APSIPA Transactions on Signal and Information Processing, 2024, 13, e6

Original Paper

Heterogeneous Convolutional Recurrent Neural Network with Attention Mechanism and Feature Aggregation for Voice Activity Detection

YingWei Tan[1] and XueFeng Ding[1]*

[1] *Volkswagen-Mobvoi (Beijing) Information Technology Co., Ltd, Beijing, China*

ABSTRACT

Voice activity detection (VAD) is a fundamental prerequisite for tasks involving speech processing, particularly automatic speech recognition (ASR). Traditional supervised VAD systems employ a single type of network to acquire frame-level labels from the ASR pipeline, yet their detection performance often falls short of satisfactory levels, impeding the identification of high-quality speech by these systems. In this study, we present a novel heterogeneous convolutional recurrent neural network (HCRNN) with an attention mechanism and feature aggregation for voice activity detection. This approach effectively integrates the advantages of distinct networks, aiming to achieve superior performance in voice activity detection. We begin by presenting our detection framework, which employs a convolutional neural network (CNN) as the initial component of a long short term memory (LSTM) or gated recurrent unit (GRU) architecture. The feature map obtained from this front-end CNN is subsequently fed into the LSTM or GRU component of the system. The choice of LSTM or GRU lies in their ability to model long-term dependencies between inputs, a

*Corresponding author: YingWei Tan, ywtan@vw-mobvoi.com.

Received 24 August 2023; Revised 28 January 2024
ISSN 2161-1823; DOI 10.1561/116.00000158

crucial aspect in voice activity detection. To enhance the framework's performance, we introduce two novel attention mechanisms. The first mechanism focuses on the fusion of both spatial and channel-wise information within local receptive fields. Given an intermediate feature map, our module generates attention maps along two independent dimensions: channel and spatial. These attention maps are then multiplied with the input feature map to achieve adaptive feature refinement. The second attention mechanism is dedicated to discovering contextual features from embedded sequences using a multi-head self-attention (MHSA) layer. This layer allows the model to capture relationships between different elements within sequences, further enhancing the representation power of the system. Finally, refined features from the LSTM or GRU back-end are aggregated using either trainable scalar weights or vector-based attention weights. This aggregation step ensures that the most relevant features are emphasized, contributing to more accurate voice activity detection. To evaluate the efficacy of our proposed method, we conducted experiments on synthetic VAD datasets, Kaggle VAD datasets and AVA-speech datasets. The results demonstrate that the proposed method outperforms the baseline CRNN in low signal-to-noise ratio and noisy scenarios, exhibiting robustness against various noise types. Summarizing, our framework effectively integrates the strengths of CNN and RNN (LSTM or GRU) to enhance detection performance. The inclusion of attention mechanisms and feature aggregation further optimizes system performance, making it a promising approach for voice activity detection.

1 Introduction

In recent years, the application of deep learning (DL) in voice activity detection (VAD) has achieved remarkable performance in various downstream tasks. These DL-based VAD methods have demonstrated promising results, outperforming traditional algorithms in numerous applications, such as Renevey and Drygajlo [18], Ramirez *et al.* [17], Sohn *et al.* [20], and Tan *et al.* [21]. Due to its scalability, the proposed method enables us to effectively handle large-scale data, exhibiting superior generalization performance compared to previous machine learning techniques. VAD is also motivated by this trend to incorporate hybrid acoustic features as input to neural networks in Drugman *et al.* [5] and Meier and Kellermann [15], or by modeling contextual information using popular neural architectures such as Deep Neural Networks (DNN)

in Zhang and Wang [29], convolutional neural network (CNN) in Thomas *et al.* [22], Zazo Candil *et al.* [28], and Obuchi [16], recurrent neural network (RNN) in Gelly and Gauvain [7], Eyben *et al.* [6], and Hughes and Mierle [9], and convolutional recurrent neural network (CRNN) in Dinkel *et al.* [4].

The extraction of salient features in context using convolutional neural networks (CNNs) or deep neural networks (DNNs) has been demonstrated to be effective. Nevertheless, the inherent input length constraint of these models limits their ability to process long messages, which can be crucial for VAD tasks. Previous attempts to model contextual information using recurrent neural networks (RNNs) have not been promising. In this regard, the attention mechanism in Kim and Hahn [12], Jo *et al.* [11], and Sofer and Chazan [19] demonstrates practical high performance in various context-involving settings. ACAM in Kim and Hahn [12] uses LSTM and attention mechanism to generate VAD, which is essential for capturing the correlation between the hidden vectors of the encoder and decoder. In Jo *et al.* [11], VAD is based on the Self Attention (SA) framework. The idea behind this architecture is to model contextual information between acoustic input frames. Unfortunately, the SA mechanism is only used for short context frames. Also, without using a CNN on the input, the spectral information of the speech may not be fully exploited. Sofer and Chazan [19] utilizes a Convolutional Neural Network (CNN) to exploit the spatial information of the noisy input spectrum to extract a sequence of frame-wise embeddings, followed by a self-attention encoder designed to find contextual information from the embedding sequence. Unlike Jo *et al.* [11] which is used separately for each frame (with context frames), this method is able to process the entire waveform at once, allowing for long receptive fields.

In this study, we present a voice activity detection (VAD) system that combines CRNN modules with an attention mechanism and feature aggregation. First, the CRNN module is designed to capture frame-wise features and extend the effective receptive field. Second, the attention mechanism focuses on essential features while suppressing irrelevant ones. To this end, we introduce two novel attention mechanisms. On one hand, we employ a method that emphasizes meaningful features along the channel and spatial axes. This approach leverages the fact that convolution operations integrate cross-channel and spatial information to refine feature representation. On the other hand, we address less context-sensitive issues using the multi-head self-attention (MHSA) module. Finally, refined features from backend LSTMs or GRUs are aggregated based on trainable scalar weights or vector-based attention weights across different scales. Our results demonstrate that the combined architecture outperforms individual components. Furthermore, we demonstrate that the proposed method not only achieves state-of-the-art performance on various benchmarks but is also computationally efficient, processing the entire waveform in a single step.

