Scale-invariant Online Voice Activity Detection under Various Environments

Ryu Takeda* and Kazunori Komatani*
* SANKEN, Osaka University, Japan
E-mail: {rtakeda, komatani}@sanken.osaka-u.ac.jp Tel: +81-6-6879-8416

***Abstract*—Online voice activity detection (VAD) is an important front-end for spoken dialogue systems. However, different signal amplitudes and speech distortions under various environments cause performance degradation of neural VAD models due to the model mismatch. The amplitude and distortion problems were addressed during the feature extraction and training processes of neural networks, respectively. First, the signal amplitude was normalized block-wise to ensures the scale invariance mathematically. Such block-wise normalization was naturally introduced in our formulation of online VAD based on a recursive Bayesian estimation of speech activity. Second, over 1,000 hours of training data was augmented by simulating speech distortions, such as reverberations. Our VAD outperformed open VAD models, such as Silero, for a variety of datasets including a real spoken dialogue dataset in terms of speech and non-speech discrimination. Codes and models have been publicly released in a toolkit.**

I. Introduction

A. Background and Motivation

Online voice activity detection (VAD) [1] is an important front-end technology of spoken dialogue systems. The role of VAD is to detect the active-speech sections, i.e. pairs of start and end times of speech, from the (noisy) input signals captured by microphones. Additionally, online/real-time processing of VAD is an essential requirement because the latency of VAD obviously leads to the response time from the system, which affects the user experience [2]. Since spoken dialogue systems, such as guide robots, need to continue to operate and serve on demand from various speakers under various environments [3], [4] including remote spoken dialogue [5], the demand of *tuning-free and robust* VAD and its publicly-released models increases for real system developers. We assume a single speaker and a monaural microphone for the input signal hereafter.

Audio input always depends on target speakers, acoustic environments and recording conditions (Figure 1). Their influences are observed as the different signal amplitudes and speech distortions. The *signal amplitude* is affected by the *gain setting of audio devices* and the audio format used in recording and by the volume of each utterance by speakers. For example, there are both quiet and loud speakers, and the dynamic range of audio signal differs among them. Speech distortions can occur due to the various reverberations and background non-speech signals, as well as the lossy compression of speech data. For example, audio signals are compressed to reduce the amount of communication data in the case of dialogues on remote conference applications. These factors are usually not controllable by system developers.

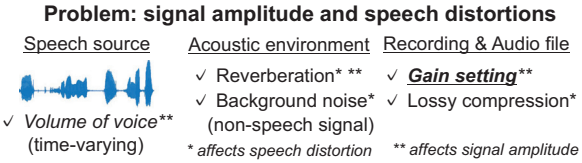

Fig. 1. Difficulties in VAD under various environments

Different signal amplitudes and speech distortions degrade the performance of current neural VAD models due to the model mismatch. Online VAD is more difficult than batch VAD because of the constraints of causal processing and low latency. Here, the online processing is different from batch processing that assumed in diarization tasks because the future inputs are not available (causality). For example, although Silero [6] is an open real-time VAD model that adopts the adaptive normalization of amplitude, its performance degrades for audio signals including long non-speech sections, background noises and reverberations with small amplitude. PyAnnote [7], [8] for diarization tasks (batch VAD) is not robust against background noises. Their models are based on a recurrent neural network (RNN) [9] (e.g., long short-term memory (LSTM) [10]), and a convolutional neural network (CNN) [11], [12]. Therefore, their actual performance indicates that the network structure or the training data of models were not designed for severe audio signals.

In light of this background, we propose a block-wise *scale-invariant* normalization and data augmentations for the amplitude and distortion problems, respectively. First, the signal amplitude was normalized in a block-wise manner to ensure the scale invariance of signal *mathematically*. Such block-wise normalization was naturally introduced in our formulation of online VAD based on a recursive Bayesian estimation of speech activity, i.e., a hybrid model of hidden Markov model (HMM) [13] and deep neural networks (DNN) using transformer-encoder. Second, over 1,000 hours of training data for VAD models was augmented by simulating speech distortions, such as reverberations. Since the process of DNN (transformer-encoder) is independent from recursive Bayesian filtering, its training becomes efficient for such large amount of data by mini-batch parallelization. The robustness of our VAD, Silero and PyAnnote was evaluated by using various open datasets including real spoken dialogue data.

The contributions of this paper are as follows.

