Voice Activity Detection Using an Adaptive Context Attention Model

Juntae Kim and Minsoo Hahn

***Abstract*—Voice activity detection (VAD) classifies incoming signal segments into speech or background noise; its performance is crucial in various speech-related applications. Although speech-signal context is a relevant VAD asset, its usefulness varies in unpredictable noise environments. Therefore, its usage should be adaptively adjustable to the noise type. This letter improves the use of context information by using an adaptive context attention model (ACAM) with a novel training strategy for effective attention, which weights the most crucial parts of the context for proper classification. Experiments in real-world scenarios demonstrate that the proposed ACAM-based VAD outperforms the other baseline VAD methods.**

***Index Terms*—Deep neural network (DNN), voice activity detection (VAD).**

I. Introduction

THE Voice activity detection (VAD) discriminates speech from background-noise signal segments and is used for preprocessing in most speech-related applications, including speech recognition [1]. crucial aspects of VAD are its robustness to unexpected noise types and reasonable computational cost. Various methods, traditionally based on feature engineering [2]–[9], statistical signal processing [10]–[17] and, more recently, on deep neural networks (DNNs) [18]–[28] (which outperform the former methods), have been proposed.

Three research branches of DNN-based VADs can be identified as follows.

1) The first combines several acoustic features as DNN inputs and determines the useful ones [19]–[21].
2) The second models the data-driven acoustic features directly from raw speech (or a spectrogram) and uses convolutional neural network based methods to classify the speech/nonspeech parts [22]–[24].
3) The last focuses on the effective utilization—by the DNN—of context information (CI) along the duration of the speech signals [25]–[28].

In [26] and [27], a long short-term memory recurrent neural network (LSTM-RNN) is adopted for VAD because it can internally encode the long short-term CI from the input features for classification. However, an LSTM-RNN based VAD could not outperform the DNN-based VAD [20]; in addition, an LSTM-RNN is much slower than DNN because the classification is sequential. In [28], a boosted DNN (bDNN) based VAD was proposed; it exploits the input/output CI by adopting multiple input/output units for the DNN. However, the bDNN is based on a fully connected neural network (FNN) with a fixed-size input/output; hence, it can only use a restricted portion of CI (e.g., if the bDNN input/output size decreases, bDNN will depend on only short-term CI). In addition, to aggregate the long short-term CI, they proposed an ensemble model—multiresolution stacking (MRS)—that includes bDNNs of various sizes. It exhibits outstanding performance, but the choice of the bDNNs for MRS follows a rule of thumb and the reported computational cost is approximately 11 times higher than that of a single-bDNN-based VAD.

Within the scope of the last branch, we propose an adaptive context attention model (ACAM) based VAD for noise robustness, which effectively exploits the CI, based on an attention strategy. The main contributions of this letter are as follows.

1) To the best of our knowledge, this is the first attempt to adopt an attention strategy with DNN for VAD.
2) A novel training strategy for the effective context attention process is proposed, inspired by reinforcement learning.
3) The proposed method is benchmarked against several state-of-the-art DNN-based methods, under various noise-independent scenarios, including real-world datasets (the training and test noise corpora are separated).
4) The operation of the attention strategy is investigated across various types of noise and SNRs.

II. Proposed VAD

The ACAM-based VAD is a frame-based speech or noise classifier. The input speech signal is divided into overlapping 25-ms frames with 10-ms shifts. Acoustic feature vectors are extracted from each frame. Our dataset is $\{(\mathbf{x}_m,\ y_m^{\text{truth}})\}_{m=1}^{M}$, where $m = 1, \ldots,\ M$ is the frame index and $\mathbf{x}_m \in R^D$ is the acoustic feature vector for frame m, labeled as $y_m^{\text{truth}} \in \{0,\ 1\}$. To use the input/output CI, we expand our dataset to $\{(\mathbf{v}_m, \mathbf{y}_m^{\text{truth}})\}_{m=1}^{M}$ by selecting the neighboring frames of index $\{-W,\ -W+u,\ -W+2u,\ \ldots,\ -1-u,\ -1,\ 0,\ 1,\ 1+u,\ \ldots,\ W-2u,\ W-u,\ W\}$, and flattening the selected

Manuscript received December 1, 2017; revised February 10, 2018; accepted February 27, 2018. Date of publication March 8, 2018; date of current version June 29, 2018. This work was supported by the Ministry of Trade, Industry and Energy (MOTIE, South Korea) under Industrial Strategic Technology Development Program (10076757, free-running embedded speech recognition technology for natural language dialogue with robots). The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Hakan Erdogan. *(Corresponding author: Minsoo Hahn.)*

The authors are with the School of Electrical Engineering, Korea Advanced Institute of Science and Technology, Daejeon 5-701, South Korea (e-mail: jtkim@kaist.ac.kr; mshahn2@kaist.ac.kr).