2 Methods

The VAD systems proposed in this paper take an acoustic feature vector extracted from the audio as an input, and should produce frame-level class labels for two categories: non-speech (ns) and speech (s). We implemented the first six different architectures to built VAD systems, as illustrated by Figure 1. The last two diverse methods are built in Figure 2. Table 1 shows different configurations for eight algorithms.

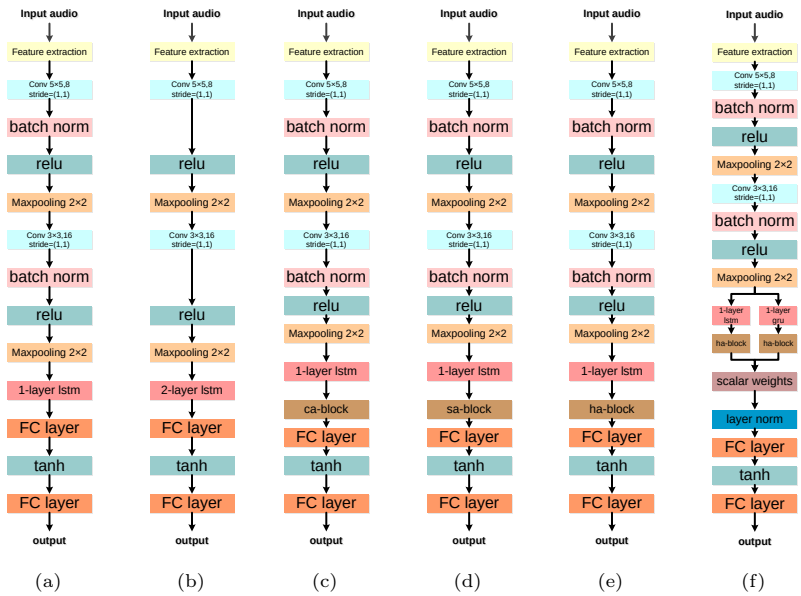

Figure 1: The first six different methods for implementing VAD

Table 1: Different configurations for eight algorithms in Figures 1 and 2.

algorithm	batch norm	1-layer lstm	2-layer lstm	ca-block	sa-block	ha-block	scalar weights	mhsa-block	vector-based attention weights
Figure 1a	✓	✓	✗	✗	✗	✗	✗	✗	✗
Figure 1b	✗	✗	✓	✗	✗	✗	✗	✗	✗
Figure 1c	✓	✓	✗	✓	✗	✗	✗	✗	✗
Figure 1d	✓	✓	✗	✗	✓	✗	✗	✗	✗
Figure 1e	✓	✓	✗	✗	✗	✓	✗	✗	✗
Figure 1f	✓	✓	✗	✗	✗	✓	✓	✗	✗
Figure 2a	✓	✓	✗	✗	✗	✗	✗	✓	✗
Figure 2b	✓	✓	✗	✗	✗	✗	✗	✓	✓

To mimic the nonlinear response to the human ear's acoustic spectrum, we choose a logarithmic mel-scale filter bank energy (fbank). In order to prepare for input into the neural network, the adjacent 11 acoustic feature frames are concatenated as input to the convolution operation.

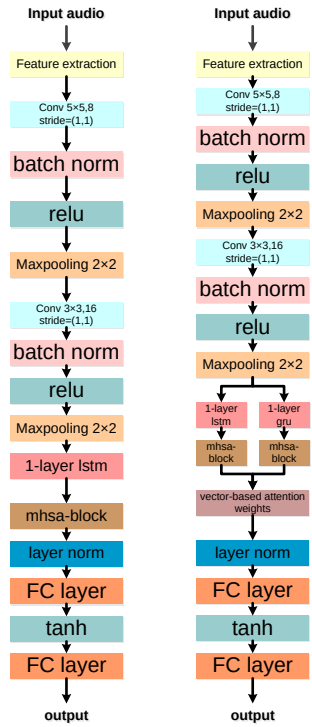

Figure 2: The last two different methods for implementing VAD

2.1 The CRNN architectures

Our first approach to implementing a VAD system utilizes a convolutional recurrent neural network (CRNN) backend. The architecture consists of a two-layer CNN (utilizing 5×5 and 3×3 convolutions), each followed by 2×2 max pooling. A single layer of long short-term memory (LSTM) is appended after the last CNN output, enhancing the temporal consistency of our model. The final two layer is fully-connected layers, which outputs whether speech exists in each frame. The framework and specific parameters can be seen in Figure 1a.

The second method is depicted as Figure 1b. Compared with Figure 1a, The operation of batch normalization is excluded. A single layer long short term memory (LSTM) will be replaced with a two-layer long short term memory (LSTM). Others remain unchanged. We also tried larger networks but saw no performance gain, possibly due to the limited diversity of the training data.

2.2 The channel attention module

As shown in Figure 1c, a channel attention block (ca-block) is inserted after the single-layer long short term memory (LSTM). The idea of channel-wise attention is borrowed from SENets in Hu *et al.* [8]. Specifically, the ca-block in Figure 3 consists of two operations: squeeze and excitation. The squeeze operation employs global average pooling of feature maps to generate channel statistics. Formally, the statistics $\mathbf{z} \in \mathbb{R}^C$ are generated by shrinking feature maps $\mathbf{U} = \left[\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_C\right]$ through its spatial dimensions $H \times W$, so that the c-th element of $\mathbf{z}$ is calculated by:

$$\mathbf{z}_c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} u_c(i, j) \tag{1}$$

Therefore, $C \times H \times W$ feature maps are reduced to $C \times 1 \times 1$ channel-wise statistics where H is the input height, W is the input width, and C is the number of channels. The excitation operation then takes the channel-wise statistics as inputs and computes the scaled values with a range of $\left[0, 1\right]$ through two fully-connected (FC) layers forming a bottleneck. A simple gating mechanism with sigmoid activation is formulated as follows:

$$\mathbf{s} = \sigma(\mathbf{W}_2 \delta(\mathbf{W}_1 \mathbf{z})) \tag{2}$$

where σ refers to the sigmoid function, δ means the ReLU function, $\mathbf{W}_1 \in \mathbb{R}^{\frac{C}{r} \times C}$, $\mathbf{W}_2 \in \mathbb{R}^{C \times \frac{C}{r}}$ and r is a reduction ratio. The final output of the ca-block is obtained by rescaling $\mathbf{U}$ with the activations $\mathbf{s}$:

$$\mathbf{x}_c = \mathbf{s}_c \otimes \mathbf{u}_c \tag{3}$$

where $\mathbf{X} = \left[\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_C\right]$, $\mathbf{s} = \left[\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_C\right]$, and $\otimes$ refers to channel-wise multiplication.