- We focus on the problem of different signal amplitudes and speech distortions in terms of system development.
- We developed an online robust VAD based on scale-invariant transformation, and we also trained the model with data augmentation as solutions for the problems.
- A comprehensive evaluation of our VAD and two publically available VADs was conducted using various kinds of data in terms of scale invariance and noise robustness.
- Codes and models have been publicly released in the VAD toolkit[1].

B. Related Work

Although there has been extensive research on VAD, comprehensive research toward *tuning-free* and open-model VAD have not been tackled yet. While earlier works focused on clean speech signals, more recent works have examined additional functions such as online/real-time processing or noise robustness, as the performance of classifiers has been steadily improving thanks to the latest neural networks.

Standard VAD methods classify speech and non-speech sections by applying pattern recognition techniques. Traditionally, features and classifiers have been modeled separately. Examples of major features include energy [14], zero-crossing rate [15], and classifiers include logistic regression, support vector machine [16], and HMM [15]. The latest VADs are based on DNNs (RNN) which can optimize feature extraction and classifier simultaneously, e.g., feature learning with raw-waveform [17]. A variety of aspects of VAD has been also investigated: online VAD [18] with low-latency [19], noise-robust VAD using context information [20], [21], and integration of VAD with automatic speech recognition (ASR) [22].

II. Preliminaries

A. Problem Statement

Some variables are defined for our VAD formulation in the feature domain. The monaural signal is converted into a feature vector $\mathbf{x}_t$ at discrete time-frame index t, and the voice activity of speech at frame t is represented by its label $v_t \in \{0, 1\}$. If the speech exists at frame t, $v_t = 1$, otherwise, $v_t = 0$.

Online VAD is formulated as a sequential labelling problem, that is, the estimation of voice activity state z_t at frame t from the sequence of feature vectors $\mathbf{x}_{1:t} = [\mathbf{x}_1, ..., \mathbf{x}_t]$. The posterior probability $p(z_t|\mathbf{x}_{1:t})$ is usually utilized as its score, and the optimal $\hat{z}_t$ is estimated as

$$\hat{z}_t = \text{argmax}_{z_t} p(z_t|\mathbf{x}_{1:t}). \tag{1}$$

As an alternative to this criterion, $\hat{z}_t$ can be determined by thresholding the posterior probability. The definition of the voice activity state z_t may change depending on the design of VAD. For example, $z_t = v_t$ means the *filtering* setting, and $z_t = v_{t-m} (m > 0)$ means the *smoothing* setting in terms of sequential signal processing [23]. The setting of the allowed latency m depends on applications. We can also set the joint state of voice activity such as $z_t = (v_t, v_{t-m})$, which can model the transition of v_t explicitly.

Voice-active sections are estimated by using the $\hat{z}_t$ and its posterior probability. The detection is usually based on the transition of state $\hat{z}_t$, and the post-processing for the integration and the rejection of sections are usually applied. The detail of our implementation is explained in Section 3.

B. RNN Implementation

RNN series also used in Silero [6] can directly model the posterior probability $p(z_t|\mathbf{x}_{1:t})$ via hidden vector $\mathbf{g}_t$. The advantage of this approach is the end-to-end modelling whose parameters can be optimized in a supervised manner by using training data. Here, PyAnnote [7], [8] assumes a non-causal batch processing that is different from this sequential formulation.

The implementation of posterior probability *conceptually* consists of the function $\mathbf{G}$ for hidden vector extraction and the probability function $p(z_t|\mathbf{x}_t, \mathbf{g}_t)$, as

$$p(z_t|\mathbf{x}_{1:t}) := p(z_t|\mathbf{g}_t, \mathbf{x}_t), \text{and} \tag{2}$$

$$\mathbf{g}_t = \mathbf{G}(\mathbf{x}_t, \mathbf{g}_{t-1}) \tag{3}$$

where $\mathbf{g}_t$ represents all the hidden vectors among layers. Each function can be implemented by any networks, such as many-layered LSTM and CNN. The loss function is usually the corss-entropy between the probabilities of the ground truth label and the model's prediction.

C. DNN-HMM Implementation

A DNN-HMM is a state-space model [13] in which the likelihood is replaced by the point-wise posterior probability (snapshot estimation of state) using DNN [24]–[26]. The design of DNN posterior probability and the recursive Bayesian filtering are separated, that is, conditional independent each other in this model. Therefore, the training of the DNN becomes more efficient thanks to parallel processing using mini-batch.

