represent the exponential decay. The Normal dominates the middle region of the kernel mixture model. The Normal distribution contributed to the shoulders of the kernel and helped support bandwidth modeling. Finally, the Students-t distribution's contribution was used to make the tails thicker. This tail effect is important in modeling the noise floor as discussed in the next section.

These kernels were fitted using the Expectation Maximization technique [13]. The parameter set consisted of a common mean μ_n, a variance σ_n for the Gaussian distributions, and the degree of freedom ν_n for the Students-t distribution.

The contribution vector, $r_n^T = [r_{N,n}, r_{T,n}]$, represent the responsibility of each member function to the shape of n^{th} kernel:

$$K_n\left(f \mid V_n\right) = r_n^T D\left(f \mid V_n\right) \tag{1}$$

where $D\left(f \mid V_n\right)$ is a vector representing the values of the compound distributions using parameter set V_n at frequency f.

$$D\left(f \mid V_n\right) = \begin{bmatrix} \mathcal{N}(f|\mu_{N,n}, \sigma_{N,n}) \\ \mathcal{T}\left(f|\mu_{T,n}, \sigma_{T,n}, v\right) \end{bmatrix} \tag{2}$$

A second set of weights ($w_n^T = [w_{N,n}, w_{T,n}]$) are used to represent each kernel's contribution to the overall multimodal distribution.

$$\hat{p}\left(f\right) = \ w^T K\left(f \mid V_n\right) \tag{3}$$

A. Long Short-Term Memory - Recurrent Neural Networks

Long Short-Term Memory - Recurrent Neural Networks (LSTM-RNN) were used to determine which segments are voiced and unvoiced, and to track the frequency with a continuity constraint. Tracking the F_0 began with identifying those segments of the recorded speech that are voiced. It is important to note that unvoiced segments do not have harmonic content. The continuity constraint ensured that the F_0 estimates represent the physical processes of speech. The speech varies continuously over time.

The LSTM node contains special units called memory blocks in the recurrent hidden layer. The first LSTM block uses the initial state of the network and the initial sequence values to compute the first output and the updated cell state. At time step t, the block uses the current state of the network (c_{t-1},h_{t-1}) and the next time step of the sequence to compute the output and the updated cell state c_t.

The layer's state is composed of an output state and a cell state. The output state at any instant in time contains the output of the LSTM layer. The cell state contains information learned from the previous states. With each new iteration, the layer adds information to or removes information from the cell state. The layer controls these updates using gates. [14]

The learnable weights of an LSTM layer are the input weights $\mathbf{W}$, the recurrent weights $\mathbf{R}$, and the bias $\mathbf{b}$. The matrices $\mathbf{W}$, $\mathbf{R}$, and vector $\mathbf{b}$ are concatenations of the input weights, the recurrent weights, and the bias of each component, respectively. These matrices are concatenated as follows:

$$\mathbf{W} = \begin{bmatrix} W_i \\ W_f \\ W_g \\ W_o \end{bmatrix}, \mathbf{R} = \begin{bmatrix} R_i \\ R_f \\ R_g \\ R_o \end{bmatrix}, \mathbf{b} = \begin{bmatrix} b_i \\ b_f \\ b_g \\ b_o \end{bmatrix}, \tag{4}$$

where $\mathbf{i}$, $\mathbf{f}$, $\mathbf{g}$, and $\mathbf{o}$ denote the input gate, forget gate, cell candidate gate, and output gate, respectively.

The cell state at time step t is given by

$$\mathbf{c}_t = f_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t \tag{5}$$

where $\odot$ denotes the Hadamard product (element-wise multiplication of vectors).

The hidden state at time step t is given by

$$\mathbf{h}_t = \mathbf{o}_t \odot \sigma_c\left(\mathbf{c}_t\right) \tag{6}$$

where σ_c denotes the state activation function. The LSTM layer function, by default, uses the hyperbolic tangent function (tanh) to compute the state activation function.