2.3 The spatial attention module

A spatial attention block (sa-block) can be seen in Figure 1d. According to Woo *et al.* [26], we exploit the inter-spatial relationships of features to generate spatial attention maps. To compute the spatial attention, we first apply average-pooling and max-pooling operations along the channel axis and concatenate them to generate an efficient feature descriptor. On the concatenated feature descriptor, we apply a convolution layer to generate a spatial attention map $\mathbf{M}(\mathbf{U}) \in \mathbb{R}^{H \times W}$ which encodes where to emphasize or suppress.

We aggregate channel information of a feature map by using two pooling operations, generating two 2D maps: $\mathbf{U}^{\mathbf{avg}} \in \mathbb{R}^{1 \times H \times W}$ and $\mathbf{U}^{\mathbf{max}} \in \mathbb{R}^{1 \times H \times W}$.

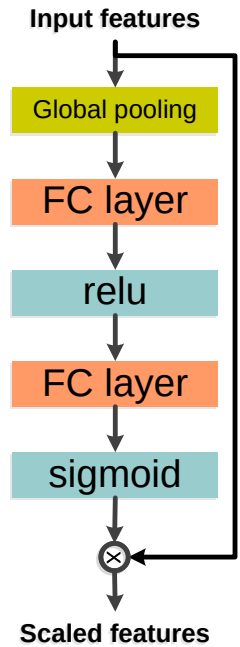

Figure 3: An overview of the proposed ca-block

Each denotes average-pooled features and max-pooled features across the channel. Those are then concatenated and convolved by a standard convolution layer, producing our 2D spatial attention map. In short, the spatial attention is computed as:

$$\begin{aligned}\mathbf{M}(\mathbf{U}) &= \delta(f^{3\times3}(\big[AvgPool(\mathbf{U}); MaxPool(\mathbf{U})\big])) \\ &= \delta(f^{3\times3}(\big[\mathbf{U^{avg}}; \mathbf{U^{max}}\big]))\end{aligned} \tag{4}$$

where δ is the sigmoid function and $f^{3\times3}$ represents a convolution operation with the filter size of 3×3. The spatial attention process can be summarized as:

$$\mathbf{Y} = \mathbf{M}(\mathbf{U}) \otimes \mathbf{U} \tag{5}$$

where $\mathbf{Y}$ is the final refined output. This detailed operation is described in Figure 4.

2.4 The hybrid attention module

Figure 1e shows a hybrid attention block (ha-block) applied. The hybrid attention module is bred from the ca-block and sa-block. Given an intermediate

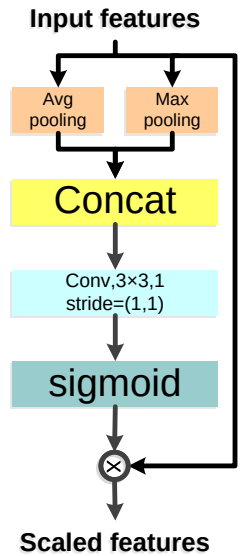

Figure 4: An overview of the proposed sa-block

feature map $\mathbf{U} \in \mathbb{R}^{C\times H\times W}$ as input, the ha-block infers a 1D channel attention map $\mathbf{s} \in \mathbb{R}^{C\times 1\times 1}$ and a 2D spatial attention map $\mathbf{M} \in \mathbb{R}^{1\times H\times W}$. The overall attention process can be summarized as:

$$\hat{\mathbf{X}} = \mathbf{s}(\mathbf{U}) \otimes \mathbf{M}(\mathbf{U}) \otimes \mathbf{U} \tag{6}$$

where $\hat{\mathbf{X}}$ is the final refined output. In our hybrid attention block (Figure 5), we exploit both spatial and channel-wise attention based on an efficient architecture and empirically verify that exploiting both is superior to using only the channel-wise attention as Hu *et al.* [8].

2.5 The scalar weights module

A feature aggregation module is shown in Figure 1f. The feature aggregation module is based on scalar weights and aims to capture more information for important estimated features. Given a feature matrix, this method assigns weights to each element of the matrix through a trainable layer. The matrix

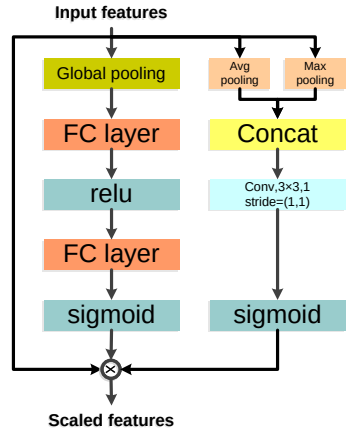

Figure 5: An overview of the proposed ha-block

$$\mathbf{Q} = \begin{bmatrix} q_{11} & q_{12} & \cdots & q_{1j} & \cdots & q_{1m} \\ q_{21} & q_{22} & \cdots & q_{2j} & \cdots & q_{2m} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ q_{i1} & q_{i2} & \cdots & q_{ij} & \cdots & q_{im} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ q_{n1} & q_{n2} & \cdots & q_{nj} & \cdots & q_{nm} \end{bmatrix} \tag{7}$$

is a parameter which can be learned by model optimization, where i is the feature dimension index, n is the total dimension, j is the RNN category index, and m is the total number of RNN categories used. The RNN categories are GRU and LSTM. m is set to 2 here. The associated scalar weights for each element of a feature row can be defined as follows:

$$\omega_{ij} = \tau(q_{ij}) \tag{8}$$

where τ is the softmax activation function calculated as follows:

$$\tau(q_{ij}) = \frac{\exp(q_{ij})}{\sum_{j=1}^{m} \exp(q_{ij})} \tag{9}$$

Given the output feature map $\mathbf{f}^{\mathbf{lstm}} \in \mathbb{R}^{C \times H \times W}$ from ha-block after the LSTM branch and the output feature map $\mathbf{f}^{\mathbf{gru}} \in \mathbb{R}^{C \times H \times W}$ from ha-block after the GRU branch, the two feature maps are reshaped to 1D feature vectors $\hat{\mathbf{f}}^{\mathbf{lstm}} = \left[\hat{f}_1^{\mathbf{lstm}}, \hat{f}_2^{\mathbf{lstm}}, \cdots, \hat{f}_i^{\mathbf{lstm}}, \cdots, \hat{f}_n^{\mathbf{lstm}}\right]$ and $\hat{\mathbf{f}}^{\mathbf{gru}} = \left[\hat{f}_1^{\mathbf{gru}}, \hat{f}_2^{\mathbf{gru}}, \cdots, \hat{f}_i^{\mathbf{gru}}, \cdots, \hat{f}_n^{\mathbf{gru}}\right]$, respectively. Here, $n = C \times H \times W$ is the

feature dimension. It is also the hidden state number in LSTM or GRU. The feature vectors are aggregated by the following equations:

$$\hat{f}_i = \omega_{i1}\hat{f}_i^{\mathbf{lstm}} + \omega_{i2}\hat{f}_i^{\mathbf{gru}} \tag{10}$$

The weighted feature vector is $\hat{\mathbf{f}} = \left[\hat{f}_1, \hat{f}_2, \cdots, \hat{f}_i, \cdots, \hat{f}_n\right]$. Then, we feed them into a LayerNorm layer.

2.6 The mhsa module

A multi-head self-attention block (mhsa-block) can be described in Figure 2a. The mhsa-block in Vaswani *et al.* [24] is the core of prominent architectures in multiple Machine Learning domains such as Natural Language Processing (NLP) and Computer Vision (CV). The main goals of the mhsa encoder is mapping a query and a set of key-value pairs to an output. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed using the given query with the corresponding key. An intermediate feature vector $\mathbf{v}_t$ is converted into an embedding vector $\mathbf{e}_t$ to be fed as an input of the self-attention network. In this process, the corresponding embeddings provided to the model are computed as follows:

$$\mathbf{e}_t = \mathbf{v}_t + \mathbf{p}_t \tag{11}$$

where t is the desired position in an input sequence, and the positional embedding $\mathbf{p}_t$ as a vector containing pairs of sines and cosines for each frequency. It is defined as:

$$\mathbf{p}_t = \begin{bmatrix} \sin(\omega_0 \cdot t) \\ \cos(\omega_0 \cdot t) \\ \sin(\omega_1 \cdot t) \\ \cos(\omega_1 \cdot t) \\ \vdots \\ \sin(\omega_{d/2-1} \cdot t) \\ \cos(\omega_{d/2-1} \cdot t) \end{bmatrix} \tag{12}$$

where $\omega_k = \frac{1}{10000^{2k/d}}$ and $d = 160$ is the encoding dimension. Given the frame-wise embedding sequence $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_t, \mathbf{e}_L] \in \mathbb{R}^{d \times L}$ where L is the input sequence length, a single layer attention of the multi-head attention with $H = 4$ attention heads is computed as:

$$\text{MultiHead} = Concat(head_0, \cdots, head_{H-1})W^O \tag{13}$$

where

$$\begin{aligned} head_i &= softmax(\frac{Q_i K_i^T}{\sqrt{d}})V_i, \\ Q_i = \mathbf{E} \cdot W_i^Q, K_i &= \mathbf{E} \cdot W_i^K, V_i = \mathbf{E} \cdot W_i^V \end{aligned} \tag{14}$$

The terms (W_i^Q, W_i^K, W_i^V, W^O) are the learned parameter sets of the multi-head attention layer. Then, we feed them into a LayerNorm layer.

2.7 The vector-based attention weights module

Figure 2b depicts a feature aggregation module using vector-based attention weights methods. The vector-based attention was proposed in Chen *et al.* [3] for sentence embedding and in Wu *et al.* [27] for text-independent speaker verification. Such attention provides a vectorial attention weight for each feature sequence estimated by algorithm, collecting more discriminative information.

Here, we ignore the frame index t. Given the output feature vector $\mathbf{f^{lstm}} \in \mathbb{R}^n$ from mhsa-block after the LSTM branch and the output feature vector $\mathbf{f^{gru}} \in \mathbb{R}^n$ from mhsa-block after the GRU branch, the two feature vectors are $\mathbf{f^{lstm}} = \left[f_1^{\mathbf{lstm}}, f_2^{\mathbf{lstm}}, \cdots, f_i^{\mathbf{lstm}}, \cdots, f_n^{\mathbf{lstm}}\right]$ and $\mathbf{f^{gru}} = \left[f_1^{\mathbf{gru}}, f_2^{\mathbf{gru}}, \cdots, f_i^{\mathbf{gru}}, \cdots, f_n^{\mathbf{gru}}\right]$, respectively. The input vector of vector-based attention methods is defined as $\mathbf{h}_i = \begin{bmatrix} f_i^{\mathbf{lstm}} \\ f_i^{\mathbf{gru}} \end{bmatrix}$. The corresponding attention weight $\mathbf{a_i} = \left[\omega_{i1}, ..., \omega_{ij}, ..., \omega_{im}\right]$ for each element of the vector can be computed as:

$$\mathbf{a_i} = \tau(\mathbf{W_2} f(\mathbf{W_1 h_i} + \mathbf{b_1}) + \mathbf{b_2})^{\mathrm{T}} \tag{15}$$

