

Received April 6, 2020, accepted April 21, 2020, date of publication April 29, 2020, date of current version May 14, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2990974

# EdgeRNN: A Compact Speech Recognition Network With Spatio-Temporal Features for Edge Computing

SHUNZHI YANG<sup>1</sup>, ZHENG GONG<sup>1</sup>, KAI YE<sup>1</sup>, YUNGEN WEI<sup>1</sup>,  
ZHENHUA HUANG<sup>1</sup>, (Senior Member, IEEE),  
AND ZHENG HUANG<sup>2</sup>

<sup>1</sup>School of Computer Science, South China Normal University, Guangzhou 510631, China

<sup>2</sup>School of Information Security Engineering, Shanghai Jiaotong University, Shanghai 200240, China

Corresponding author: Zheng Gong (cis.gong@gmail.com)


This work was supported in part by the National Natural Sciences Foundation of China under Grant 61572028, in part by the National Cryptography Development Fund under Grant MMJJ20180206, and in part by the Project of Science and Technology of Guangzhou under Grant 201802010044.

**ABSTRACT** Driven by the vision of Internet of Things, some research efforts have already focused on designing a network of efficient speech recognition for the development of edge computing. Other researches (such as tpool2) do not make full use of spatial and temporal information in the acoustic features of speech. In this paper, we propose a compact speech recognition network with spatio-temporal features for edge computing, named EdgeRNN. Alternatively, EdgeRNN uses 1-Dimensional Convolutional Neural Network (1-D CNN) to process the overall spatial information of each frequency domain of the acoustic features. A Recurrent Neural Network (RNN) is used to process the temporal information of each frequency domain of the acoustic features. In addition, we propose a simplified attention mechanism to enhance the portion of the network that contributes to the final identification. The overall performance of EdgeRNN has been verified on speech emotion and keywords recognition. The IEMOCAP dataset is used in speech emotion recognition, and the unweighted average recall (UAR) reaches 63.98%. Speech keywords recognition uses Google's Speech Commands Datasets V1 with a weighted average recall (WAR) of 96.82%. Compared with the experimental results of the related efficient networks on Raspberry Pi 3B+, the accuracies of EdgeRNN have been improved on both of speech emotion and keywords recognition.

**INDEX TERMS** RNN, speech emotion recognition, speech keywords recognition, edge computing.

## I. INTRODUCTION

According to the IHS Markit perspective [1], the number of Internet of Things (IoT) devices is expected to reach 125 billion by 2030. Those IoT has attracted lots of attention in the industry and academia because they can be widely used in many applications [2]. Because of their constrained resources [3], those micro-instruments are commonly named as edge computing devices. This large-scale edge computing devices will revolutionize the processing center of data from large-scale cloud centers to a wide range of terminal devices [4]. Processing data on terminal devices near the edge of the network is also called edge computing.

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Maaz Rehan .

Edge computing can address issues such as privacy, bandwidth cost, data processing costs, and scalability [5].

However, processing tasks directly on edge computing devices requires real-time and high accuracy. Further research is required because this area is still a challenging work. Customer satisfaction can be improved by offloading compute-intensive applications to the cloud. However, this will have many problems mentioned above. In general, increasing the complexity of the network in the deep learning method can improve the performance of the network [6]. Under the edge computing device, this expanded network structure may become a bottleneck. Therefore, a key issue at present is how to reduce the complexity of the deep network without significantly reducing the performance. Specifically, it is to reduce the amount of computations and parameters

of the model. In the study of Cheng *et al.* [7], the network showed some redundant weights. This research provides a theoretical basis for designing a deep neural network (DNN) for edge computing devices.

DNNs have achieved remarkable performance in computer vision, natural language processing, and speech recognition. However, in the field of DNNs for edge computing devices, only computer vision has achieved rapid development. This phenomenon is mainly due to two reasons. On the one hand, the field of natural language processing and speech recognition are mainly dealing with time-series issues. The time-series issues basically require the use of recurrent neural networks (RNN). However, RNN is computationally intensive and storage costly. Compared with a typical Convolutional Neural Network (CNN) cell [8], a RNN neuron requires 8 times of weights and multiply-accumulate operations. On the other hand, advances in computer vision have benefited from the discovery of group convolutions [9]. But the computation of RNN relies heavily on historical records. Therefore RNN is an atomic entity which cannot be grouped.

As a result, many proposals use 2-Dimensional (2-D) CNN for speech recognition. For example, Sainath and Parada [10] used 2-D CNN and DNN to build a speech keywords recognition network. However, 2-D CNN deals with local spatial information. In the spatial information of the processing sequence problem, 1-D CNN is better than 2-D CNN. In the time information of the processing sequence problem, RNN is also more efficient than 2-D CNN. 1-D CNN can extract spatial information over the entire time series features. The hidden layer in RNN retains the information of the previous time step to predict the value of the current time step. This technique is helpful to deal with different speech speed and time dependence problems in speech recognition [11].

Consequently, a combination of 1-D CNN and RNN is required to design a speech recognition network model for edge computing devices. The actual running speed, accuracy and model size in the network for edge computing are the most important indicators. The computations of a model can affect its processing speed, whilst the parameters of a model can enlarge its storage size. Hence, in the process of designing a network model for edge computing, it is necessary to consider the amount of computations and parameters. In [12], Krizhevsky have proposed two observations that we recall as follows.

- The convolutional layer accounts for approximately 90-95% of the computation.
- The fully connected layer occupies 95% of the parameter.

These two observations provide a statistical basis for designing a networks for edge computing devices. In language analysis, utterance level denotes that a single utterance, not a single word, or multiple utterances. In addition, not all frames in speech recognition have the same contribution to the final utterance level representation. For example, the short silent period in speech usually has little relevance with emotion [13]. The attention mechanism can focus on the part

that contributes to the final recognition while solving the dependency problem. Due to its good results, the attention mechanisms have been widely used in speed recognition. For example, Sun and Wu [14] combined attention mechanisms with sparse-autoencoder for speech emotion recognition. Therefore, a network model for edge computing can obtain better network performance by adding attention mechanisms. However, many attention mechanisms have intensive parameters, such as the multi-head attention mechanism [15]. Thus the number of parameters should be considered for the performance of attention mechanism.

To solve the performance and accuracy problems of speech recognition on edge computing devices, we propose a compact RNN which is named EdgeRNN. EdgeRNN consists of 1-D CNN, RNN and attention mechanism, which is a very common network structure for speech recognition. To our best knowledge, it is the first to be used in speech recognition tasks for edge computing devices. This is mainly because of the computations and parameters of 1-D CNN, RNN and attention mechanism. EdgeRNN focuses on solving the problems of computation and parameter of RNN, fully connected layers, and attention mechanisms.