The following formulas describe the components at time step t.

$$\mathbf{i}_t = \sigma_g\left(\mathbf{W}_i \mathbf{x}_t + \mathbf{R}_i \mathbf{h}_{t-1} + \mathbf{b}_i\right), \tag{7}$$

$$\mathbf{f}_t = \sigma_g\left(\mathbf{W}_f \mathbf{x}_t + \mathbf{R}_f \mathbf{h}_{t-1} + \mathbf{b}_f\right), \tag{8}$$

$$\mathbf{g}_t = \sigma_g\left(\mathbf{W}_g \mathbf{x}_t + \mathbf{R}_g \mathbf{h}_{t-1} + \mathbf{b}_g\right), \tag{9}$$

$$\mathbf{o}_t = \sigma_g\left(\mathbf{W}_o \mathbf{x}_t + \mathbf{R}_o \mathbf{h}_{t-1} + \mathbf{b}_o\right) \tag{10}$$

In these calculations, σ_g denotes the gate activation function. The LSTM layer function, by default, uses the sigmoid function given by $\sigma(x) = (1 + e^{-x}) - 1$ to compute the gate activation function [14].

1) The voiced/unvoiced Segments Network Structure: We created a Matlab script that uses the pitch tracking data base's (PTDB) [15], [16] audio recordings (MIC files) and the supplied ground truth (REF files) developed from applying the RAPT algorithm tuned to process PTDB's glottal recordings. We also chose a diverse set of noise sources recovered from the NOIZEUS data set [17] developed at the University of Texas, Dallas using spectral subtraction. The environments include

Babble (crowd of people); Car; Exhibition Hall; Restaurant; Street; Airport; and Train Station.

The network architecture used two bidirectional long short-term memory (BLSTM) networks. The first layer had 256 nodes and the second layer used 128 nodes. Between two layers, we used a dropout layer with a 20% probability of randomly setting an input element to zero. This dropout technique minimized the chance the network will be over trained. The BLSTM layers feed into a fully connected layer. The final two layers consist of a softmax and classification layer. The classification layer outputs the probability that the frame being processed is either a voiced or unvoiced frame.

The input vector consisted of 24 features of the 40ms frame: spectral centroid; spectral crest; spectral entropy; spectral Flux; spectral kurtosis; spectral rolloff point; spectral skewness; spectral slope; and harmonic ratio. A gammatone [18] filter bank with 15 cepstral coefficients was also used.

The network was trained with the Adaptive Moment estimation (ADAM) [19] algorithm using 20 epochs with a minibatch size of 20. The ADAM algorithm computed adaptive learning rates for the parameters using a running average of the first and second moments of the gradient of the stochastic. [19]. The unbiased moments ($\hat{\mathbf{m}}_t$ and $\hat{\mathbf{v}}_t$, respectively) are defined as follows:

$$\hat{\mathbf{m}}_t = \left[\beta_1 \hat{\mathbf{m}}_{t-1} + (1-\beta_1)\,\mathbf{g}_t\right] / \left(1-\beta_1^t\right), \tag{11}$$

$$\hat{\mathbf{v}}_t = \left[\beta_2 \hat{\mathbf{v}}_{t-1} + (1-\beta_2)\,\mathbf{g}_t^2\right] / \left(1-\beta_2^t\right) \tag{12}$$

where $\mathbf{g}_t$ gradients respect to the stochastic objective at time step t; $\mathbf{g}_t^2$ is defined as $\mathbf{g}_t \odot \mathbf{g}_t$; β_1 and β_2 are exponential decay rates for the moment estimates $\beta_1, \beta_2 \in [0, 1)$, typically set to 0.9 and 0.999 respectively. The parameters θ_t are updated using equation

$$\theta_t = \theta_{t-1} - \alpha\ \hat{\mathbf{m}}_t / \left(\sqrt{\hat{\mathbf{v}}_t} + \epsilon\right) \tag{13}$$

where α is set to 0.001.