where $\mathbf{W_1} \in \mathbb{R}^{d \times 2}$ and $\mathbf{W_2} \in \mathbb{R}^{2 \times d}$ are weight matrices, $\mathbf{b_1} \in \mathbb{R}^d$ and $\mathbf{b_2} \in \mathbb{R}^2$ are bias items, $d = 16$ is a hyper-parameter, $m = 2$ is the total number of used RNN categories, $f(\cdot)$ is a non-linear activation function, e.g. ReLU. The softmax function ensures that the sum of all elements is 1 in the weight vector $\mathbf{a_i}$. Each element ω_{ij} of the vector $\mathbf{a_i}$ is the attention weight for the element of the $\mathbf{h_i}$. The final enhanced feature vector is formulated as follows:

$$\mathbf{f} = \left[f_1, ..., f_i, ..., f_n\right] \tag{16}$$

where $f_i = \omega_{i1} f_i^{\mathbf{lstm}} + \omega_{i2} f_i^{\mathbf{gru}}$. Then, we feed them into a LayerNorm layer.

3 Experiments

3.1 Datasets

3.1.1 Synthetic VAD datasets

To evaluate the effectiveness of the proposed methods, we resort to the AISHELL-1 corpus set by Bu *et al.* [1]. We choose it because it is open and available. 340 speakers' data is employed for training (120098 utterances, 340 hours). Test set contains 7176 utterances from 20 speakers. The corpus is sampled at 16 kHz with 16 bit quantization. For all the datasets, to produce the

frame-level ground truth personal VAD labels used in training and evaluation, we run forced alignment with a pretrained speech recognition model.

To generate noisy speech for evaluating the robustness of the proposed algorithm, white noise, babble noise, factory1 noise, factory2 noise, and pink noise from Noisex-92 in Varga and Steeneken [23] are added to the clean speech. a total of 25 different conditions are considered: 5 types of noise with 5 signal-to-noise ratio (SNR) each (-10, -5, 0, 5, and 10 dB). To make the train set, we randomly select 30000 utterances from the AISHELL-1 training set. There is a noisy condition of random selection for each utterance in training. To make the test set, we use the whole AISHELL-1 testing set (7176 utterances) and consider that the test incorporates the type of noise (f16 and leopard) that is absent in the train set.

3.1.2 Kaggle VAD datasets

Kaggle VAD datasets (kaggle-speech) in lazyrac00n [14] collect 719 audio from three different databases (TIMIT, PTDB-TUG and Noizeus). one of those creates an annotation file which write, in a txt format, where the silent and sounding intervals are in the signal. Noise signal (from AURORA database) is artificially added to the speech signal, in particular the database contains audio corrupted with babble (crowd of preople), street, train, train station, car and restaurant noise at SNRs of 5dB, and the original ones. To make the train set, we randomly select 519 utterances. To make the test set, we select the remaining 200 utterances.

3.1.3 AVA-speech datasets

AVA-Speech is a publicly available dataset of movies densely labeled with speech activity in Chaudhuri *et al.* [2]. At time of writing, AVA-Speech consisted of 160 segments from movies hosted on YouTube, each 15 minutes in duration, totalling 40 hours of labelled data. The segments are densely labelled for speech activity using the following mutually exclusive labels: “NoSpeech”, “CleanSpeech”, “Speech+Music” and “Speech+Noise”. We select 39 segments from it and make 3988 utterances ranging from 3s to 15s. To make the train set, we randomly select 3188 utterances. To make the test set, we select the remaining 800 utterances.

3.2 Experimental setup

In order to train the models, 6 epochs were utilized for the synthetic VAD datasets, while 100 epochs were allocated for both the Kaggle VAD datasets and AVA-speech datasets. The model is trained with Adam optimizer in Kingma

and Ba [13] and a learning rate of 0.00005, using an effective batch size of 16. We use a LSTM or GRU network with 160 cells, followed by a fully-connected layer with 160 neurons.

To evaluate the performance of the proposed approach, we calculated the Average Precision (AP) in Zhu [30] for each class and the mean Average Precision (mAP) over all the classes.

4 Results

With an input frame $\mathbf{x}$ and the corresponding ground truth label $y \in \{\text{ns}, \text{s}\}$, The VAD task can be thought of as a binary classification problem. The network outputs the unnormalized distribution of $\mathbf{x}$ over the two classes, denoted as $\mathbf{z} = f_{\text{VAD}}(\mathbf{x})$. We use $\mathbf{z}^k$ to denote the unnormalized probability of the k-th class. To train the model, we minimize the cross entropy loss as:

$$\mathcal{L}_{\text{CE}}(y, \mathbf{z}) = -\log \frac{\exp(\mathbf{z}^y)}{\sum_k \exp(\mathbf{z}^k)} \tag{17}$$

where $k \in \{\text{ns}, \text{s}\}$.

Results of the first six different methods for synthetic VAD datasets are listed in Table 2. It suggest that each of the components contribute to overall performance, with hybrid attention mechanism and feature aggregation based on scalar weights yielding the largest performance boost. The proposed model in Figure 1f shows decent performance for white and babble noises, which are present in the train set. A similar result for comparably f16 scenario, which is absent in the train set, displays the robustness of the proposed algorithm. The proposed method in Figure 1f outperforms the other methods for all the noisy conditions, except the leopard.

This states the benefit of this approach. It is further evident that our model, which combines a CRNN embedder before the hybrid attention encoder and applies the feature aggregation based on scalar weights, gains even better results. The parameter of the proposed model is 389K which is much smaller than 440K of the algorithm with two-layer LSTM structures in Figure 1b, with better performance (Table 2). This suggests that the proposed model is robust enough to maintain the performance in a harsh environment while lessening the parameter size.

In Table 3, the results for synthetic VAD datasets prove the strength of the proposed model, which merges a CRNN embedder before the mhsa block and employs the feature aggregation based on vector-based attention weights, with similar or better performance. We show that the fusion of mhsa-block and vector-based attention weights outperforms methods based solely on mhsa-block. Figure 6 further confirms this conclusion.