DNN-HMM estimates the activity state z_t recursively by

$$p(z_t|\mathbf{x}_{1:t}) \propto \frac{p(z_t|\mathbf{x}_{1:t-1})p(z_t|\mathbf{x}_t; \mathbf{\Theta})}{p(z_t)}, \text{and} \tag{4}$$

$$p(z_t|\mathbf{x}_{1:t-1}) = \sum_{z_{t-1}} p(z_t|z_{t-1})p(z_{t-1}|\mathbf{x}_{1:t-1}), \tag{5}$$

where $p(z_t|\mathbf{x}_t; \mathbf{\Theta})$ is a posterior probability, $p(z_t)$ is a prior probability of voice state, and $p(z_t|z_{t-1})$ is a state transition probability. Here, the posterior $p(z_t|\mathbf{x}_t; \mathbf{\Theta})$ is modelled by a DNN with a parameter set $\mathbf{\Theta}$. The parameters of state transition and prior probabilities are trained or set manually if the size of them is small. The DNN parameters are also trained, and the cross-entropy is usually used as a loss function.

III. Proposed Online VAD

In this section, we first describe our strategy for the model, feature, and target state v_t for the proposed VAD. Then, we explain the network architecture including scale invariance

[1] pyadintool - https://github.com/ouktlab/pyadintool

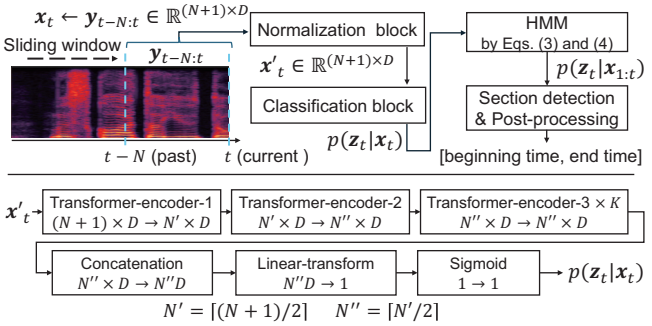

Fig. 2. Overview (top) and classification block (bottom)

transformation for the signal amplitude problem. Finally, we explain the training data augmentation for the speech distortion problems. The overview and relationships among each component are illustrated in Fig. 2.

A. Strategy for Model, Feature and Target State

We adopted a DNN-HMM because we can separate the design of the posterior from that of the sequential filtering. Since the latest neural architectures has a high discriminative ability, sequential filtering using HMM is enough for online VAD. The transformer encoder [27] is used to model the posterior probability $p(z_t|\mathbf{x}_t)$ because of its non-recurrent architecture and potential flexibility using prompts. The neural networks consist of normalization and classification blocks.

The input feature for DNN is a block of the amplitude spectrogram that is obtained by applying short-time Fourier transform (STFT) to the input signal. Given the D-dimensional amplitude spectrogram $\mathbf{y}_t \in \mathbb{R}^D$ in the STFT domain at frame t, our feature is the block-spectrogram $\mathbf{y}_{t-N:t} = [\mathbf{y}_{t-N}, ..., \mathbf{y}_t]$ from past $(t-N)$ to current t that corresponds to $\mathbf{x}_t$ in Eq. (1), that is, $\mathbf{x}_t \leftarrow \mathbf{y}_{t-N:t}$.

We applied the smoothing setting for Eq. (1) to estimate voice activities stably, in other words, our target state z_t for VAD is $v_{t-m} (m > 0)$. The parameter m controls the tradeoff between the latency and the estimation accuracy. We assumed about 200 ms for this delay that will be acceptable for ASR.

B. Scale-invariant Architecture and Post-processing

Normalization block: The scale of each block spectrogram $\mathbf{x}_t (\leftarrow \mathbf{y}_{t-N:t})$ is normalized to ensure the scale-invariant behavior of online VAD *mathematically*. We applied the *average-scale normalization* and layer normalization method [28]. While the latter normalization includes trainable parameters, the former normalization does not. The average-scale normalization also standardizes the parameter training behavior of the following networks including layer normalization.

Normalized block-spectrogram $\mathbf{x}'_t$ is obtained by normalizing each $\mathbf{x}_t$ by the following average-scale (avg.) and layer normalization (layer), respectively:

$$\tilde{\mathbf{x}}_t \xleftarrow[\text{avg.}]{} \frac{\mathbf{x}_t}{\text{avg}[\mathbf{x}_t] + \epsilon}, \tag{6}$$

$$\mathbf{x}'_t \xleftarrow[\text{layer}]{} \frac{\tilde{\mathbf{x}}_t - \text{avg}[\tilde{\mathbf{x}}_t]}{\sqrt{\text{var}[\tilde{\mathbf{x}}_t] + \epsilon}} \gamma + \beta, \tag{7}$$

where avg$[\cdot]$ and var$[\cdot]$ represent averaging and variance operator over all elements of $\mathbf{x}_t$, and ϵ is a small value for regularization. γ and β are trainable scale and offset parameters of layer normalization.