2) Continuity Constraint Network Structure: The continuity constraints network uses individual speech file segments and it was trained using 2 separate data sets, the first using conventional data values and one using the RES.

The F_0 tracking network architecture used two LSTM networks layers of 512, and 256 nodes respectively, with a dropout layer with a 20% probability of zeroing an input between the LSTM layers. The regression statistics resulted from using a fully connected layer of 256 down to a single node with linear activation. The final regression layer produced an estimated F_0 value based on previous F_0 estimates and spectral data.

The STFT input vector consisted of the spectrum frequency bins from 50 to 300; estimated F_0 using the maximum peak in desired band; gammatone cepstral coefficients for the frequencies [62, 97, 135, 177, 225, 278, 337] Hz; Spectral Centroid; and the harmonic ratio for a frame.

The relative power input features consisted of the first 12 relative power bin values, and the estimated F_0 using the Relative Power Spectrum. The F_0 estimate was formed using the mode of the differences between peaks in the RPS or the relative power F_0 estimate. The Specrtral Centroid was also computed using the relative energy index.

The network was trained with the Stochastic Gradient Descend with Momentum (SGDM). The SGDM update is

$$\theta_{t+1} = \theta_t - \alpha\ \nabla E\left(\theta_t\right) + \gamma\left(\theta_t - \theta_{t-1}\right) \tag{14}$$

where t is the iteration number, $\alpha > 0$ is the learning rate, θ is the parameter vector, $E\left(\theta\right)$ is the loss function and γ determines the contribution of the previous gradient step to the current iteration. This last parameter is actually the momentum parameter of the algorithm. By contrast, at each iteration the stochastic gradient descent algorithm evaluates the gradient and updates the parameters according to this gradient. It can oscillate along the path of steepest descent towards the optimum. Adding a momentum term to the parameter update is one way to reduce this oscillation [13].

B. Deep Belief Network

A Restricted Boltzmann Machine (RBM) is a pairwise Markov Random Field [20] with layers of hidden nodes $\mathbf{h} \in \mathbb{R}^{d_h}$ and visible nodes $\mathbf{v} \in \mathbb{R}^{d_v}$ [21] restricted so that nodes within the layer are not connected. In this manner, a joint probability distribution of the states of each node can be factored, and then the learning task is tractable [22], [23].

The most used configuration of the posterior probability distribution $p(h_i|\mathbf{v})$ or $p(v_j|\mathbf{h})$ of a node given the rest is a Bernoulli distribution, which assumes that the states of the nodes are binary. In the context of this work, the hidden nodes are fed with feature vector of the noise, which has been normalized, so that their components are between 0 and 1. The visible nodes are interpreted as the probability that their state is 1. The relationship between the hidden and the visible layers can be written as.

$$\mathbf{v} = \mathbf{W}\mathbf{h} + \mathbf{b} \tag{15}$$

$$\mathbf{h} = \mathbf{W}^\top \mathbf{v} + \mathbf{c} \tag{16}$$

where the matrix $\mathbf{W} \in \mathbb{R}^{d_n \times d_h}$ is called the generative matrix, and its transpose is the recognition matrix, and where $\mathbf{b}$ and $\mathbf{c}$ are bias terms. Thus, the vector of posterior probabilities can be approximated by $p(\mathbf{v}|\mathbf{h}) = \text{sigm}(\mathbf{W}\mathbf{h} + \mathbf{b})$ and $p(\mathbf{h}|\mathbf{v}) = \text{sigm}(\mathbf{W}^\top \mathbf{h} + \mathbf{c})$ where sigm is a sigmoid function. The training method proposed by Hinton [23], [24] consists of reducing the so-called contrastive divergence between both distributions. Roughly speaking, this can be interpreted as the difference between the cross-correlation matrix of the actual values of the visible and hidden nodes and the cross-correlation matrix of values randomly sampled from their probability distributions. Assuming a set of normalized input patterns $\mathbf{v}_i$, $1 \leq i \leq N$, the training consists of computing values $\mathbf{h}_i$ for each input $\mathbf{v}_i$. Then, a set of random values $\mathbf{v}_i'$ and $\mathbf{h}_i'$ are sampled from distribution $p(\mathbf{h}|\mathbf{v}_i)$ and $p(\mathbf{v}|\mathbf{h}_i)$ and the update at iteration k is computed as