Table 2: Comparison results of the first six proposed algorithm in white, babble, f16 and leopard noises with different SNRs. We report the Average Precision (AP) for each class, and the mean Average Precision (mAP) over all the classes.

Noise scenario		Figure 1a			Figure 1b			Figure 1c		
Noise	SNR	ns	s	mean	ns	s	mean	ns	s	mean
white	-10	0.7861	0.9244	0.8959	0.7897	0.9258	0.9023	0.8026	0.9264	0.9058
	-5	0.8296	0.9340	0.9149	0.8121	0.9306	0.9101	0.8319	0.9315	0.9155
	0	0.8455	0.9373	0.9218	0.8251	0.9324	0.9141	0.8465	0.9341	0.9202
	5	0.8535	0.9386	0.9245	0.8366	0.9329	0.9171	0.8530	0.9351	0.9224
	10	0.8590	0.9391	0.9264	0.8431	0.9330	0.9183	0.8575	0.9359	0.9244
babble	-10	0.6874	0.8964	0.8650	0.6754	0.8932	0.8594	0.6637	0.8767	0.8486
	-5	0.7408	0.9137	0.8852	0.7200	0.9089	0.8777	0.7248	0.9046	0.8762
	0	0.7945	0.9264	0.9027	0.7730	0.9227	0.8966	0.7829	0.9220	0.8981
	5	0.8286	0.9329	0.9129	0.8140	0.9309	0.9097	0.8226	0.9301	0.9109
	10	0.8463	0.9361	0.9182	0.8376	0.9340	0.9172	0.8429	0.9337	0.9166
f16	-10	0.5655	0.8180	0.7870	0.6512	0.8834	0.8467	0.5614	0.7960	0.7700
	-5	0.6608	0.8896	0.8543	0.6983	0.9048	0.8693	0.6445	0.8748	0.8362
	0	0.7450	0.9189	0.8876	0.7610	0.9215	0.8900	0.7355	0.9139	0.8772
	5	0.7869	0.9283	0.9026	0.8085	0.9300	0.9050	0.7828	0.9250	0.8951
	10	0.8137	0.9323	0.9119	0.8328	0.9330	0.9151	0.8091	0.9292	0.9076
leopard	-10	0.5288	0.7982	0.7618	**0.6778**	0.8921	**0.8447**	0.5987	**0.8959**	0.8417
	-5	0.5694	0.8369	0.7934	**0.7207**	0.9067	0.8599	0.7002	**0.9152**	**0.8611**
	0	0.6407	0.8788	0.8297	0.7655	0.9187	**0.8749**	**0.7677**	**0.9242**	0.8708
	5	0.7197	0.9123	0.8599	**0.8075**	0.9282	**0.8908**	0.8029	**0.9286**	0.8756
	10	0.7669	0.9258	0.8727	**0.8356**	**0.9332**	**0.9050**	0.8243	0.9315	0.8787
network parameters		233906			439618			233988		
Noise scenario		Figure 1d			Figure 1e			Figure 1f		
Noise	SNR	ns	s	mean	ns	s	mean	ns	s	mean
white	-10	0.7974	0.9297	0.9076	0.7869	0.9271	0.9035	**0.8188**	**0.9353**	**0.9140**
	-5	0.8315	0.9358	0.9185	0.8240	0.9333	0.9156	**0.8419**	**0.9406**	**0.9225**
	0	0.8485	0.9377	0.9232	0.8439	0.9358	0.9210	**0.8527**	**0.9426**	**0.9247**
	5	0.8570	0.9384	0.9261	0.8542	0.9368	0.9241	**0.8595**	**0.9432**	**0.9247**
	10	0.8625	0.9388	0.9279	0.8594	0.9374	0.9258	**0.8649**	**0.9434**	**0.9285**
babble	-10	0.6639	0.8767	0.8477	0.6747	0.8837	0.8550	**0.7033**	**0.9062**	**0.8703**
	-5	0.7225	0.9049	0.8759	0.7278	0.9064	0.8774	**0.7758**	**0.9246**	**0.8968**
	0	0.7844	0.9234	0.9000	0.7831	0.9229	0.8980	**0.8238**	**0.9346**	**0.9149**
	5	0.8248	0.9319	0.9139	0.8213	0.9314	0.9114	**0.8479**	**0.9397**	**0.9236**
	10	0.8450	0.9355	0.9210	0.8414	0.9348	0.9179	**0.8604**	**0.9419**	**0.9281**
f16	-10	0.5243	0.8161	0.7733	0.6091	0.8305	0.7977	**0.6965**	**0.8951**	**0.8606**
	-5	0.6202	0.8891	0.8464	0.6755	0.8885	0.8464	**0.7724**	**0.9246**	**0.8942**
	0	0.7012	0.9187	0.8834	0.7501	0.9191	0.8784	**0.8247**	**0.9375**	**0.9135**
	5	0.7469	0.9282	0.9014	0.7987	0.9296	0.8970	**0.8510**	**0.9423**	**0.9210**
	10	0.7844	0.9321	0.9095	0.8215	0.9326	0.9104	**0.8649**	**0.9436**	**0.9244**
leopard	-10	0.5636	0.8573	0.8087	0.6362	0.8911	0.8384	0.5622	0.82550	0.7813
	-5	0.5750	0.8796	0.8269	0.6688	0.9096	0.8548	0.6078	0.8657	0.8155
	0	0.6301	0.9030	0.8482	0.7029	0.9201	0.8650	0.6746	0.8988	0.8452
	5	0.7086	0.9209	0.8662	0.7325	0.9253	0.8708	0.7445	0.9196	0.8653
	10	0.7607	0.9296	0.8758	0.7571	0.9278	0.8741	0.7958	0.9304	0.8768
network parameters		233924			234006			388986		

Figure 7 shows the receiver operating characteristic (ROC) curves for all systems tested on kaggle-speech. We see that the Figure 2b systems we propose outperform the CNN-biLSTM VAD in Wilkinson and Niesler [25] and the MARBLENET VAD in Jia *et al.* [10] across all operating points. When threshold=0.5, the CNN-biLSTM VAD, the MARBLENET VAD, and the Figure 2b method obtain accuracy rates of 91.49%, 91.56% and 90.99%,

Table 3: Comparison results of the last two proposed algorithm in white, babble, f16 and leopard noises with different SNRs. We report the Average Precision (AP) for each class, and the mean Average Precision (mAP) over all the classes.