The performance of EdgeRNN has been experimented on speech keywords and emotion recognition tasks. EdgeRNN has been successfully run on the Raspberry Pi 3B+<sup>1</sup>. To facilitate subsequent research, the complete project implementation can be obtained by its Github repository<sup>2</sup>. In general, the contributions of this paper are listed as follows:

- 1) The accuracies of the EdgeRNN model in both speech keywords and emotion recognition are better than existing efficient networks.
- 2) The EdgeRNN model runs on the Raspberry Pi 3B+ can recognize and process 2 voices faster than the time taken to collect the speech. This performance meets the practical requirements of speech recognition for edge computing.

The remainder of this paper is organized as follows. Section II introduces the related work of speech recognition. Section III describes the EdgeRNN model. Section IV presents the performance comparison and experimental analysis of EdgeRNN and other related networks. Section V concludes the paper.

## II. RELATED WORK

Speech emotion recognition is a branch of the field of speech recognition that aims to use speech to understand user emotions and opinions. Speech keywords recognition is also a branch of the field of speech recognition, aiming to find all corresponding keywords from the speech data. These two branches are popular research topics in speech recognition. Currently, speech recognition using deep learning methods can be classified into the following two types.

<sup>1</sup><https://www.raspberrypi.org/>

<sup>2</sup><https://github.com/yangshunzhi1994/EdgeRNN>

The first method is to use deep learning to directly model the original speech. For example, Zeghidour *et al.* [16] replaced the artificially generated acoustic features with trainable filterbanks. Latif *et al.* [17] used multi-convolution kernels and multi-step parallel convolution layers to extract features directly from audio. However, there are two issues here. The first problem is that the number of parameters becomes larger. For example, the acoustic feature is obtained by extracting 1-second audio at a sampling frequency of 16KHz. There are 4864 parameters with 152 dimensions, each dimension is 32 in length. The number of parameters extracted by acoustic features is 30.4% of the model proposed by Latif *et al.* The second problem is that the accuracy is low. For example, the current highest accuracy from the original speech directly modeled on the IEMOCAP dataset [18] is only 60.23% [17].

The second method is deep learning modeling after acoustic feature extraction, which is the most widely studied method. The most widely used acoustic features are Mel-frequency cepstral coefficients (MFCC) and Mel spectrogram. Certainly, extracting more acoustic features is not necessary. Because the computations will be costly and not all of those features are valid for identification [19].

The above analysis shows that design a speech recognition network model for edge computing should use acoustic features. In addition, there are two ways to obtain a deep learning network for edge computing. The first way is to accelerate the network from the perspective of algorithm and hardware through pruning, quantification, etc. For instance, BCRNN [20] uses a binary neural network to compress the speech recognition network model.

The second way is to directly design an efficient deep learning network. For instance, tpool2 [21] uses a total of 5 layers of CNN network for speech keywords recognition. SANAS [22] uses the Neural Architecture Search (NAS) method to dynamically adjust the architecture of the neural network for real-time speech keywords recognition. res15 [23] uses extended convolutions of residual structure for speech keywords recognition. L-CRNN [24] performs speech keywords recognition by reducing the number of parameters between the RNN and the fully connected layer. DS-CNN [25] uses depthwise separable convolution for speech keywords recognition. Full-DCNN [26] uses depthwise separable convolution for speech keywords recognition on specific hardware platforms (Movidius' Myriad 2).

Furthermore, many methods of running speech recognition on edge computing devices use traditional machine learning methods. For example, both Walid *et al.* [27] and Wiem and Lachiri [28] use support vector machines (SVM) classifiers for speech recognition systems on Raspberry Pi. Mnassri *et al.* [29] use the discrete wavelet transform (DWT) SVM method for speech keywords recognition.

### III. DESIGN OF EdgeRNN

The EdgeRNN model is divided into the following parts:

- 1) Acoustic feature extraction layer.

- 2) Spatial information extraction layer.
- 3) Feature pooling layer.
- 4) Time information extraction layer.
- 5) Self-attention mechanism and classification layer.

The above layers are illustrated separately.

#### A. ACOUSTIC FEATURE EXTRACTION LAYER

First, acoustic feature extraction of the original speech is required. In EdgeRNN, several experiments were carried out from the two aspects of accuracy and speed. Finally, 128-dimensional (128-D) Mel spectrogram, 12-D delta, and 12-D double-delta features were selected. The delta features are obtained by the first-order difference of the MFCC features. The double-delta features are obtained by the second-order difference of the MFCC features. The extraction of double-delta features needs to go through the following processes in order:

- 1) Obtaining the Mel spectrogram features.
- 2) Taking the logarithm of the Mel spectrogram features for each dimension.
- 3) Obtaining the MFCC features by discrete cosine transform (DCT).
- 4) Obtaining the delta features based on the MFCC features.
- 5) Obtaining double-delta features based on delta features.

Because the existing acoustic feature extraction libraries (such as the librosa [30] library) have encapsulated the features mentioned above. Thus, the extraction of acoustic features needs to avoid double computations. If the delta features of librosa are used directly, the Mel spectrogram features have been computed. However, the features of the Mel spectrogram are also an intermediate result required by the experiment. Accordingly, the EdgeRNN model avoids the repeated computation of features by gradually acquiring acoustic features. For details, see the implementation code for data processing in the EdgeRNN model.

EdgeRNN uses the librosa [30] library to extract the 152-D acoustic features, which include 128-D Mel spectrogram, 12-D delta and 12-D double-delta features. The neighboring features of the Mel spectrogram are highly correlated. There is overlapped between adjacent filter banks, which is beneficial for CNN modeling. Moreover, the Mel spectrogram has a higher dimension than MFCC, which retains more information. The delta features represent the relationship of adjacent frames, whilst the double-delta features depict the relationship among three adjacent frames. They are all good at expressing the dynamics of speech on the MFCC. Therefore, the delta and double-delta features used in EdgeRNN are extracted after MFCC.

Take the four emotions in IEMOCAP dataset [18] as examples. The extracted acoustic features are shown in Figure 1. The horizontal direction represents the acoustic features of a certain dimension, such as the pitch features of the speech. The vertical direction represents the acoustic features of a certain moment. As can be seen from Figure 1, speech

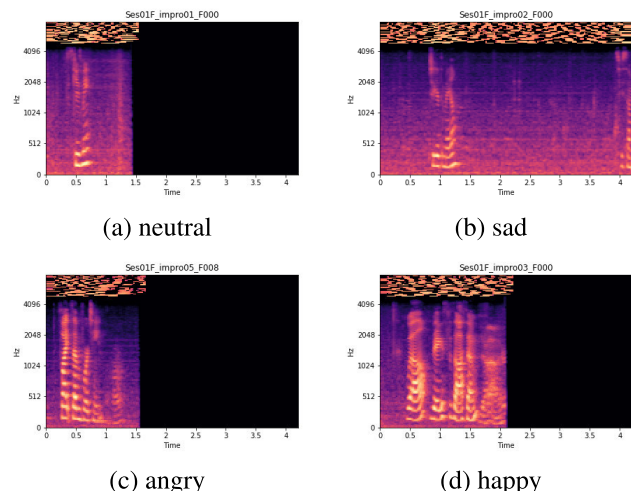


FIGURE 1. Acoustic features (Black blocks indicate that they are filled with “0”).