$$\Delta_{W_k} = \mathbb{E}(\mathbf{v}\mathbf{h}^\top|\mathbf{v}_i) - \mathbb{E}(\mathbf{v}\mathbf{h}^\top) \approx \sum_i \mathbf{v}_i {\mathbf{h}_i}^\top - \sum_i \mathbf{v}'_i {\mathbf{h}'_i}^\top \tag{17}$$

$$\mathbf{W}_k = \mathbf{W}_{k-1} + \mu\Delta_{W_k} \tag{18}$$

The operation for $\mathbf{b}$ and $\mathbf{c}$ is analogous. Our implementation includes the use of two stacked RBMs, which can be trained in a sequential way [25].

1) RBM Feature Set: As in the case of the RNN networks, we used the PTDB [15] audio recordings (MIC files). The noise sources recovered from the NOIZEUS data set [17] to represent the background noise sources. The voice signal was processed in 10ms frames as suggested by Van Segbroeck [10].

The feature set consisted of processing the 64 channel Gammatone filter bank emulating human hearing, spectral characterizations of the STFT frames to include centroid, entropy, flux, kurtosis, rolloff point, skewness, and slope as implemented in MATLAB, Long Term Spectral Variablity (LTSV), and an estimate of voiced probability of a denoised signal. The LTSV [26] is a measure of the variance of the entropy measured over all frequency bins of the normalized short-time Gammatone filtered time-frequency representation of the speech spectrum. It is intended to identify variations resulting from phoneme production.The final stage is then passed through a sigmoid function to enforce the probability constraint (all values lying in the interval from 0 to 1) [10]. The estimation of the voiced probability was developed using an algorithm developed by Van Segbroeck [10] that sequentially removes voicing information from the signal; uses the devoiced signal to estimate the noise; and then subtracts the noise from the signal. The denoised signal is then used to estimate the probability of voice being present in the frame.

III. Experimentation

Our experimentation is recorded in Table I shows that the algorithms perform consistently in a low noise environment. In high noise environments, the PEFAC and the STFT-RNN were the only algorithms capable of detecting voiced segments in a high noise environment.

TABLE I
AVERAGE OPTIMAL POINT

	SNR $100dB$			SNR $-10dB$		
Method	TPR	FPR	EER	TPR	FPR	EER
PEFAC	0.85	0.14	0.25	0.72	0.23	0.30
STFT-RNN	0.89	0.12	0.23	0.72	0.25	0.30
Deep Belief Network	0.87	0.13	0.25	0.60	0.39	0.41
Gated Recurrent Unit	0.90	0.09	0.22	0.83	0.15	0.25

It should be noted that for an SNR of 100dB show significant differences between the receiver operation curves (ROC) and DET curve values. The values are more consistent between the two methods.