Digital Object Identifier 10.1109/LSP.2018.2811740

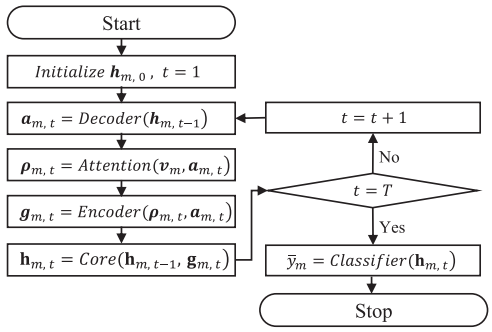

Fig. 1. ACAM block diagram of the *m*th frame classification.

frames as follows:

$$\mathbf{v}_m = [\mathbf{x}_{m-W}^T, \mathbf{x}_{m-W+u}^T, \ldots, \mathbf{x}_{m-1-u}^T, \mathbf{x}_{m-1}^T, \mathbf{x}_m^T, \mathbf{x}_{m+1}^T, \mathbf{x}_{m+1+u}^T, \ldots, \mathbf{x}_{m+W-u}^T, \mathbf{x}_{m+W}^T]^T \quad (1)$$

$$\mathbf{y}_m^{\text{truth}} = [y_{m-W}^{\text{truth}}, y_{m-W+u}^{\text{truth}}, \ldots, y_{m-1-u}^{\text{truth}}, y_{m-1}^{\text{truth}}, y_m^{\text{truth}}, y_{m+1}^{\text{truth}}, y_{m+1+u}^{\text{truth}}, \ldots, y_{m+W-u}^{\text{truth}}, y_{m+W}^{\text{truth}}]^T \quad (2)$$

where *W* and *u* are user-defined parameters described in [28].

A. Model Architecture

The ACAM's backbone is a recurrent attention model used in image and speech recognition [29]–[31], and can be broadly described as taking several input frames and repeatedly deciding which, among them, is most crucial for classification. Fig. 1 describes the model, which includes a decoder, attention, encoder, core network, and classifier, to be described in the following.

1) Decoder: At each sequential step *t* of the core network, the previous internal state $\mathbf{h}_{m,t-1}$ is forwarded to the decoder, as described in Fig. 1. The decoder then decides which of the frames in $\mathbf{v}_m$ should be focused upon by the ACAM to generate the attention vector $\mathbf{a}_{m,t}$

$$\begin{aligned}\mathbf{a'}_{m,t} &= \text{Relu}\left(\text{Affine}\left(\mathbf{h}_{m,t-1}|\theta_{\text{attention}}\right)\right) \\ &= [a'_{m,t,1}, \ldots, a'_{m,t,k}, \ldots, a'_{m,t,L}]^T \end{aligned} \quad (3)$$

$$a_{m,t,k} = \sigma(a'_{m,t,k}) \Big/ \sum_{k=1}^{L} \sigma(a'_{m,t,k}) \quad (4)$$

$$\mathbf{a}_{m,t} = [a_{m,t,1}, \ldots, a_{m,t,k}, \ldots, a_{m,t,L}]^T \quad (5)$$

where *k* is the sample index, $L = 2\,(\mathrm{W} - 1)/u + 3$ [derived from (1)] is the vector length, $\text{Affine}(\mathbf{x}|\theta) = \mathbf{W} \cdot \mathbf{x} + \mathbf{b}$ is the affine transformation with $\mathbf{W}, \mathbf{b} \in \theta$, $\text{Relu}(\mathbf{x})$ is the rectified linear unit (RELU) [32], and $\sigma(x) = 1/(1 + \exp(-x))$ is the sigmoid function. We used a smoothed softmax function in (4) by replacing the exponential function of the ordinary softmax function—which sharpened the attention excessively, degrading the CI deployment and negatively affecting VAD performance—by the sigmoid function.