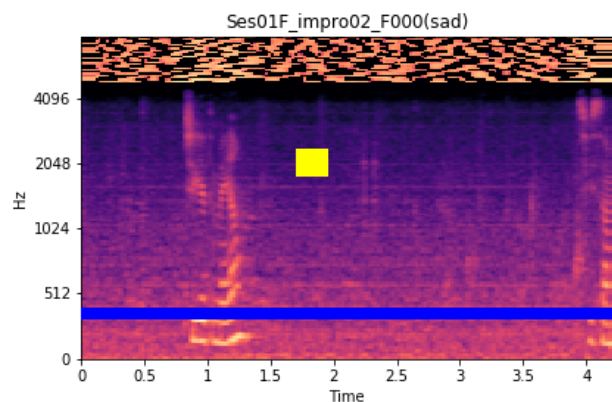


FIGURE 2. 2-D convolution (yellow block representation) and 1-D convolution (blue bar representation).

recognition has both time and spatial domain information. The time domain is the process of a voice signal changing with time, and the spatial domain is a feature of a voice signal. In the acoustic features of speech, the time domain information refers to the time-series features in the frequency-domain direction. The spatial domain information refers to the spatial features in the frequency-domain direction. The frequency-domain direction is shown in the blue bar representation in Figure 2.

Therefore, in the speech recognition task, how to integrate the information in time and spatial domain is critical issues. This requires a combination of CNN and RNN to handle speech recognition. The extracted acoustic features are low-level, while CNN is suited for the extraction of low-level features.

**B. SPATIAL INFORMATION EXTRACTION LAYER**

After extracting the acoustic features, these low-level features are modeled by using a deep learning approach. In our previous work [31] elaborated on the importance of learned group

convolution [32] and DenseNet [33] for designing a network model for edge computing. Although group convolution can reduce the multiply computations, it adds extra memory accesses [34]. This means that group convolution depends on the specific use environment and network model. To verify the effect of group convolution on speech recognition tasks, EdgeRNN-G was designed. The only difference between EdgeRNN and EdgeRNN-G is whether the convolutional layer uses learned group convolution. EdgeRNN-G uses a learned group convolution, and EdgeRNN uses a normal convolution. The learned group convolution reduces convolution operation in multiples. It also enables the network to automatically group in the learning process to solve the information circulation problem. The principle of DenseNet’s feature reuse can be expressed by the following formula:

$$X_L = H_L([X_0, X_1, \dots, X_{L-1}]) \tag{1}$$

$X_L$  represents the output of the  $L$ th layer, and  $H_L(\cdot)$  is a composite function representing the combined operation. DenseNet is used to improve the performance through feature reuse. Each layer in DenseNet generates only a few features. But after the connection, the last layer of the network can obtain the feature maps of all previous layers.

The design of the EdgeRNN model still uses the EdgeBlock designed by EdgeCNN [31] and the introduced network model principles. However, the previous convolution layer process is a computer vision task. 2-D convolution can be used for computer vision tasks. The different ways of processing the 2-D and 1-D convolutional layers are shown in Figure 2. Obviously, the 2-D convolution deals with the local spatial information of the acoustic features. However, 1-D convolution deals with the overall spatial information of the frequency-domain direction of the acoustic features. Accordingly, the 1-D convolutional layer is more suitable for modeling acoustic features. EdgeRNN-G’s EdgeBlock changes the previous learned group convolution from 2-D to 1-D for speech recognition. Similarly, EdgeRNN’s EdgeBlock uses a normal 1-D convolution.

The difference between EdgeBlock used by EdgeRNN and EdgeCNN are that in addition to the 1-D CNN used for the convolution layer, the activation layer used is also different.

As shown in Figure 3, the main activation functions used in speech recognition are Tanh, ReLU and its variants. In computer vision, the pixel value of a picture is between [0, 255]. ReLU can be used in computer vision because the gradient does not decay when the pixel value is greater than 0. However, there are negative values when converting speech into digital signals. If the activation function in EdgeRNN’s EdgeBlock uses ReLU, nearly half of the features will be lost. The ELU activation function does not distinguish much when it is negative. Our experiments prove that the performance  $PRReLU > ELU > Tanh > ReLU$  in the process of learning higher-level features. Therefore, both EdgeRNN-G and EdgeRNN’s EdgeBlock use the PRReLU activation function. The details of EdgeRNN-G’s EdgeBlock are present



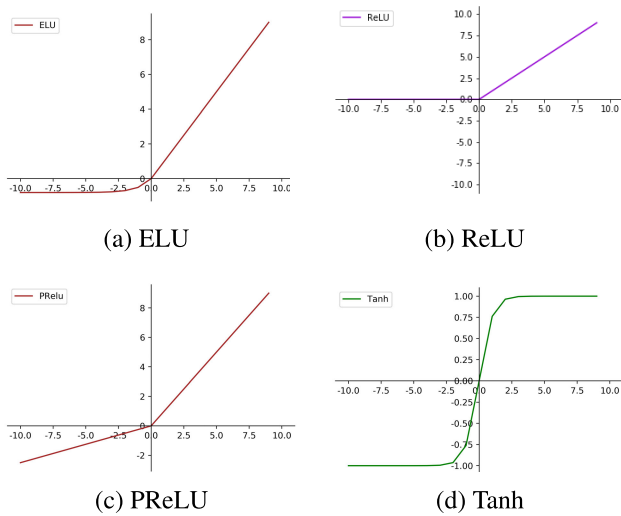


FIGURE 3. Four commonly used activation functions in speech recognition.

TABLE 1. Structure diagram of EdgeBlock in EdgeRNN-G (1D-LConv stands for 1-Dimensional learned group convolution, G stands for number of groups, C is Condensation Factor. The out\_channels represents the feature dimension of the output, and the growth\_rate represents the growth factor of the features).

EdgeRNN-G's EdgeBlock	Layer
Convolutional block 1	$3 \times 3$ 1D-LConv, G=4, C=4, out_channels= $4 \times$ growth_rate
	BatchNorm Layer
	Activation layer (PReLU)
Convolutional block 2	$3 \times 3$ 1D-LConv, G=8, C=8, out_channels= $1 \times$ growth_rate
	BatchNorm Layer

TABLE 2. Structure diagram of EdgeBlock in EdgeRNN (1D-Conv stands for 1-Dimensional normal convolution).