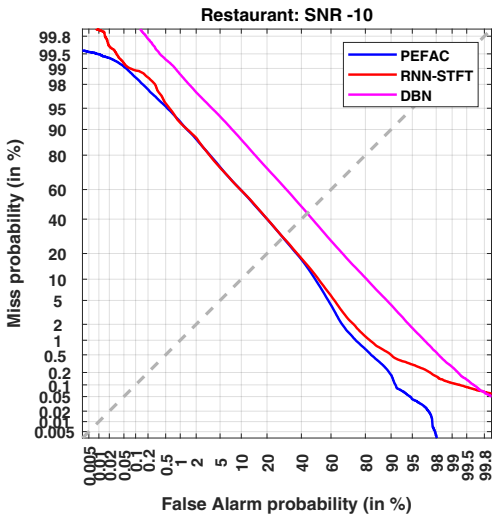

Fig. 1. Comparison of Voiced-Unvoiced DET Curves - The DET curve plots are for the Restaurant noise source at an SNR of -10dB. The blue line represents the performance of the PEFAC algorithm; the red line represents the performance of the STFT-RNN classifier, the magenta line is the Deep Belief Network. The gray dashed line is the equal error rate line.

A. Fundamental Frequency Tracking

The next stage of the study is to combine the use of the voiced/unvoiced decisions models for voiced/unvoiced decisions and the F_0 tracking RNN model.

The F_0 tracking data treats speech recording as separate time-series samples. Again, the leading and trailing silences were removed from each sample and then each one was corrupted with the noise environment and SNR value. The resulting sets were sorted based on the length of each sample. This was done to improve the training performance.

We used the following evaluation metrics [27] to quantify the performance of two STFT and RPS RNN models in 6 noise environments at two SNR values (100dB, and -10dB).

- Gross Pitch Error (GPE): the proportion of frames – considered voiced by both pitch tracker and ground truth – where the relative pitch error is higher than 20%.
- Fine Pitch Error (FPE): the standard deviation of the distribution of relative error values (in Hertz) from the frames that do not have gross pitch errors.
- Voicing Decision Error (VDE): the proportion of frames for which an incorrect voiced/unvoiced decision is made.
- F0 Frame Error (FFE): the proportion of frames for which an error (either according to the GPE or the VDE criterion) is made. FFE can be considered a single measure of overall performance.

Examination of Table II shows that both the STFT and RPS based RNN models provide equivalent performance in the low noise environment. This validates that the RNN algorithms perform consistently with the accepted PEFAC and the RAPT generated ground truth.

The RNN approaches shows significant improvements in the percent gross F_0 error metric. The RES-RNN approach also shows a significant decrease in the standard deviation or fine F_0 errors. (see Table III).

TABLE II
AVERAGE FREQUENCY TRACKING MEASUREMENTS, SNR 100DB

Environment	DET STFT RNN	ROC STFT RNN	RPS RNN	PEFAC
GP E(percent)	3.6	3.7	3.8	3.7
FPE (Hz)	5.93	6.04	5.79	4.54
VDE (percent)	15.6	15.9	16.4	13.2
FFE (percent)	19.2	19.6	20.2	16.9
VDE/FFE (percent)	81.3	81.1	81.2	78.1

TABLE III
AVERAGE FREQUENCY TRACKING MEASUREMENTS, SNR -10DB)

Environment	DET STFT RNN	ROC STFT RNN	RPS RNN	PEFAC
GPE (percent)	7.0	6.8	5.7	26.1
FPE (Hz)	9.17	9.35	7.13	11.85
VDE (percent)	30.1	40.4	39.8	38.5
FFE (percent)	37.0	47.7	45.5	64.6
VDE/FFE (percent)	81.4	84.7	87.5	59.6

The RNN and PEFAC make roughly the same percentage of voice decision errors confirm the observations made when analyzing the voiced/unvoiced decision performance (Table III). When comparing the ratio of the Voicing Decision Error to the total Frame Error, the RNN method ratios show that the overall contribution of the Voicing Decision Error is 84% (STFT) and 87% (RES). The ratio for PEFAC is 59% indicating the tracking contribution is significantly higher. These ratios clearly show the RNN methodologies perform better at tracking the fundamental frequency and highlight that the VDE is the more important factor in tracking the Fundamental Frequency.

Our future work will expand to examine the use of Gated Recurrent Units. Some of our initial studies that need verification indicate that we may be ale to get comparable results to the LSTMs studied here. We are also planning to look at applying a similar study methodology to speech synthesis approaches.