Noise scenario		Figure 2a			Figure 2b		
Noise	SNR	ns	s	mean	ns	s	mean
white	-10	0.8623	0.9381	0.9310	**0.8696**	**0.9405**	**0.9313**
	-5	0.8792	0.9422	0.9380	**0.8841**	**0.9450**	**0.9397**
	0	0.8865	0.9438	0.9414	**0.8904**	**0.9469**	**0.9437**
	5	0.8892	0.9441	0.9428	**0.8927**	**0.9474**	**0.9454**
	10	0.8896	0.9436	0.9432	**0.8932**	**0.9473**	**0.9461**
babble	-10	**0.7957**	0.9148	**0.9056**	0.7953	**0.9163**	0.9034
	-5	0.8199	0.9254	0.9159	**0.8268**	**0.9299**	**0.9180**
	0	0.8467	0.9347	0.9263	**0.8563**	**0.9395**	**0.9300**
	5	0.8682	0.9403	0.9339	**0.8754**	**0.9444**	**0.9373**
	10	0.8811	0.9429	0.9387	**0.8849**	**0.9465**	**0.9411**
f16	-10	0.7792	0.9099	0.8919	**0.8112**	**0.9254**	**0.9120**
	-5	0.8022	0.9234	0.9061	**0.8418**	**0.9361**	**0.9232**
	0	0.8383	0.9350	0.9208	**0.8655**	**0.9422**	**0.9311**
	5	0.8673	0.9406	0.9310	**0.8789**	**0.9452**	**0.9363**
	10	0.8810	0.9425	0.9371	**0.8864**	**0.9467**	**0.9403**
leopard	-10	0.6922	0.8862	0.8703	**0.7548**	**0.9100**	**0.8720**
	-5	0.7100	0.8966	0.8779	**0.7693**	**0.9187**	**0.8786**
	0	0.7404	0.9102	**0.8887**	**0.7871**	**0.9267**	0.8849
	5	0.7806	0.9230	**0.8989**	**0.8072**	**0.9332**	0.8906
	10	0.8197	0.9318	**0.9056**	**0.8259**	**0.9375**	0.8958
network parameters		363186			647108		

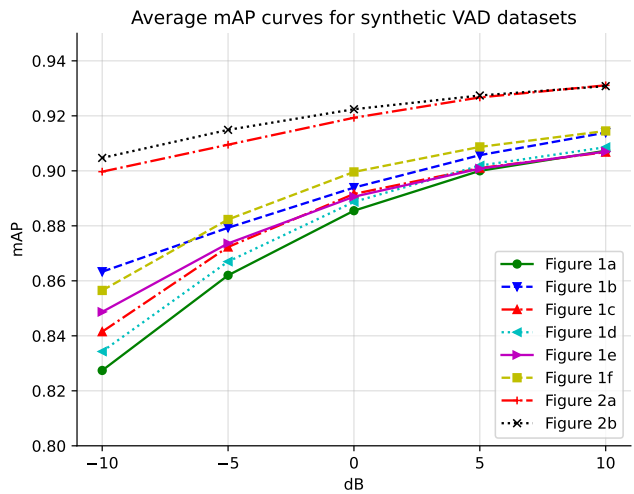

Figure 6: Average mAP curves of four noises (white, babble, f16, and leopard) under different SNRs for VAD systems tested on synthetic VAD datasets.

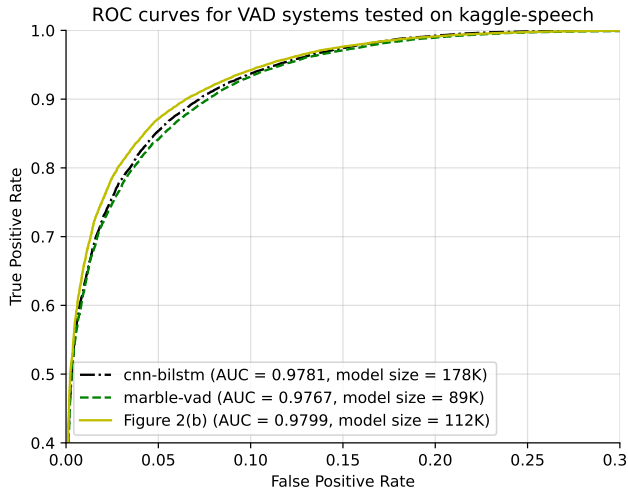

Figure 7: ROC curves for VADs tested on kaggle-speech.

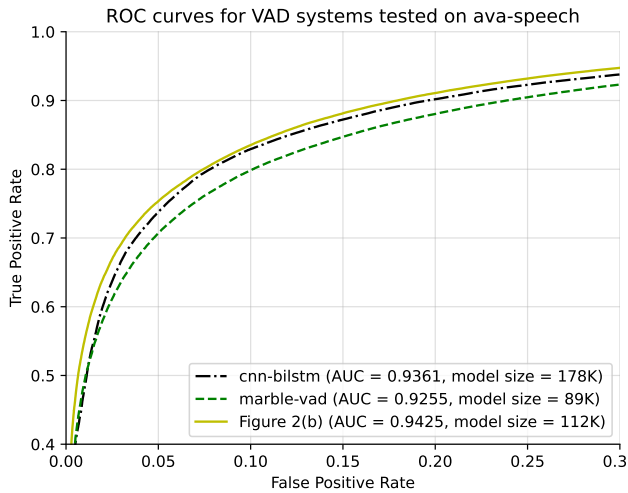

Figure 8: ROC curves for VADs tested on ava-speech.

respectively. The feature scales of the CNN-biLSTM VAD and the Figure 2b method are the same. Figure 8 shows the receiver operating characteristic (ROC) curves for all systems tested on ava-speech. We also see that the

Figure 2b systems we propose outperform the CNN-biLSTM VAD and the MARBLENET VAD across all operating points. When threshold=0.5, the CNN-biLSTM VAD, the MARBLENET VAD, and the Figure 2b method obtain accuracy rates of 86.03%, 84.21% and 86.65%, respectively.