EdgeRNN's EdgeBlock	Layer
Convolutional block 1	$3 \times 3$ 1D-Conv, out_channels= $4 \times$ growth_rate
	BatchNorm Layer
	Activation layer (PReLU)
Convolutional block 2	$3 \times 3$ 1D-Conv, out_channels= $1 \times$ growth_rate
	BatchNorm Layer

in Table 1, and the details of EdgeRNN's EdgeBlock are shown in Table 2.

For acoustic feature maps, EdgeRNN uses the principle of DenseNet to continuously extract higher-level features. Different from the previous model [31], EdgeRNN retains the original acoustic feature map. Meanwhile, EdgeRNN continuously extracts higher-level features using 1-D CNN based on all previous layers. The original acoustic and the higher-level features are incorporated into the RNN layer to extract time-series information.

C. FEATURE POOLING LAYER

The pooling layer is one of the common components in deep learning networks. It has four main functions: feature

invariance, feature dimensionality reduction, noise reduction, and prevent overfitting. Feature invariance means that the model will pay more attention to the existence of features rather than the specific location. Feature dimension reduction refers to reducing the amount of computations and parameters. The noise reduction means reducing local noise. Preventing overfitting is due to the number of parameters reduction.

The pooling layer of EdgeRNN reduces the dimensionality of spatial features, thereby reducing the computations and parameters of the model. This is because the computations after the pooling layer are intensive, such as the RNN layer. Pooling is necessary before these layers. Unlike computer vision tasks, EdgeRNN uses a 1-D max pooling layer. The schematic diagram of the 1-D max pooling layer is depicted in Figure 4.

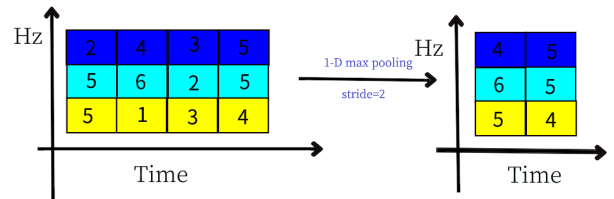


FIGURE 4. The schematic diagram of the 1-D max pooling layer.

D. TIME INFORMATION EXTRACTION LAYER

Speech recognition not only has time domain information, but also spatial domain information. It is inefficient to identify speech separately from the time or spatial domain dimensions. The strength of CNN lies in spatial information, whilst the strength of RNN lies in time information. Consequently, many proposals use a combination of CNN and RNN for speech recognition networks. For instance, Zhao et al. [35] take advantage of CNN and Long Short Term Memory (LSTM) for speech emotion recognition. Wang [36] combined CNN and Gated Recurrent Unit (GRU) for hate speech recognition. However, the RNNs used in these models are variants of RNN, such as LSTM [37] and GRU [38]. RNN uses one gate, GRU uses three gates, and LSTM uses four gates. This means that the computational complexity of the GRU is three times that of the RNN. The computational complexity of the LSTM is four times that of the RNN. It is known that RNN has the problem of gradient explosion or gradient disappearance in long sequences. Therefore, LSTM and GRU are introduced to solve the time dependency problem in long sequences. However, many tasks in speech emotion and keywords recognitions are short/medium-term dependent (i.e., short-time pronunciation). RNN is applicable in the short/medium-term dependency issues. Moreover, attention mechanisms can be used to enable RNNs to learn short/medium-term dependencies in short/medium-term sequences.

**TABLE 3. Architecture of EdgeRNN-G (Feature Transpose refers to the process of converting the two-dimensional tensor (344,90) to (90,344) to meet the RNN input format requirements. 152, 344, and 4 are the dimensions of the feature. 181 and 90 represent the length of features. 1 indicates a single-layer RNN).**

Layer	Operator	Output
Acoustic Feature	128-D Mel spectrogram + 12-D delta + 12-D double-delta	(152,181)
EdgeBlock	EdgeRNN-G's EdgeBlock $\times$ 12	(344,181)
MaxPool	1-D MaxPool(kernel_size=2, stride=2)	(344,90)
Feature Transpose	(Transpose the output features of MaxPool.)	(90,344)
RNN	RNN(344,344,1)	(90,344)
Self Attention Layer	-	(344)
Classification Layer	fully-connected, softmax	(4)

### E. SELF-ATTENTION MECHANISM LAYER AND CLASSIFICATION LAYER

In order to make up for the shortcomings of the RNN and to enhance the network part that helps the final recognition, an attention mechanism should be added to EdgeRNN. However, most of the attention mechanisms use the fully connected layer to remap features. For example, the multi-head attention mechanism [15] uses multiple fully connected layers to capture different aspects of the input features. Capturing multiple aspects of information helps improve the accuracy of predictions [39]. But the fully connected layer contains a large number of parameters and will increase the computational cost. In order to meet the needs of edge computing, this paper proposes an efficient self-attention mechanism with negligible parameters and computations.

Assuming that  $x_i$  is the contextual feature of the current time series computed by the RNN,  $\alpha_i$  is scored by the following formula:

$$\alpha_i = \frac{1}{1 + e^{-x_i}} \quad (2)$$

In fact, the  $\alpha_i$  calculation method is the sigmoid function. Its value is between [0, 1].  $\alpha_i$  can be interpreted as the contribution score of the frame to the final utterance level representation of the speech. The attention score  $\alpha_i$  is used for the weighted average to obtain the representation of the utterance level C:

$$C = \frac{\sum_i \alpha_i \cdot x_i}{\sum_i \alpha_i} \quad (3)$$

$\sum_i \alpha_i \cdot x_i$  is the weighted sum of features at each location. After the results obtained by the utterance level are activated, they are passed to the fully connected layer to summarize the final results. The final result of the summary is passed to the softmax layer of the network to obtain the posterior probability of each speech category.

### F. THE OVERALL STRUCTURE OF EdgeRNN

EdgeRNN and EdgeRNN-G still use the feature reusing principle introduced by DenseNet [33]. In other words, the

**TABLE 4. Architecture of EdgeRNN (Same parameters as EdgeRNN-G).**

Layer	Operator	Output
Acoustic Feature	128-D Mel spectrogram + 12-D delta + 12-D double-delta	(152,181)
EdgeBlock	EdgeRNN's EdgeBlock $\times$ 12	(344,181)
MaxPool	1-D MaxPool(kernel_size=2, stride=2)	(344,90)
Feature Transpose	(Transpose the output features of MaxPool.)	(90,344)
RNN	RNN(344,344,1)	(90,344)
Self Attention Layer	-	(344)
Classification Layer	fully-connected, softmax	(4)

features obtained at each step will enter the final classification layer. The model structures of EdgeRNN-G and EdgeRNN are shown in Table 3 and Table 4, respectively. The model structure is present in a 4-category IEMOCAP dataset.