REFERENCES

[1] T. Hughes and K. Mierle, "Recurrent neural networks for voice activity detection," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7378–7382.

[2] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 6645–6649.

[3] A. Graves, N. Jaitly, and A.-r. Mohamed, "Hybrid speech recognition with deep bidirectional lstm," in *2013 IEEE workshop on automatic speech recognition and understanding*. IEEE, 2013, pp. 273–278.

[4] Y. Dissen and J. Keshet, "Formant estimation and tracking using deep learning." in *INTERSPEECH*, 2016, pp. 958–962.

[5] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Fifteenth annual conference of the international speech communication association*, 2014.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] A. De Cheveigné and H. Kawahara, "Yin, a fundamental frequency estimator for speech and music," *The Journal of the Acoustical Society of America*, vol. 111, no. 4, pp. 1917–1930, 2002.

[8] S. Gonzalez and M. Brookes, "Pefac-a pitch estimation algorithm robust to high levels of noise," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 2, pp. 518–530, 2014.

[9] X.-L. Zhang and J. Wu, "Deep belief networks based voice activity detection," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 4, pp. 697–710, 2012.

[10] M. Van Segbroeck, A. Tsiartas, and S. Narayanan, "A robust frontend for vad: exploiting contextual, discriminative and spectral cues of human voice." in *INTERSPEECH*, 2013, pp. 704–708.

[11] G. Camps-Valls, J. L. Rojo-Álvarez, M. Martínez-Ramón *et al.*, *Kernel methods in bioengineering, signal and image processing*. Idea Group Pub., 2007.

[12] J. L. Rojo-Álvarez, M. Martínez-Ramón, J. M. Marí, and G. Camps-Valls, *Digital signal processing with Kernel methods*. Wiley Online Library, 2018.

[13] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[14] "Voice activity detection in noise using deep learning," https://www.mathworks.com/help/deeplearning/ug/long-short-term-memory-networks.html, accessed: 2020-01-01.

[15] G. Pirker, M. Wohlmayr, S. Petrik, and F. Pernkopf, "The pitch-tracking database from graz university of technology."

[16] ——, "A pitch tracking corpus with evaluation on multipitch tracking scenario," in *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[17] Y. Hu, "Subjective evaluation and comparison of speech enhancement algorithms," *Speech Commun.*, vol. 49, pp. 588–601, 2007.

[18] R. D. Patterson, I. Nimmo-Smith, J. Holdsworth, and P. Rice, "An efficient auditory filterbank based on the gammatone function," in *a meeting of the IOC Speech Group on Auditory Modelling at RSRE*, vol. 2, no. 7, 1987.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[20] K. A. Murphy, *Machine Learning. A probabilistic perspective*. Cambridge, MA: The MIT Press, 2012.

[21] H. Ackley, E. Hinton, and J. Sejnowski, "A learning algorithm for boltzmann machines," *Cognitive Science*, pp. 147–169, 1985.

[22] P. Smolensky, "Information processing in dynamical systems: Foundations of harmony theory," Colorado Univ at Boulder Dept of Computer Science, Tech. Rep., 1986.

[23] G. E. Hinton, "Training products of experts by minimizing contrastive divergence." *Neural Computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[24] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[25] R. Salakhutdinov and G. Hinton, "Deep boltzmann machines," in *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, D. van Dyk and M. Welling, Eds., vol. 5. Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA: PMLR, 16–18 Apr 2009, pp. 448–455.

[26] P. K. Ghosh, A. Tsiartas, and S. Narayanan, "Robust voice activity detection using long-term signal variability," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 3, pp. 600–613, 2010.

[27] S. Strömbergsson, "Today's most frequently used f0 estimation methods, and their accuracy in estimating male and female pitch in clean speech." in *INTERSPEECH*, 2016, pp. 525–529.