Classification block: Our classification block consists of transformer-encoder, linear transform and sigmoid layers. The input of this classification block is a normalized block-spectrogram $\mathbf{x}'_t$ with the dimension of $(N+1) \times D$, and the output is a posterior probability, i.e. $p(z_t|\mathbf{x}_t)$. Here, the transformer-encoder consists of multi-head attention and feed-forward networks. Positional encoding is applied to the input $\mathbf{x}'_t$ at first by considering the block-axis as a time-axis.

The three kinds of transformer-encoder networks were applied by reducing the block size $(N+1)$ to N''. This process can reduce the computational cost of these layers in proportion to the reduced block size. For example, the block sizes of each output from the first and second layers were halved as $N' = \lceil (N+1)/2 \rceil$ and $N'' = \lceil N'/2 \rceil$, respectively. The third transformer-encoder layer is repeated K times.

The block vectors from the three kinds of transformer-encoder layers were concatenated into a single vector, and linear transformation network was applied to it. The probability of voice activity state z_t was approximated by applying sigmoid function to the output from the linear transformation network.

Detection of Voice Active Section and Post Processing: The voice-active section is detected by using the estimated voice-activity state $\hat{z}_t$. In our case of $z_t = v_{t-m}$, the beginning frame t^s_k and the end frame t^e_k of the k-th voice-active section were detected by using the difference between $\hat{z}_t$ and $\hat{z}_{t-1}$. If $\hat{z}_t - \hat{z}_{t-1} = 1$, then it indicates the beginning frame: $t^s_k \leftarrow t - m$. If $\hat{z}_t - \hat{z}_{t-1} = -1$, then it indicates the end frame: $t^e_k \leftarrow t - m$.

The post processing is also applied to reject or merge the detected sections. The interval of the section is less than u_r, we discard it as a fluctuation. If the interval between the k-th and $(k+1)$-th sections is less than u_c, the two sections were merged into a new section t^s_k and t^e_{k+1}. Here, u_c should also be determined by the allowed latency of the system because it needs to wait for a while. Expansion of each section like $[t^s_k - u_m, t^e_{k+1} + u_m]$ with a margin parameter u_m is sometimes introduced to reduce the mismatch of voice-activity annotation criteria among datasets.

C. Augmentation Strategy for Speech Distortions

We augmented training data from the given seed audio signals. Three kinds of clean audio signals were used as the seed: clean speech signal, clean non-speech signal, and real impulse responses for reverberation simulation. Speech distortions were

simulated by combining with acoustic transformations excepts for the signal amplitude.

Our transformations for data augmentation are as follows.

1) Clean audio: Seed speech and non-speech signals.
2) Reverberant speech: Convolved speech signals by randomly selected impulse responses.
3) Reverberant and noisy speech: Non-speech signals added to the set 2) with randomly selected SNRs.
4) Collapsed speech: lossy compressions, such as mp3 and μ-low conversion, applied to the set 3).

After generating 2), 3) and 4), we applied oversampling [29] of the seed non-speech signals to relax the imbalance between active and non-active labels/data.

IV. Experiments

Our experiments investigate the *scale-invariance* and noise robustness of each VAD: two public VAD models (Silero with online processing and PyAnnote with batch processing) and our model (online processing). Note that the results of PyAnnote is reference because the processing style of VAD, such as online or batch (non-causal), usually affects VAD performance.

A. Data set

Seed corpora of speech: The corpora for speech signals mainly consist of nine *public* Japanese speech corpora: a core set of CSJ [30], S-JNAS, TWM, JEIDA-JCSD, ETL-WD copora[2], APP, APPDIC copora[3], SLC-3[4], and JVS [33]. These corpora cover over 4,000 speakers from 6 to 91 years old, utterances of words and sentences, and voices of whispers and falsetto. Extra utterances of single and double syllables were augmented by Google TTS and collected by our group, and the amount of its speech sections was about 50 hours. These total 677-hours speech data were downsampled to 16kHz.

Seed corpora of non-speech: The corpora for non-speech signals mainly consist of three *public* corpora: MUSAN [34], the training set of WHAM! [32], and the ProSoundEffects (PSE) corpus[5]. We also prepared 20 hours of different additional kinds of non-speech signals: pure tone signals, white/brown/pink noise signals, simulated babble noise signals, and environmental sound noise signals recorded by our group. These total 400-hours non-speech data were downsampled to 16kHz.