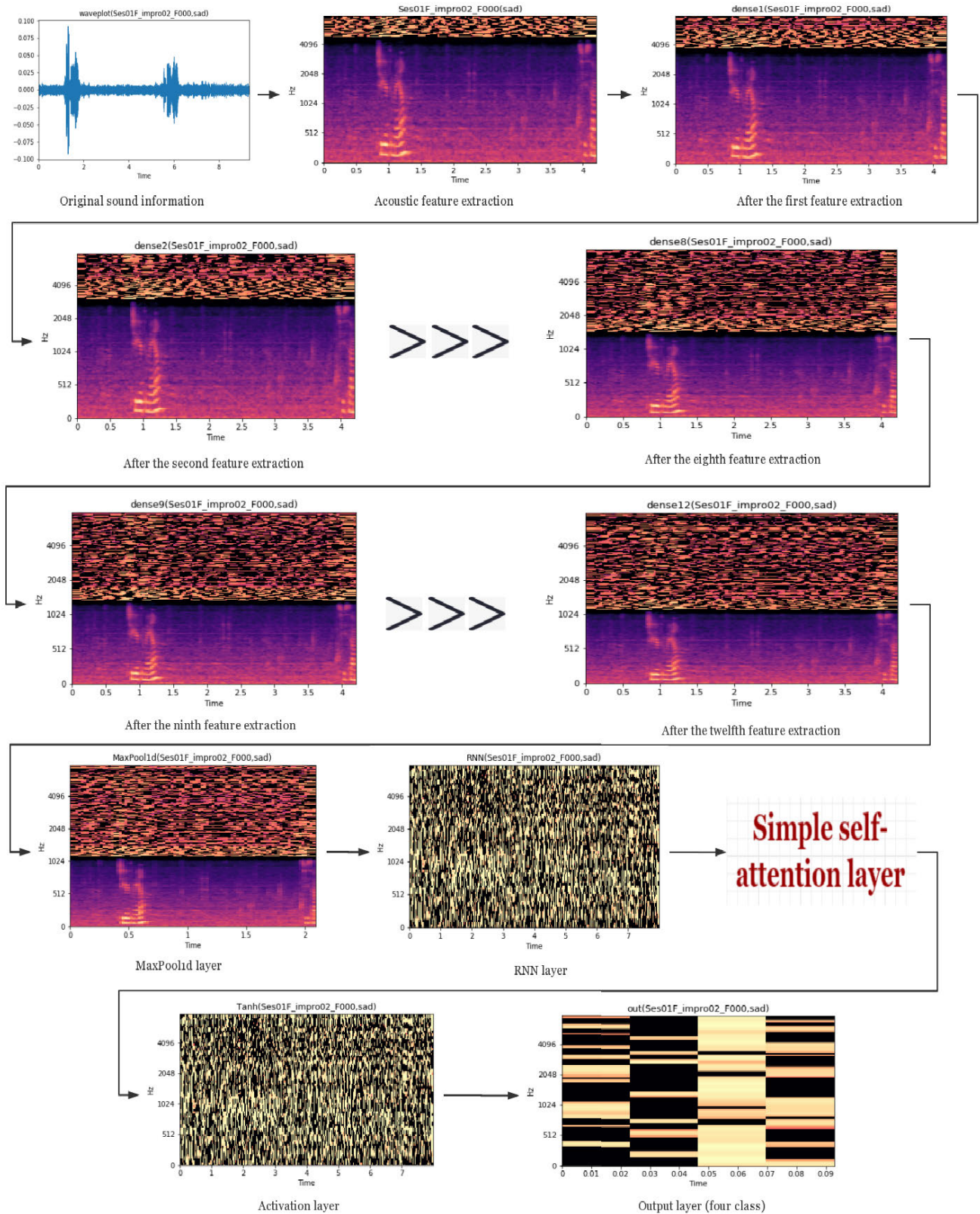
The EdgeRNN model retains the acoustic feature maps generated by the audio, and continuously extracts time and spatial features based on all previous layers. The visualization of EdgeRNN is shown in Figure 5. The Ses01F\_impro02\_F000 sample in the IEMOCAP dataset was used to visually display the model structure of EdgeRNN. Because EdgeRNN and EdgeRNN-G are approximately the same, there is no need to display the structure of EdgeRNN-G visualization.

Figure 5 illustrates the principle of feature reuse. EdgeRNN adds 16-D higher-level spatial features from EdgeBlock at a time. EdgeRNN uses EdgeBlock 12 times, which adds 192-D higher-level spatial features. It is worth noting that new feature is extracted based on all previous layers. For example, the new 16-dimensional higher-level features of dense9 in Figure 5 are extracted based on the entire feature map of dense8. Due to the limited space of a page, the visualizations of the third to seventh higher-level spatial feature extraction layers are omitted in Figure 5. Meanwhile, the visualization diagrams of the tenth and eleventh higher-level spatial feature extraction layers are also omitted. However, it does not affect viewing the visual structure of EdgeRNN. Instead, it better shows the visual structure of EdgeRNN. Because the added features are more obvious, such as the features added from the second to the eighth are obvious.

After the model is designed, the loss function and the setting of an optimizer need to be chosen. The experiments used Softmax loss function and SGD optimization technology. The SGD optimizer minimizes the Softmax loss function with a learning rate of 1e-2 and a weight decay of 5e-4. Meanwhile, after 80 epochs, the learning rate dropped by 10% after every 5 epochs.

### IV. EXPERIMENTAL PERFORMANCE OF EdgeRNN

The performances of EdgeRNN are demonstrated on two subtasks: speech emotion and keywords recognition. A widely used dataset for speech emotion recognition is Interactive Emotional Dyadic Motion Capture Database (IEMOCAP) [18]. IEMOCAP dataset was recorded by the ten



**FIGURE 5.** Visualization of EdgeRNN network structure (Each picture is the result of EdgeRNN’s latest concatenation. In EdgeRNN, the latter network has more features than the former. However, each picture is compressed to a uniform size).



actors of the Signal Analysis and Interpretation Laboratory at the University of Southern California (USC). It is generated during a binary interaction. The IEMOCAP dataset is a multi-modal emotional corpus of action, including audio, video and text data. The experiment uses only audio data. Consistent with most previous work, excitement was added to the happy category to achieve a more balanced label distribution. There are totally 4 emotion classes {happy, sad, angry, neutral}. The training and validation datasets are not specified in the IEMOCAP dataset. Moreover, its category distribution is not uniform. Therefore, EdgeRNN uses 80% of each category for training and 20% for testing. The total data used was 5531, of which 4423 was used for training and 1108 was used for testing.

The length of the IEMOCAP dataset is inconsistent. First, this paper arranges the IEMOCAP dataset from short to long. Second, selects 75% of the positions (5.78 seconds) as the unified audio length. For audio less than 5.78s, fill it with "0" to 5.78s (Figure 1); those longer than that will be trimmed to 5.78s.