2) Attention: Using $\mathbf{a}_{m,t}$, the attended input $\boldsymbol{\rho}_{m,t}$ is obtained as a Hadamard product with $\mathbf{v}_m$ as follows:

$$\boldsymbol{\rho}_{m,t} = [\mathbf{x}_{m-W}^T \cdot a_{m,t,1}, \ldots, \mathbf{x}_{m+W}^T \cdot a_{m,t,L}]^T. \quad (6)$$

3) Encoder: The encoder then extracts the aggregated feature $\mathbf{g}_{m,t}$ from $\mathbf{a}_{m,t}$ and $\boldsymbol{\rho}_{m,t}$ as follows:

$$\mathbf{g}_{m,t} = f_g(f_a(\mathbf{a}_{m,t}|\theta_a), f_\rho(\boldsymbol{\rho}_{m,t}|\theta_\rho)|\theta_g) \quad (7)$$

where $f(x|\theta)$ is the FNN with parameter set θ. Both $f_a(\cdot)$ and $f_\rho(\cdot)$ map their inputs onto the respective hidden spaces, followed by $f_g(\cdot)$, to aggregate the hidden spaces information.

4) Core Network: This network proposes the next action to the succeeding module (e.g., the decoder), based on the current internal state $\mathbf{h}_{m,t}$. The internal state is encoded by summarizing the noise environment and the input information in $\mathbf{g}_{m,t}$ with $\mathbf{h}_{m,t-1}$, which contains the previous action information of the preceding modules. We adopt an LSTM-RNN for the core network as follows:

$$\mathbf{h}_{m,t} = \text{LSTM}(\mathbf{g}_{m,t}, \mathbf{h}_{m,t-1}|\theta_{\text{LSTM}}). \quad (8)$$

5) Classifier: The decoder acts on the frames that should be attended, based on $\mathbf{h}_{m,t}|_{t=1}^{T-1}$. In the last *T* steps (i.e., based on $\mathbf{h}_{m,T}$), the last action is carried out by the classifier as follows:

$$\mathbf{y'}_m = f_c(\mathbf{h}_{m,T}|\theta_c) \quad (9)$$

$$\mathbf{y'}_m = [y_{m-W,1}, y_{m-W+u,2}, \ldots, y_{m,k}, \ldots, y_{m+W-u,L-1}, y_{m+W,L}]^T \quad (10)$$

where $y_{m,k} \in [0,\ 1]$ is the soft prediction for y_m^{truth}. The *m*th frame label can then be predicted by aggregating all the soft predictions relative to frame *m* across *k* (obtaining $\hat{y}_m$), and the hard decision label $\bar{y}_m$ is obtained by thresholding with η

$$\hat{y}_m = \frac{1}{L}\sum_{k-1}^{L} y_{m,k}, \quad \bar{y}_m = \begin{cases} 1, & \text{if} \quad \hat{y}_m \geq \eta \\ 0, & \text{otherwise} \end{cases}. \quad (11)$$

B. Training

Training the ACAM-based VAD can be considered as a common supervised learning problem with the loss function $J_{sv}(\mathbf{y}_m^{\text{truth}}, \mathbf{y'}_m)$ set to the mean-squared error loss, as described in [28]. Furthermore, we propose an additional loss function considering the attention procedure in the training phase. Note that the attention was smoothed by adopting (4) for diverse CI usage, i.e., most of the input frames affect the model output because the attention was smoothed. However, as the noise level increases, most of the input frames are likely to contain noisy information. Therefore, sharpening the attention on the most crucial frame (i.e., the most attended frame) used for correct classification can be beneficial. To emphasize this frame, our proposed additional loss function is

$$k' = \arg\max_k (a_{m,t,k}) \quad (12)$$

$$J_{rf} = -\sum_{m=1}^{M}\sum_{t=1}^{T} \log\left(a_{m,t,k'}\right) \cdot (R_m - b_m) \quad (13)$$

$$R_m = \begin{cases} 1, & \bar{y}_m = y_m^{\text{truth}} \\ 0, & \text{otherwise} \end{cases}. \quad (14)$$