Training set (full and medium): The about 4,300 hours of the *full* training set consists of the seed non-speech corpora (four-times oversampled) and the augmented speech data. The transformations for augmentation were applied to the seed speech corpora, and we obtained the augmented speech data that was increased to 2,688 hours in total. The real impulse responses with 540 positions in a room of our group (RT_{20} 640 ms) were used to simulate *various real* reverberations. The SNR settings were randomly selected from $-10, -5, 0, 5, 10$, and 20 dB. The *medium* training set was randomly selected from the full set, and its amount was about 1,100 hours. The ratio between the speech and non-speech *sections* was $0.45 : 0.55$.

Test set: We prepared the three kinds of *open* test sets: non-speech, speech and dialogue sets. Non-speech set consist of *public* ESC-50 [31], the test set of WHAM! and the held-out set of PSE. Dialogue set consist of *public* real recorded spoken dialogue data called Hazumi 1911 and Hazumi2010 [5]. Speech set includes the clean test set of CSJ (*clean*), the simulated reverberant speech (*rev.*), and the simulated reverberant noisy speech (*rev.+bgn.*). Five kinds of impulse responses from the *public* RWCP-SSD corpus [35] were used to generate the reverberant speech from the CSJ test set. Then, signals in the non-speech set were added to them with randomly selected SNRs from $5, 10, 15$, and 20 dB. We changed their signal amplitude by multiplying $1.0, 0.5, 0.2$, and 0.05 and saved as 16bit-WAV format files to evaluate the robustness against amplitude changes.

Validation set: The validation set for monitoring the stationarity of F1-score change among epochs was constructed with different source signals by the same procedure as the test set: clean, rev. and rev.+bng sets. Source speech signals from the held-out set of CSJ, non-speech signals from the held-out set of PSE and the validation set of WHAM!. Different impulse responses were selected from the RWCP-SSD. The ratio between the speech and non-speech sections was almost equal.

B. Configurations

Our VAD model was built from scratch with the PyTorch library and its pre-defined classes [36]. The STFT parameters were the 512-point Hanning window and the 160-points shift (0.1 s). The label delay m was 20 (0.2 s), and the block size N was 50 (0.5 s). The transformer-encoder with eight-heads (default) and 512-hidden vectors was used, and the number of third transformer-encoders K was 2. The parameter initializations of each layer followed default setting of PyTorch. Gradient clipping [37] and Adam [38] were applied with clipping value 5 and learning rate 1.0×10^{-5}, respectively. The number of epochs was 30. The parameter sets of contiguous 10 epochs were **averaged** to reduce the influence of fluctuations at each epoch like [39], and then it was used for evaluations. The section (start and end epoch) for the parameter averaging was determined by the best corresponding averaged F1 score of *csj+rev+bgn.* set in our validation set. Here, the number of parameters of our models was five million while those of Silero (for 16k) and PyAnnote (for segmentation) public models were 0.15 and 1.5 million, respectively.

The parameters of post processing were the following: both u_c, and u_r were set to 10 (0.1 s), respectively. u_m was set to 10 (0.1 s) to reduce the influence of different annotation criteria among datasets. These parameters for PyAnnote and Silero were set to 0 because it performed best. Other hyper parameters of them were set to defaults. The prior probabilities

[2]https://research.nii.ac.jp/src/list.html
[3]https://www.atr-p.com/products/sdb.html
[4]https://alaginrc.nict.go.jp/slc-outline.html
[5]Pro Sound Effects Library. http://www.prosoundeffects.com

TABLE I

DETAILED RESULTS: F1-SCORE OF FRAME-WISE BINARY CLASSIFICATION IN PERCENTAGE. NOTE THAT THE RESULTS OF PYANNOTE ARE AS A REFERENCE BECAUSE OF ITS BATCH PROCESSING (NON-CAUSAL). THE LOWER AND UPPER BOUNDS OF ALMOST ALL CONFIDENCE INTERVALS WERE WITHIN ± 0.1. † AND ‡ INDICATE THAT THE CORRESPONDING BOUNDS WERE WITHIN ± 0.2 AND ± 0.3, RESPECTIVELY.