The most widely used dataset for speech keywords recognition is Google's Speech Commands datasets V1 [40]. The dataset consists of 65,000 1-second long audio clips consisting of 30 keywords and 1 noise. Each clip containing only one keyword. EdgeRNN focuses on identifying the following 11 keywords: {yes, no, up, down, left, right, on, off, stop, go, unknown}. The {unknown} includes the remaining 20 keywords and 1 noise in the dataset. The Speech Commands datasets V1 is divided into training, validation and test datasets in a ratio of 80:10:10. EdgeRNN adds validation to test datasets to verify the performance of the network model. That is to say, 80% of the Speech Commands Datasets V1 is used for training and 20% is used for testing.

**TABLE 5. EdgeRNN and EdgeRNN-G performance on IEMOCAP database for speech emotion recognition.**

Network	UAR(%)	Model size (MB)	Millions Params	Millions Ops
BCRNN [20]	61.90	4.34	-	-
EdgeRNN-G	63.91	5.47	0.83	16.77
EdgeRNN	63.98	3.24	0.83	152.52

Few work in speech emotion recognition tasks has focused on the design of efficient network models. Only BCRNN [20] was found in speech emotion recognition. The method used by BCRNN has been introduced in the related work of this paper. BCRNN shows the accuracy and size of the model on the IEMOCAP dataset. The performance comparison of EdgeRNN, EdgeRNN-G and BCRNN are shown in Table 5. The accuracy used in Table 5 is the unweighted average recall rate (UAR). The UAR is defined as the accuracy per class averaged over all classes so that the accuracy for each class has the same importance [41]. In short, UAR is the average accuracy of the class. UAR is a widely used indicator in speech emotion recognition due to class imbalanced of the dataset. The indicators for evaluating the accuracy of speech emotion recognition also have weighted average recall (WAR). The WAR is the ratio of the total number of correctly

predicted and test audio. In short, WAR is the accuracy of the test dataset. It can be seen from Table 5 that the UAR and model size of EdgeRNN have been improved from the work of BCRNN. In experiments, EdgeRNN has a UAR of 63.98% and a WAR of 64.00% in speech emotion recognition.

Research on speech keywords recognition is more popular. The performance comparison of EdgeRNN, EdgeRNN-G and other network models on speech keywords recognition are shown in Table 6. The speech keywords recognition in Table 6 uses Google's Speech Commands Datasets V1 [40]. There are multiple network models in other works. In Table 6, only the most accurate models are listed. WAR is a widely used indicator in speech keywords recognition. The other methods used in Table 6 for speech keywords recognition works have been introduced in related work in this paper. It can be seen from Table 6 that the accuracies of the EdgeRNN and EdgeRNN-G models can be improved from other works. In experiments, EdgeRNN has a WAR of 96.82% and a UAR of 93.63% in speech keywords recognition.

**TABLE 6. EdgeRNN and EdgeRNN-G performance on Google's Speech Commands datasets V1 for speech keywords recognition.**

Network	WAR(%)	Millions Params	Millions Ops
SANAS [22]	80.70	-	37.70
Full-DCNN [26]	88.50	0.01	5.94
tpool2 [21]	91.97	1.09	103.00
DS-CNN [25]	95.40	0.49	56.90
res15 [23]	95.80	0.23	894.00
L-CRNN [24]	96.17	0.57	6.60
EdgeRNN-G	96.62	0.83	2.96
EdgeRNN	96.82	0.83	26.96

**TABLE 7. Time delay and power consumption for EdgeRNN and EdgeRNN-G running on Raspberry Pi 3B+ (Energy calculations and peak power exclude idle power draw).**

Network	Dataset	Total latency(s)	Time delay for acoustic features(s)	Energy (J)	Peak Power(W)
tpool2 [21]	Speech Commands [40]	0.20	-	0.384	2.21
EdgeRNN-G	Speech Commands [40]	0.68	0.10	1.496	2.82
EdgeRNN	Speech Commands [40]	0.59	0.10	1.234	2.70
EdgeRNN-G	IEMOCAP [18]	1.12	0.27	2.464	2.82
EdgeRNN	IEMOCAP [18]	1.01	0.27	2.113	2.70

The EdgeRNN and EdgeRNN-G models have been successfully run on Raspberry Pi 3B+. The performance of the running speed and power consumption compared with other networks are shown in Table 7. As with tpool2 [21], energy calculations and peak power values do not include the energy consumed by the Raspberry Pi 3B+ in its idle state. Two power meters (UT230C-II and kewode) are used for power consumption testing of EdgeRNN and EdgeRNN-G. The power of the Raspberry Pi 3B+ tested by the UT230C-II power meter in the idle state is 1.4W, but the kewode power meter is 1.754W. The power of the Raspberry Pi 3B+ in the idle state tested by the two power meters is different. However, the energy consumptions of the speech recognitions are the same in our power meters tests. Because the energy consumed by speech recognition equals the energy during speech recognition minus the energy in idle state. As shown by the speech keywords recognition task in Table 7, the speed of EdgeRNN is 1/3 of tpool2. In the meanwhile, EdgeRNN consumes three times as much energy as tpool2 for processing one voice. It implies that EdgeRNN and tpool2 consume



similar energy in the same time despite the different model complexity. As energy consumption equals power times time, the energy consumption is mainly reduced by increasing the speed of the model.

Total latency in Table 7 is the total time to process a voice, which includes speech acoustic feature extraction. The tpool2 [21] network uses the 2-D CNN structure. But the EdgeRNN uses a combined structure of 1-D CNN, RNN and attention mechanism. As can be seen from Table 7, EdgeRNN runs slower on the Raspberry Pi 3B+ than the full 2-D CNN network model. However, in speech emotion recognition, 5.78 seconds of speech can be processed on the Raspberry Pi 3B+ in 1.01 seconds. In speech keywords recognition, 1 second of speech can be processed on the Raspberry Pi 3B+ in 0.59 seconds. In practical applications, speech recognition by edge computing devices consists of the following two steps:

- The sensor collects speech from the surrounding environment.
- The edge computing device processes the speech collected by the sensor.

It should be noted that these two steps are parallel. Obviously, EdgeRNN takes less time to perform speech recognition than to collect speech. This means that EdgeRNN can meet the practical requirements of speech recognition for edge computing. In addition, EdgeRNN is 5.08% better than tpool2 [21] in terms of accuracy. Compared to tpool2, EdgeRNN is a more efficient model in terms of accuracy. Consequently, we construct a compact model with improved accuracy, while its processing speed reaches the actual requirements of edge computing devices.