If we assume a multinomial distribution parameterized with $\mathbf{a}_{m,t}$, (13) is equivalent to the loss function of the REINFORCE rule [33], including the negative log likelihood with rewards, $R_m \cdot b_m = E[R_m] \approx \sigma(\text{Affine}(\mathbf{h}_{m,T}))$ is used to

stabilize the gradient estimates [33] and is regressed by minimizing $\sum_{m=1}^{M}\{R_m - \sigma(\text{Affine}(\mathbf{h}_{m,T}))\}^2$, R_m being treated as a constant. Note that, if $R_m = 1$, (13) becomes a monotonically decreasing function over $a_{m,t,k'}$; minimizing J_{rf} increases $a_{m,t,k'}$ and decreases $a_{m,t,k\neq k'}$ because its sum over *k* is unity. If $R_m = 0$, the opposite occurs.

Our final, hybrid, loss function is set to $J = J_{sv} + J_{rf}$. Notably, the attention trained with this loss function can also be considered as a softened version of max-pooling, an effective operation that ignores noisy information in noisy image-recognition tasks [34]. However, in this task, applying naïve max-pooling—instead of (6) trained with *J*—did not result in improved performance. Additionally, the ACAM performance when trained with only J_{sv} (thus discarding the effect of J_{rf}) was not better than with the full *J*. This implies that there is a tradeoff between CI deployment and focusing on a frame less distorted by noise. We believe that learning with *J* can enable the model to suitably adjust the tradeoff, i.e., to focus on a frame less distorted by noise or with crucial CI, while accessorily using the CI of other frames.

III. Experiments and Results

A. Experimental Setup

1) Dataset Preparation: For training, we used the training dataset in the TIMIT corpus [35] as the speech dataset. Given that TIMIT utterances have considerably shorter silences than speech, class imbalance problems may occur. To address these problems, 2-s-long silence segments were added before and after each utterance. We used the ground truth label in TIMIT. For the noise dataset, a sound effect library [36] containing approximately 20 000 sound effects was used. Initially, we randomly selected 5000 sound effects and concatenated them into a long sound wave. We then randomly selected an utterance from the silence-added TIMIT train dataset and added it to the long sound wave with a randomly selected SNR between -10 and 12 dB; this was repeated until the end of the long sound wave. We used FaNT [37] for adding the noise to the utterance; 95% of the training dataset was used to train the model, and the remaining 5% in model validation.

As test datasets, we used the TIMIT test dataset with NOISEX-92 [38] (D1), a directly recorded dataset (D2), and the HAVIC corpus [39] (D3).

1) For D1, we concatenated all the 1344 silence-added utterances (approximately 2.4 h) in the TIMIT test dataset, and added nine types of noise from the NOISEX-92 database with various SNRs (-5, 0, 5, and 10 dB). In addition, we used the clean utterances to investigate the SNR mismatch conditions.
2) For D2, a real, noisy speech dataset was recorded at four locations (bus stop, construction site, park, and room), for approximately 2 h. Because of space limitations, details concerning D2 may be found in [40].
3) For D3, we used all 72 h of web videos with the ground truth labels in the HAVIC corpus.

Multiresolution cochleagram (MRCG) features were adopted for $\mathbf{x}_m$. The extracted 768-dimensional MRCG features were z-score-normalized for training. To normalize the MRCG features for testing, the same normalization factors used in training were applied.

TABLE I
Number of Parameters for Each Model

	HFCL	MSFI	DNN	bDNN	LSTM	ACAM
# param.	▪	~2 k	~3015 k	~3018 k	~2097 k	~953 k

TABLE II
Usage Ratio (%) Between the Long and Short CI for ACAM-2 and 3, According to the Noise Type and SNR

Model	SNR	Babble	F16	Gun	Military	Volvo	White
ACAM-2	-5 dB	40.4	43.3	42.7	42.6	46.6	41.8
	10 dB	41.4	43.8	42.5	43.6	44.9	41.4
ACAM-3	-5 dB	44.2	43.9	47.7	42.5	51.1	44.4
	10 dB	41.0	39.3	47.8	37.9	50.7	40.5

2) Parameter Setting: The model parameters *W*, *u*, and *T* were set to 19, 9, and 7, respectively. The decision threshold η was set to 0.5. Each FNN in the encoder, namely $f_a(\cdot|\theta_a)$, $f_\rho(\cdot|\theta_\rho)$, and $f_g(\cdot|\theta_g)$ had a hidden layer with 128 units. The FNN in the classifier $f_c(\cdot|\theta_c)$ had two hidden layers with 256 units in each layer. The output layer activation function of the classification network was a sigmoid; RELUs were used for the hidden layers of all the FNNs. The core network had 128 LSTM units. All the aforementioned parameters are from our validation dataset; the initial weights were all drawn from a normal Gaussian distribution with a variance of 10^{-8}.