	Dataset	Non-speech set						Speech set (CSJ eval. [30])						Dialogue set [5]			
	Corpus/Condition	ESC-50 [31]		PSE		WHAM! [32]		clean		rev.		rev.+bgn.		Hazumi1911		Hazumi2010	
	Amplitude change	1.00	0.05	1.00	0.05	1.00	0.05	1.00	0.05	1.00	0.05	1.00	0.05	1.00	0.05	1.00	0.05
Baseline	Silero (online) [6]	†94.2	**99.2**	97.1	99.4	**94.8**	**100.0**	95.8	93.9	92.4	†72.5	†88.4	‡60.5	†85.9	†82.2	†92.6	†87.4
	PyAnnote (batch) [7]	93.6	†94.0	†95.2	95.9	87.4	91.5	**97.4**	**97.4**	91.5	91.5	90.8	90.4	†87.9	†87.1	†92.7	†92.4
Proposed (online)	medium training set	**97.3**	97.2	**98.6**	98.5	91.1	91.0	94.8	94.6	93.8	**94.0**	93.4	**93.4**	†89.9	†89.9	†92.9	†**92.9**
	full training set	**97.3**	97.3	98.3	98.3	93.4	93.4	94.6	94.5	93.0	93.5	92.9	92.9	†90.5	†**90.5**	†92.0	†92.8
Ablation	medium set w/o norm.	95.2	98.5	95.3	**99.6**	†84.0	99.8	94.4	†73.6	**94.4**	†69.0	**94.0**	†68.0	†**92.4**	‡60.0	†**94.0**	‡58.2

TABLE II

TOTAL PERFORMANCE AND REAL-TIME FACTOR OF PROPOSED METHOD

	Baseline		Proposed		Ablation
	Silero	PyAnnote	medium set	full set	medium w/o norm.
F1-score	90.51	92.48	**94.24**	94.14	89.80
Accuracy	89.45	90.75	**93.36**	93.20	89.64

	GPU: 1-board	CPU: 1-core	CPU: 2-core	CPU: 4-core
RTF	0.019	0.125	0.070	0.039

in HMM were set to 0.5 that means no prior knowledge, and the probability of self-transition was set to 0.99 manually.

C. Results

The evaluation metrics were frame-wise F1-score and accuracy, which were calculated as follows. First, the detected speech sections and ground-truth sections were converted into a sequence of binary voice-activities (0 or 1) at every 0.1-s frame for each test signals. The binary labels of the detection and the ground-truth compared to calculate the metrics by using the software [40] with 95% confidence-interval computation (without "condition" setting). The number of samples (labels) before applying amplitude changes was 1,469,155 for the non-speech set, 695,397 for the speech set, and 756,037 for the Hazumi set, respectively. **F1-score of non-active labels was used for non-speech set**. Here, the sections of system utterances in the Hazumi set were eliminated on the basis of their annotations.

Table I lists the F1-score results with the selected amplitudes 1.0 and 0.05. The baselines were Silero and PyAnnote, with PyAnnote listed as a *reference* due to its batch processing. The performance of our method without normalization processes was also investigated as an ablation study.

We found that the two baselines have some weaknesses. The change of signal amplitude and the speech distortion seriously degraded the performance of Silero by a maximum of about 20 points when we focus on the results of *rev.+bgn.* set in the Speech set. As for the Non-speech set, the performances under the small scale condition (0.05) were better than those under large scale condition (1.00). This indicates that the information of the absolute signal amplitude (not SNR) was implicitly used as a feature for classification in the Silero model. The performance of PyAnnote also degraded in the sets of WHAM! and *rev.+bgn.*, which may come from the training data set and the problem setting of PyAnnote. The performances of our methods with medium and full set training data were stable while our method *without normalization (ablation study)* did not work for different amplitude signals (over 20 points degradation for speech and dialogue set) even with the same number of parameters. These performance improvements related with the signal amplitude were brought mainly by our mathematical scale-invariant transformation, and slightly by the different number of parameters of each model.

Table II lists the macro-averaged F1-score and accuracy over non-speech, speech and dialogue sets as summary (data with four amplitude settings contained in each set). Our methods both with medium and full set outperformed others in both F1-sore and accuracy from 1 to 3 points. The performance of medium set model was slightly better than that of full set one just for data sets used in this experiment. Real-time factors (RTFs) with GPU (GeForce RTX 2080 Ti) and CPU (Intel Core i9-9980XE 3.0GHz) of our method are also shown. All RTFs were less than 0.15, and RTF with 2-core (2 threads) was less than 0.1. Thus, our VAD will not cause serious delays.

Limitations of our method are that 1) the balance between precision and recall depends on data set, 2) evaluations were limited to Japanese data, and 3) network structures were not optimized. As for the language dependency, our model may not cause serious performance degradations compared with natural language processing area because the speech feature used in ASR is usually similar among different languages. Of course, model training using multiple language data set will improve the VAD performance more thanks to the various of speech signals. Construction of lighter models is also a future work.