5 Conclusions

In this paper, we propose a heterogeneous CRNN network framework with attention mechanism and feature aggregation for voice activity detection, which introduces hybrid attention block (ha-block) or multi-head self-attention (mhsa) block into the rear of LSTM layer in the network architecture. The attention module can improve representation power of CRNN networks. In the ha-block, we apply attention-based feature refinement with two distinctive modules, channel and spatial, and achieve considerable performance improvement while keeping the overhead small. In the mhsa-block, it is applied on the frame-wise embedding sequence to gain contextual information. Finally, a novel feature aggregation method has been implemented based on scalar weights or vector-based attention weights. It can extract more robust embeddings from the CRNN network structure.

We evaluate the proposed method with the baseline CRNN system. Experiments conducted on synthetic VAD datasets, kaggle VAD datasets and AVA-speech datasets. The mean Average Precision (mAP) and receiver operating characteristic (ROC) curves demonstrate the effectiveness of the proposed method. The algorithm with mhsa modules and feature aggregation based on vector-based attention weights achieves the best mAP, on average. In the future, we plan to incorporate the new attention mechanism and feature aggregation methods into more CRNN-based network architectures and evaluate the effect of different configurations.

References

[1] H. Bu, J. Du, X. Na, B. Wu, and H. Zheng, "Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline", in *2017 20th conference of the oriental chapter of the international coordinating committee on speech databases and speech I/O systems and assessment (O-COCOSDA)*, IEEE, 2017, 1–5.

[2] S. Chaudhuri, J. Roth, D. P. Ellis, A. Gallagher, L. Kaver, R. Marvin, C. Pantofaru, N. Reale, L. G. Reid, K. Wilson, *et al.*, "Ava-speech: A densely labeled dataset of speech activity in movies", *arXiv preprint arXiv:1808.00606*.

[3] Q. Chen, Z.-H. Ling, and X. Zhu, "Enhancing sentence embedding with generalized pooling", *arXiv preprint arXiv:1806.09828*.

[4] H. Dinkel, S. Wang, X. Xu, M. Wu, and K. Yu, "Voice activity detection in the wild: A data-driven approach using teacher-student training", *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29, 1542–55.

[5] T. Drugman, Y. Stylianou, Y. Kida, and M. Akamine, "Voice activity detection: Merging source and filter-based information", *IEEE Signal Processing Letters*, 23(2), 252–6.

[6] 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 *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, 2013, 483–7.

[7] G. Gelly and J.-L. Gauvain, "Optimization of RNN-based speech activity detection", *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(3), 646–56.

[8] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, 7132–41.

[9] 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, 7378–82.

[10] F. Jia, S. Majumdar, and B. Ginsburg, "Marblenet: Deep 1d time-channel separable convolutional neural network for voice activity detection", in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2021, 6818–22.

[11] Y. R. Jo, Y. K. Moon, W. I. Cho, and G. S. Jo, "Self-attentive vad: Context-aware detection of voice from noise", in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2021, 6808–12.

[12] J. Kim and M. Hahn, "Voice activity detection using an adaptive context attention model", *IEEE Signal Processing Letters*, 25(8), 1181–5.

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

[14] lazyrac00n, 2020, https://www.kaggle.com/datasets/lazyrac00n/speech-activity-detection-datasets.

[15] S. Meier and W. Kellermann, "Artificial Neural Network-Based Feature Combination for Spatial Voice Activity Detection.", in *INTERSPEECH*, 2016, 2987–91.

[16] Y. Obuchi, "Framewise speech-nonspeech classification by neural networks for voice activity detection with statistical noise suppression", in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2016, 5715–9.

[17] J. Ramirez, J. M. Górriz, and J. C. Segura, "Voice activity detection. fundamentals and speech recognition system robustness", *Robust speech recognition and understanding*, 6(9), 1–22.

[18] P. Renevey and A. Drygajlo, "Entropy based voice activity detection in very noisy conditions.", in *INTERSPEECH*, 2001, 1887–90.

[19] A. Sofer and S. E. Chazan, "CNN self-attention voice activity detector", *arXiv preprint arXiv:2203.02944*.

[20] J. Sohn, N. S. Kim, and W. Sung, "A statistical model-based voice activity detection", *IEEE signal processing letters*, 6(1), 1–3.

[21] Y.-W. Tan, W.-J. Liu, W. Jiang, and H. Zheng, "Hybrid svm/hmm architectures for statistical model-based voice activity detection", in *2014 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2014, 2875–8.

[22] S. Thomas, S. Ganapathy, G. Saon, and H. Soltau, "Analyzing convolutional neural networks for speech activity detection in mismatched acoustic conditions", in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2014, 2519–23.

[23] A. Varga and H. J. 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 communication*, 12(3), 247–51.

[24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need", *Advances in neural information processing systems*, 30.

[25] N. Wilkinson and T. Niesler, "A hybrid CNN-BiLSTM voice activity detector", in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2021, 6803–7.

[26] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module", in *Proceedings of the European conference on computer vision (ECCV)*, 2018, 3–19.

[27] Y. Wu, C. Guo, H. Gao, X. Hou, and J. Xu, "Vector-Based Attentive Pooling for Text-Independent Speaker Verification.", in *Interspeech*, 2020, 936–40.

[28] R. Zazo Candil, T. N. Sainath, G. Simko, C. Parada, *et al.*, "Feature learning with raw-waveform CLDNNs for voice activity detection".

[29] 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*, 24(2), 252–64.

[30] M. Zhu, "Recall, precision and average precision. Department of Statistics and Actuarial Science, University of Waterloo", in *Waterloo Working paper, Tech. Rep.* 2004.