3) Baseline Methods: To investigate the effect of the proposed network architecture (irrespective of the effect of the input features), three approaches were used with the MRCG features: DNN, bDNN, and LSTM-based VADs. Additionally, two state-of-the-art methods, the harmonic frequency components in a likelihood ratio test (HFCL) [15] and the merging source and filter-based information (MSFI) [21] based VADs were compared.

For the DNN and the bDNN, we followed the model architecture and parameters described in [28]. Note that bDNN is a state-of-the-art method based on MRCG features. For the LSTM, we followed the specification proposed in [23]; all parameters were the same as in [23], except for the number of hidden units, which was set to 256, as per our validation dataset. The parameters of the HFCL- and MSFI-based VADs were set to the optimal ones specified in [15] and [21]. The approximate number of parameters for each model is listed in Table I.

4) Training Methods: All the MRCG-based models, ACAM, DNN, bDNN, and LSTM, were trained using Adam [41]. The optimal initial learning rates for each model were obtained by a random search method [42], using our validation dataset. The batch sizes of the ACAM, DNN, and bDNN were set to 4096. For training, the LSTM was unrolled for 20 time steps and had five target delays, as described in [23]. For regularization, the batch and layer normalizations in [43] and [44] were applied to the fully connected and LSTM layers, respectively. A dropout [45] with rate of 0.5 was used. The early stopping method in [46] was adopted for deciding the number of weight updates. Note that these regularization methods were fairly applied to each MRCG-based model.

5) Evaluation Metric: As an evaluation metric, we used the area under the curve (AUC), a quantitative measure of the receiver operating characteristic curve [28].

TABLE III
D1, D2, AND D3 AUC (%) COMPARISON

Dataset	Noise	HFCL	MSFI	DNN	bDNN	LSTM	ACAM-1	ACAM-2	ACAM-3
D1	Clean	94.23	99.39	99.20	**99.84**	98.55	99.79	99.80	99.82
	Babble	68.07	86.48	89.17	**91.75**	79.48	90.04	91.07	91.13
	Destroyer	72.75	90.94	95.97	93.97	84.80	94.10	**95.50**	95.48
	F16	72.91	91.06	93.85	93.68	88.45	94.88	**95.92**	95.58
	Factory 1	67.99	86.47	88.64	90.63	84.43	92.26	92.24	**92.32**
	Factory 2	75.75	94.71	96.77	97.20	97.67	97.83	**98.12**	98.03
	Gun	71.20	98.33	97.28	98.57	97.58	98.74	98.94	**99.13**
	Military	76.20	93.55	96.03	96.99	93.65	97.72	98.01	**98.12**
	Volvo	90.65	99.60	99.50	**99.70**	99.15	99.67	99.53	99.55
	White	74.11	97.04	98.05	98.62	92.10	98.09	**98.65**	98.34
D2	Bus stop	79.43	97.04	92.91	98.80	96.42	98.90	**98.93**	98.92
	Cons. site	84.16	94.98	89.61	99.39	97.89	**99.51**	99.46	99.43
	Park	83.34	98.00	90.04	99.01	98.10	99.05	**99.09**	99.05
	Room	82.17	97.58	90.83	99.22	96.55	**99.37**	99.33	99.34
D3	▪	66.07	69.21	71.41	69.64	71.00	72.15	72.29	**73.42**

For D1, AUC is averaged over the SNRs (dB) $\{-5, 0, 5, 10\}$, except for clean speech. The numbers in bold indicate the best results.

6) Implementation Environment: All the experiments were conducted on an Intel(R) Core(TM) i7-6700k workstation with an NVidia GTX 1080.