The memory access speed of the mobile phone, Raspberry Pi 3B+ and its SD Card interface were tested in this paper. The write speed of the SD card interface of the Raspberry Pi 3B+ is 22.61 MB/s. The memory access speed of the Raspberry Pi 3B+ is 267 MB/s. The memory access speed of the ordinary mobile phone (honor8 lite) is 222.26 MB/s. Although the memory access speed of the Raspberry Pi 3B+ is faster than that of a mobile phone, the Raspberry Pi uses an SD card to store data. What determines the actual memory access speed of the Raspberry Pi is the write speed of the SD card interface. This implies that those embedded devices can access memory 10 times faster than edge computing devices. Therefore, those embedded devices can use models based on group convolution for its fast memory access speed. Experiments with EdgeRNN and EdgeRNN-G show that using group convolution on edge computing devices with slow memory access is not necessarily efficient. The use of group convolution depends on the specific network model and operating environment. In general, group convolution needs to consider the cost of memory access in resource-constrained edge computing devices.

The number of computations will definitely affect the speed of the network model. Therefore, the number of computations needs to be reduced as much as possible. It is worth noting that the number of computations cannot be used as an

indicator of the network for edge computing. The number of computations is only one of the indicators of the speed of the network. Other factors can also affect the speed of the network, such as the memory access speed of the network operating platform [34]. This means the factors that affect the speed include not only the number of computations, but also the speed of memory access and others. Only the sum of all factors (such as the number of computations and memory access costs) can determine the speed of the network. For example, the A network has a computation amount of 10M and other factors of 30M. The computation amount of B network is 20M and its other factors are 10M. B is better than A, not A is better than B. Even if the number of computations can affect the speed of the network, it cannot be directly used as an indicator of a compact network for edge computing. The processing speed on certain devices, for which will be the critical metrics of the network model towards edge computing environments.

As seen in Table 5, the group convolution not only reduces the speed but also increases the size of the model. Because the model not only needs to store the parameters of the network, but also the network structure. Obviously, the network structure using group convolution is relatively complicated. From the metrics of Table 5, we note that the parameter sizes of EdgeRNN and EdgeRNN-G are the same. Meanwhile, group convolution requires more storage space than normal convolution under the same conditions. In particular, it is not advantageous to use group convolution on an edge computing device with limited storage.

Besides, accuracy is the proportion of correct judgments in total observations, and it is also an important indicator. Therefore, it is also necessary to improve the accuracy of the model as much as possible when designing the model. Through the analysis, it can be concluded that accuracy, speed, and model size are the most important indicators for evaluating the performance of a network for edge computing.

As depicted in Table 6 and Table 7, both the computation amount of EdgeRNN and EdgeRNN-G are lower than tpool2. But their running time on Raspberry Pi 3B+ becomes the opposite. The main reason for this problem is the sequential processing method of RNN, which will inevitably lead to a reduction in speed. In addition to RNN, the low speed of EdgeRNN also has a 1-D CNN factor. 1-D convolution is a special case of 2-D convolution. It can also be seen from Figure 2 that in the same case, the convolution kernel of 1-D convolution is larger than that of 2-D convolution. This makes 1-D CNN more complicated than 2-D CNN. However, 1-D CNN and RNN can make full use of the time and spatial information in the acoustic features. This can improve the accuracy of network prediction. Therefore, it is necessary to combine 1-D CNN and RNN for speech recognition.

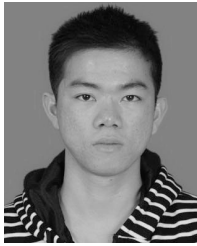
## V. CONCLUSION

In this paper, EdgeRNN has been proposed as a compact RNN model with spatio-temporal features for edge computing. Compared to the existing efficient network models,

EdgeRNN has better accuracy in the speech keywords and emotion recognition. Meanwhile, EdgeRNN can be practically implemented on the typical edge computing devices, such as Raspberry Pi 3B+. The downside is that EdgeRNN cannot actually run faster on the Raspberry Pi than the full 2-D CNN network model. In the future, its performance tuning will be a time-costly work for practical applications of EdgeRNN.