V. CONCLUSION

Online voice activity detection (VAD) is an important front-end for spoken dialogue systems. Different signal amplitudes and speech distortions under various environments cause performance degradation of neural VAD models due to the model mismatch. We addressed the two problems by scale-invariant normalization and data augmentation of various speech data, respectively. Our VAD outperformed Silero and PyAnnote for various datasets including a real spoken dialogue dataset. Future work includes the evaluation with other languages and spoken dialogue dataset.

ACKNOWLEDGMENT

This work was partly supported by JSPS KAKENHI Grant Numbers JP23H03457 and JP22H00536, and JST Moonshot R&D Grant Number JPMJPS2011, Japan.

REFERENCES

[1] B. Atal and L. Rabiner, "A pattern recognition approach to voiced-unvoiced-silence classification with applications to speech recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 24, no. 3, pp. 201–212, 1976. DOI: 10.1109/TASSP.1976.1162800.

[2] Y. Shangguan, R. Prabhavalkar, H. Su, J. Mahadeokar, Y. Shi, *et al.*, "Dissecting user-perceived latency of on-device E2E speech recognition," in *Proc. of Interspeech*, 2021, pp. 4553–4557. DOI: 10.21437/Interspeech.2021-1887.

[3] R. Higashinaka, T. Minato, K. Sakai, T. Funayama, H. Nishizaki, and T. Nagai, "Spoken dialogue system development at the dialogue robot competition," *The Journal of The Acoustic Society of Japan*, vol. 77, no. 8, pp. 512–520, 2021, ISSN: 03694232. DOI: 10.20697/jasj.77.8_512.

[4] T. Iio, Y. Yoshikawa, M. Chiba, T. Asami, Y. Isoda, and H. Ishiguro, "Twin-robot dialogue system with robustness against speech recognition failure in human-robot dialogue with elderly people," *Applied Sciences*, vol. 10, no. 4, 2020, ISSN: 2076-3417. DOI: 10.3390/app10041522.

[5] K. Komatani, R. Takeda, and S. Okada, "Analyzing differences in subjective annotations by participants and third-party annotators in multimodal dialogue corpus," in *Proc. of Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, 2023, pp. 104–113. DOI: 10.18653/v1/2023.sigdial-1.9.

[6] Silero Team, *Silero VAD: Pre-trained enterprise-grade voice activity detector (VAD), number detector and language classifier*, https://github.com/snakers4/silero-vad, 2021.

[7] H. Bredin, R. Yin, J. M. Coria, G. Gelly, *et al.*, "Pyannote.audio: Neural building blocks for speaker diarization," in *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2020, pp. 7124–7128.

[8] H. Bredin and A. Laurent, "End-to-end speaker segmentation for overlap-aware resegmentation," in *Proc. of Interspeech*, 2021, pp. 3111–3115.

[9] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*. Cambridge, MA, USA: MIT Press, 1986, pp. 318–362, ISBN: 026268053X.

[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997, ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735.

[11] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, pp. 193–202, 1980.

[12] Y. LeCun, B. Boser, J. Denker, *et al.*, "Handwritten digit recognition with a back-propagation network," in *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, vol. 2, 1989, pp. 396–404.

[13] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer-Verlag New York, 2006.

[14] L. Ye, W. Tong, C. Huijuan, and T. Kun, "Voice activity detection in non-stationary noise," in *Proc. of Multiconference on Computational Engineering in Systems Applications*, vol. 2, 2006, pp. 1573–1575. DOI: 10.1109/CESA.2006.4281886.

[15] J.-C. Junqua, B. Reaves, and B. Mak, "A study of endpoint detection algorithms in adverse conditions: Incidence on a DTW and HMM recognizer," in *Proc. of Eurospeech*, 1991, pp. 1371–1374. DOI: 10.21437/Eurospeech.1991-313.

[16] M. Baig, S. Masud, and M. Awais, "Support vector machine based voice activity detection," in *Proc. of International Symposium on Intelligent Signal Processing and Communications*, 2006, pp. 319–322. DOI: 10.1109/ISPACS.2006.364896.

[17] R. Zazo, T. N. Sainath, G. Simko, and C. Parada, "Feature learning with raw-waveform CLDNNs for voice activity detection," in *Proc. of Interspeech*, 2016, pp. 3668–3672. DOI: 10.21437/Interspeech.2016-268.