B. Results and Discussion

To investigate the effects of the proposed loss function and the attention, three types of ACAM were used as follows:

1) ACAM-1 was trained with a fixed attention, i.e., the values in (5) were set to nontrainable identical constants;
2) ACAM-2 was trained with J_{sv}; and
3) ACAM-3 was trained with J.

The behavior of each model with respect to CI is described by the ratio short / (short + long) $\times$ 100 = short $\times$ 100 shown in Table II. The "short" and "long" CI types are related to $\{a_{m,T,3},\ a_{m,T,4},\ a_{m,T,5}\}$ and $\{a_{m,T,1},\ a_{m,T,2},\ a_{m,T,6},\ a_{m,T,7}\}$, respectively, which are summed across k and averaged over m. Because all the $a_{m,\ T,\ k}$ values in ACAM-1 were set to be identical, the ratio for ACAM-1 was fixed at 42.9. Table II shows that both ACAM-2 and ACAM-3 use CI adaptively, according to the noise types. However, as expected, ACAM-3 favored one of the CIs more than ACAM-2, because of the effects of (13). In fact, the absolute difference between the ratios of ACAM-3 and ACAM-1 (which is fixed at 42.9) was greater than between ACAM-2 and ACAM-1, except in the case of babble noise at –5-dB SNR. In addition, ACAM-3 adaptively attended to the SNRs (it aimed for the short CI as the SNR decreased, except in the case of machine gun noise); ACAM-2 exhibited a relatively constant CI usage.

Table III compares D1, D2, and D3. For D1, the ACAM-based VADs outperformed all the baseline methods, except in the clean, babble, and volvo noise cases, in which the bDNN-based VAD performed marginally better. Possible reasons are: First, the bDNN-based VAD can fit the TIMIT speech signal better than the ACAM-based VADs because it exhibits the highest performance in clean speech. This relates to the babble-noise performance because the performance of MRCG-based methods in babble-type noise is proportional to clean-speech performance. Second, the ACAM-2 and ACAM-3 strategies can work incorrectly with volvo noise. Table II shows that, for volvo noise, they both aim for the short CI and have lower performances than the bDNN- and ACAM-1-based VADs (with a 42.9 ratio), implying that the long CI is more important in this type of noise. Among the ACAM-based VADs, ACAM-2- and ACAM-3-based VADs demonstrated similar average performances, and the ACAM-1 exhibited the worst performance.

For D2, all the ACAM-based VADs outperformed the baseline methods and exhibited similar performances. For D3, which was recorded in real-world conditions and included noise types unseen in the training dataset (e.g., music sounds) with the longest duration among the test datasets, the main results were: first, the ACAM-3-based VAD outperformed all the other methods, and second, the ACAM-1-based VAD had the worst performance among the ACAM-based VADs. These results imply that the adaptability of the attention strategy to noise type and SNR (as in ACAM-3) is necessary to achieve high generalization capabilities in VAD.

Although ACAM-based VADs show outstanding performances, they consist of several modules with a sequential process, which can cause excessive computational load. Therefore, we compared the computation time (on our GPU workstation) of the ACAM-based VAD with those of the other MRCG-based methods. The average processing time per second of speech of the DNN-, bDNN-, LSTM-, and ACAM-based VADs were 0.88, 9.24, 31.61, and 10.12 ms, respectively; this does not include the MRCG extraction time of 206.50 ms, which is relatively high compared to the 9.04 ms of HFCL- and the 67.73 ms of the MSFI-based VAD; the consideration of the MRCG extraction time is beyond the scope of this letter. Note that ACAM-based VADs have reasonable computation costs, compared to other MRCG-based VADs, because: first, although the core network of the ACAM has a sequential process and there are only seven sequential steps; and 2) the encoder can effectively compress the input features to a hidden feature space with lower dimensionality than in the other methods, by collaborating with the attention module; this is also the reason for the generalized performance of ACAM-based VADs, by a reduction-of-dimensionality effect [32].

IV. CONCLUSION

In this letter, we investigated the use of a neural network architecture in VAD (ACAM), and a training strategy for adaptively deploying the CI according to the noise type and level. Our experiments demonstrated that the ACAM-based VAD outperforms several state-of-the-art VADs in various real environments. The used MRCG-based VADs and D2 are available in [40], for reproducibility purposes.

REFERENCES

[1] D. Vlaj, B. Kotnik, B. Horvat, and Z. Kačič, "A computationally efficient Mel-filter bank VAD algorithm for distributed speech recognition systems," *EURASIP J. Adv. Signal Process.*, vol. 2005, no. 4, Dec. 2005, Art. no. 561951.

[2] A. Benyassine, E. Shlomot, H. Y. Su, D. Massaloux, C. Lamblin, and J. P. Petit, "ITU-T recommendation G.729 annex B: A silence compression scheme for use with G.729 optimized for V.70 digital simultaneous voice and data applications," *IEEE Commun. Mag.*, vol. 35, no. 9, pp. 64–73, Sep. 1997.

[3] N. Dhananjaya and B. Yegnanarayana, "Voiced/nonvoiced detection based on robustness of voiced epochs," *IEEE Signal Process. Lett.*, vol. 17, no. 3, pp. 273–276, Mar. 2010.

[4] H. Ghaemmaghami, B. J. Baker, R. J. Vogt, and S. Sridharan, "Noise robust voice activity detection using features extracted from the time-domain autocorrelation function," in *Proc. Interspeech*, Makuhari, Japan, 2010, pp. 3118–3121.

[5] K. Ishizuka, T. Nakatani, M. Fujimoto, and N. Miyazaki, "Noise robust voice activity detection based on periodic to aperiodic component ratio," *Speech Commun.*, vol. 52, no. 1, pp. 41–60, Jan. 2010.

[6] P. C. Khoa, "Noise robust voice activity detection," Ph.D. dissertation, School of Computer Engineering, Nanyang Technological Univ., Singapore, 2012.

[7] I.-C. Yoo and D. Yook, "Robust voice activity detection using the spectral peaks of vowel sounds," *ETRI J.*, vol. 31, pp. 451–453, Aug. 2009.

[8] I. C. Yoo, H. Lim, and D. Yook, "Formant-based robust voice activity detection," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 23, no. 12, pp. 2238–2245, Dec. 2015.

[9] S. O. Sadjadi and J. H. Hansen, "Unsupervised speech activity detection using voicing measures and perceptual spectral flux," *IEEE Signal Process. Lett.*, vol. 20, no. 3, pp. 197–200, Mar. 2013.

[10] S. Jongseo, K. N. Soo, and S. Wonyong, "A statistical model-based voice activity detection," *IEEE Signal Process. Lett.*, vol. 6, no. 1, pp. 1–3, Jan. 1999.

[11] D. Ying, Y. Yan, J. Dang, and F. K. Soong, "Voice activity detection based on an unsupervised learning framework," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 19, no. 8, pp. 2624–2633, Nov. 2011.

[12] P. Renevey and A. Drygajlo, "Entropy based voice activity detection in very noisy conditions," *Proc. Eurospeech*, pp. 1887–1890, Sep. 2001.

[13] G. Aneeja and B. Yegnanarayana, "Single frequency filtering approach for discriminating speech and nonspeech," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 23, no. 4, pp. 705–717, Apr. 2015.

[14] P. K. Ghosh, A. Tsiartas, and S. Narayanan, "Robust voice activity detection using long-term signal variability," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 19, no. 3, pp. 600–613, Mar. 2011.

[15] L. N. Tan, B. J. Borgstrom, and A. Alwan, "Voice activity detection using harmonic frequency components in likelihood ratio test," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Mar. 2010, pp. 4466–4469.

[16] R. J. Weiss and T. T. Kristjansson, "DySANA: Dynamic speech and noise adaptation for voice activity detection," in *Proc. Interspeech*, 2008, pp. 127–130.

[17] M. Fujimoto and K. Ishizuka, "Noise robust voice activity detection based on switching Kalman filter," *IEICE Trans. Inf. Syst.*, vol. 91, no. 3, pp. 467–477, 2008.

[18] N. Ryant, M. Liberman, and J. Yuan, "Speech activity detection on YouTube using deep neural networks," in *Proc. Interspeech*, 2013, pp. 728–731.

[19] S. Meier and W. Kellermann, "Artificial neural network-based feature combination for spatial voice activity detection," in *Proc. Interspeech*, 2016, pp. 2987–2991.