[18] P. R. Gudepu, J. M. Koroth, K. Sabu, and M. A. B. Shaik, "Dynamic encoder RNN for online voice activity detection in adverse noise conditions," in *Proc. of Interspeech*, 2023, pp. 5052–5056. DOI: 10.21437/Interspeech.2023-2466.

[19] M. Shi, Y. Shu, L. Zuo, *et al.*, "Semantic VAD: Low-latency voice activity detection for speech interaction," in *Proc. of Interspeech*, 2023, pp. 5047–5051. DOI: 10.21437/Interspeech.2023-598.

[20] X.-L. Zhang and D. Wang, "Boosting contextual information for deep neural network based voice activity detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 2, pp. 252–264, 2016. DOI: 10.1109/TASLP.2015.2505415.

[21] R. Masumura, K. Matsui, Y. Koizumi, T. Fukutomi, T. Oba, and Y. Aono, "Context-aware neural voice activity detection using auxiliary networks for phoneme recognition, speech enhancement and acoustic scene classification," in *Proc. of European Signal Processing Conference (EUSIPCO)*, 2019, pp. 1–5. DOI: 10.23919/EUSIPCO.2019.8902703.

[22] Y. Sudo, S. Muhammad, K. Nakadai, J. Shi, and S. Watanabe, "Streaming Automatic Speech Recognition with Re-blocking Processing Based on Integrated Voice Activity Detection," in *Proc. Interspeech*, 2022, pp. 4641–4645. DOI: 10.21437/Interspeech.2022-11216.

[23] S. Haykin, *Adaptive Filter Theory*, 4th. Upper Saddle River, NJ 07458: Prentice-Hall, 1991.

[24] S. Renals, N. Morgan, H. Bourlard, M. Cohen, and H. Franco, "Connectionist probability estimators in HMM speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 1, pp. 161–174, 1994. DOI: 10.1109/89.260359.

[25] F. Seide, G. Li, X. Chen, and D. Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transaction," in *Proc. of IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2011, pp. 24–29.

[26] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-R. Mohamed, N. Jaitly, *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, *et al.*, "Attention is all you need," in *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.

[28] J. L. Ba, J. R. Kiros, and G. E. Hinton, *Layer normalization*, arXiv:1607.06450, 2016. arXiv: 1607.06450 [stat.ML].

[29] N. V. Chawla, K. W. Bowyer, *et al.*, "SMOTE: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, no. 1, pp. 321–357, 2002.

[30] K. Maekawa, "Corpus of spontaneous Japanese: Its design and evaluation," in *Proc. of ISCA & IEEE Workshop on Spontaneous Speech Processing and Recognition*, 2003.

[31] K. J. Piczak, "ESC: Dataset for Environmental Sound Classification," in *Proc. of Annual ACM Conference on Multimedia*, Oct. 13, 2015, pp. 1015–1018, ISBN: 978-1-4503-3459-4. DOI: 10.1145/2733373.2806390.

[32] G. Wichern, J. Antognini, M. Flynn, *et al.*, "WHAM!: Extending speech separation to noisy environments," in *Proc. of Interspeech*, 2019, pp. 1368–1372.

[33] S. Takamichi, K. Mitsui, Y. Saito, T. Koriyama, N. Tanji, and H. Saruwatari, *JVS corpus: Free Japanese multi-speaker voice corpus*, arXiv:1908.06248, 2019. arXiv: 1908.06248 [cs.SD].

[34] D. Snyder, G. Chen, and D. Povey, *MUSAN: A Music, Speech, and Noise Corpus*, arXiv:1510.08484v1, 2015. eprint: 1510.08484.

[35] S. Nakamura, K. Hiyane, F. Asano, T. Nishiura, and T. Yamada, "Acoustical sound database in real environments for sound scene understanding and hands-free speech recognition," in *Proc. of the Second International Conference on Language Resources and Evaluation (LREC)*, 2000.

[36] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. of Advances in Neural Information Processing Systems 32 (NeurIPS)*, 2019, pp. 8024–8035.

[37] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *Proc. of International Conference on Machine Learning*, vol. 28, 2013, pp. 1310–1318.

[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. of International Conference on Learning Representations*, 2015.

[39] P. Izmailov, D. Podoprikhin, *et al.*, "Averaging weights leads to wider optima and better generalization," in *Proc. of Conference on Uncertainty in Artificial Intelligence*, 2018.

[40] L. Ferrer and P. Riera, *Confidence intervals for evaluation in machine learning*, https://github.com/luferrer/ConfidenceIntervals.