[20] F. Vesperini, P. Vecchiotti, E. Principi, S. Squartini, and F. Piazza, "Deep neural networks for multi-room voice activity detection: Advancements and comparative evaluation," in *Proc. Int. Joint Conf. Neural Netw.*, 2016, pp. 3391–3398.

[21] T. Drugman, Y. Stylianou, Y. Kida, and M. Akamine, "Voice activity detection: Merging source and filter-based information," *IEEE Signal Process. Lett.*, vol. 23, no. 2, pp. 252–256, Feb. 2016.

[22] S. Thomas, S. Ganapathy, G. Saon, and H. Soltau, "Analyzing convolutional neural networks for speech activity detection in mismatched acoustic conditions," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, 2014, pp. 2519–2523.

[23] R. Zazo, T. N. Sainath, G. Simko, and C. Parada, "Feature learning with raw-waveform CLDNNs for voice activity detection," in *Proc. Interspeech*, 2016, pp. 8–12.

[24] D. A. Silva, J. A. Stuchi, R. P. V. Violato, and L. G. D. Cuozzo, "Exploring convolutional neural networks for voice activity detection," in *Cognitive Technologies*. Cham, Switzerland: Springer, 2017, pp. 37–47.

[25] T. Hughes and K. Mierle, "Recurrent neural networks for voice activity detection," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, 2013, pp. 7378–7382.

[26] F. Eyben, F. Weninger, S. Squartini, and B. Schuller, "Real-life voice activity detection with LSTM recurrent neural networks and an application to Hollywood movies," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, 2013, pp. 483–487.

[27] J. Kim, J. Kim, S. Lee, J. Park, and M. Hahn, "Vowel based voice activity detection with LSTM recurrent neural network," in *Proc. 8th Int. Conf. Signal Process. Syst.*, 2016, pp. 134–137.

[28] X.-L. Zhang and D.-L. Wang, "Boosting contextual information for deep neural network based voice activity detection," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 24, no. 2, pp. 252–264, Feb. 2016.

[29] V. Mnih, N. Heess, and A. Graves, "Recurrent models of visual attention," in *Proc. 27th Int. Conf. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2204–2212.

[30] J. Ba, V. Mnih, and K. Kavukcuoglu, "Multiple object recognition with visual attention," *Proc. Int. Conf. Learn. Represent.*, pp. 1–2, 2015.

[31] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2015, pp. 577–585.

[32] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016, pp. 152–153.

[33] J. Schulman, N. Heess, T. Weber, and P. Abbeel, "Gradient estimation using stochastic computation graphs," in *Proc. 28th Int. Conf. Adv. Neural Inf. Process. Syst.*, 2015, pp. 3528–3536.

[34] D. Samuel and L. Karam, "Understanding how image quality affects deep neural networks," in *Proc. 8th Int. Conf. Qual. Multimedia Experience*, pp. 1–6, 2016.

[35] J. Garofolo, "TIMIT acoustic-phonetic continuous speech corpus," Linguistic Data Consortium, Philadelphia, PA, USA, LDC93S1, 1993.

[36] Sound-ideas.com, "General series 6000 combo," 2012. [Online]. Available: https://www.sound-ideas.com/Product/51/General-Series-6000-Combo

[37] H. Hirsch, "FaNT: Filtering and noise adding tool," 2005. [Online]. Available: http://dnt.kr.hsnr.de/download.html

[38] A. Varga and H. J. M. Steeneken, "Assessment for automatic speech recognition: II. NOISEX-92: A database and an experiment to study the effect of additive noise on speech recognition systems," *Speech Commun.*, vol. 12, no. 3, pp. 247–251, Jul. 1993.

[39] J. Tracey *et al.*, "HAVIC pilot transcription LDC2016V01." Web Download, Linguistic Data Consortium, Philadelphia, PA, USA, 2016.

[40] J. Kim, "VAD-Toolkit," GitHub repository, 2017. [Online] Available: https://github.com/jtkim-kaist/VAD

[41] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *Proc. Int. Conf. Learn. Represent.*, pp. 1–41, 2015.

[42] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, Feb. 2012.

[43] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. 32nd Int. Conf. Mach. Learn.*, pp. 448–456, 2015.

[44] L. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," arXiv:1607.06450, 2016.

[45] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014.

[46] R. Caruana, S. Lawrence, and L. Giles, "Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping," in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 402–408.