## REFERENCES

- [1] A. Alnoman, S. K. Sharma, W. Ejaz, and A. Anpalagan, "Emerging edge computing technologies for distributed IoT systems," *IEEE Netw.*, vol. 33, no. 6, pp. 140–147, Nov. 2019.
- [2] Z. Huang, X. Xu, J. Ni, H. Zhu, and C. Wang, "Multimodal representation learning for recommendation in Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10675–10685, Dec. 2019.
- [3] M. Chen, W. Li, G. Fortino, Y. Hao, L. Hu, and I. Humar, "A dynamic service migration mechanism in edge cognitive computing," *ACM Trans. Internet Technol.*, vol. 19, no. 2, pp. 1–15, Apr. 2019.
- [4] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge intelligence: Paving the last mile of artificial intelligence with edge computing," 2019, *arXiv:1905.10083*. [Online]. Available: <http://arxiv.org/abs/1905.10083>
- [5] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.
- [6] K. Zhen, A. Sivaraman, J. Sung, and M. Kim, "On psychoacoustically weighted cost functions towards resource-efficient deep neural networks for speech denoising," 2018, *arXiv:1801.09774*. [Online]. Available: <http://arxiv.org/abs/1801.09774>
- [7] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "A survey of model compression and acceleration for deep neural networks," 2017, *arXiv:1710.09282*. [Online]. Available: <http://arxiv.org/abs/1710.09282>
- [8] M. Hossein Samavatian, A. Bacha, L. Zhou, and R. Teodorescu, "RNNFast: An accelerator for recurrent neural networks using domain wall memory," 2018, *arXiv:1812.07609*. [Online]. Available: <http://arxiv.org/abs/1812.07609>
- [9] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*. [Online]. Available: <http://arxiv.org/abs/1704.04861>
- [10] T. N. Sainath and C. Parada, "Convolutional neural networks for small-footprint keyword spotting," in *Proc. 16th Annu. Conf. Int. Speech Commun. Assoc. (Interspeech)*, 2015, pp. 1478–1482. [Online]. Available: <https://dblp.uni-trier.de/rec/bibtex/conf/interspeech/SainathP15>
- [11] A. Shewalkar, D. Nyavanandi, and S. A. Ludwig, "Performance evaluation of deep neural networks applied to speech recognition: RNN, LSTM and GRU," *J. Artif. Intell. Soft Comput. Res.*, vol. 9, no. 4, pp. 235–245, Oct. 2019.
- [12] A. Krizhevsky, "One weird trick for parallelizing convolutional neural networks," 2014, *arXiv:1404.5997*. [Online]. Available: <http://arxiv.org/abs/1404.5997>
- [13] Z. Zhang, B. Wu, and B. Schuller, "Attention-augmented end-to-end multi-task learning for emotion prediction from speech," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 6705–6709.
- [14] T.-W. Sun and A.-Y.-A. Wu, "Sparse autoencoder with attention mechanism for speech emotion recognition," in *Proc. IEEE Int. Conf. Artif. Intell. Circuits Syst. (AICAS)*, Mar. 2019, pp. 146–149.
- [15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, K. Aisler, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
- [16] N. Zeghidour, N. Usunier, G. Synnaeve, R. Collobert, and E. Dupoux, "End-to-end speech recognition from the raw waveform," 2018, *arXiv:1806.07098*. [Online]. Available: <http://arxiv.org/abs/1806.07098>
- [17] S. Latif, R. Rana, S. Khalifa, R. Jurdak, and J. Epps, "Direct modelling of speech emotion from raw speech," 2019, *arXiv:1904.03833*. [Online]. Available: <http://arxiv.org/abs/1904.03833>
- [18] C. Busso, M. Bulut, C.-C. Lee, A. Kazemzadeh, E. Mower, S. Kim, J. N. Chang, S. Lee, and S. S. Narayanan, "IEMOCAP: Interactive emotional dyadic motion capture database," *Lang. Resour. Eval.*, vol. 42, no. 4, pp. 335–359, Dec. 2008.
- [19] T. Özseven, "A novel feature selection method for speech emotion recognition," *Appl. Acoust.*, vol. 146, pp. 320–326, Mar. 2019.
- [20] H. Zhao, Y. Xiao, J. Han, and Z. Zhang, "Compact convolutional recurrent neural networks via binarization for speech emotion recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 6690–6694.
- [21] R. Tang, W. Wang, Z. Tu, and J. Lin, "An experimental analysis of the power consumption of convolutional neural networks for keyword spotting," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 5479–5483.
- [22] T. Veniat, O. Schwander, and L. Denoyer, "Stochastic adaptive neural architecture search for keyword spotting," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 2842–2846.
- [23] R. Tang and J. Lin, "Deep residual learning for small-footprint keyword spotting," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 5484–5488.
- [24] H. Du, R. Li, D. Kim, K. Hirota, and Y. Dai, "Low-latency convolutional recurrent neural network for keyword spotting," in *Proc. Joint 10th Int. Conf. Soft Comput. Intell. Syst. (SCIS) 19th Int. Symp. Adv. Intell. Syst. (ISIS)*, Dec. 2018, pp. 802–807.
- [25] Y. Zhang, N. Suda, L. Lai, and V. Chandra, "Hello edge: Keyword spotting on microcontrollers," 2017, *arXiv:1711.07128*. [Online]. Available: <http://arxiv.org/abs/1711.07128>
- [26] G. Benelli, G. Meoni, and L. Fanucci, "A low power keyword spotting algorithm for memory constrained embedded systems," in *Proc. IFIP/IEEE Int. Conf. Very Large Scale Integr. (VLSI-Soc)*, Oct. 2018, pp. 267–272.
- [27] M. Walid, S. Bousselemi, K. Dabbabi, and A. Cherif, "Real-time implementation of isolated-word speech recognition system on raspberry Pi 3 using WAT-MFCC," *IJCSNS*, vol. 19, no. 3, p. 42, 2019.
- [28] M. Ben Henia Wiem and Z. Lachiri, "Emotion recognition system based on physiological signals with raspberry Pi III implementation," in *Proc. 3rd Int. Conf. Frontiers Signal Process. (ICFSP)*, Sep. 2017, pp. 20–24.
- [29] A. Mnassri, M. Bennisar, and C. Adnane, "A robust feature extraction method for real-time speech recognition system on a raspberry Pi 3 board," *Eng., Technol. Appl. Sci. Res.*, vol. 9, no. 2, pp. 4066–4070, 2019.
- [30] B. McFee, C. Raffel, D. Liang, D. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "Librosa: Audio and music signal analysis in Python," in *Proc. 14th Python Sci. Conf.*, 2015, p. 1.
- [31] S. Yang, Z. Gong, K. Ye, Y. Wei, Z. Huang, and Z. Huang, "EdgeCNN: Convolutional neural network classification model with small inputs for edge computing," 2019, *arXiv:1909.13522*. [Online]. Available: <https://arxiv.org/abs/1909.13522>
- [32] G. Huang, S. Liu, L. V. D. Maaten, and K. Q. Weinberger, "CondenseNet: An efficient DenseNet using learned group convolutions," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2752–2761.
- [33] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.
- [34] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "ShuffleNet v2: Practical guidelines for efficient CNN architecture design," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 116–131.
- [35] J. Zhao, X. Mao, and L. Chen, "Speech emotion recognition using deep 1D & 2D CNN LSTM networks," *Biomed. Signal Process. Control*, vol. 47, pp. 312–323, Jan. 2019.
- [36] C. Wang, "Interpreting neural network hate speech classifiers," in *Proc. 2nd Workshop Abusive Lang. Online (ALW2)*, 2018, pp. 86–92.
- [37] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [38] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014, *arXiv:1406.1078*. [Online]. Available: <http://arxiv.org/abs/1406.1078>
- [39] Z. Huang, X. Xu, H. Zhu, and M. Zhou, "An efficient group recommendation model with multiattention-based neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jan. 15, 2020, doi: [10.1109/TNNLS.2019.2955567](https://doi.org/10.1109/TNNLS.2019.2955567).
- [40] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," 2018, *arXiv:1804.03209*. [Online]. Available: <http://arxiv.org/abs/1804.03209>
- [41] E. Kim and J. W. Shin, "DNN-based emotion recognition based on bottleneck acoustic features and lexical features," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 6720–6724.



**SHUNZHI YANG** is currently pursuing the master's degree with the School of Computer Science, South China Normal University. His research focuses on the design of efficient network models for edge computing devices.



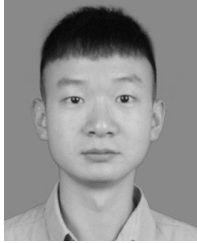
**YUNGEN WEI** is currently pursuing the master's degree with the School of Computer Science, South China Normal University. His research focuses on lightweight model and speech recognition.



**ZHENG GONG** received the M.S. degree in computer science from the South China University of Technology, in 2005, and the Ph.D. degree in computer science from Shanghai Jiaotong University, in 2008. He is currently a Professor of computer science with South China Normal University. His current research directions are lightweight security algorithms and edge computing for resource constrained devices, such as the IoTs, sensors, and mobile phones.



**ZHENHUA HUANG** (Senior Member, IEEE) received the Ph.D. degree in computer science from Fudan University, Shanghai, China, in 2008. From 2012 to 2016, he was an Associate Professor with the School of Electronics and Information Engineering, Tongji University, where he was a Professor, from 2016 to 2018. He is currently a Professor with the School of Computer Science, South China Normal University, Guangzhou, China. Since 2004, he has published three books and more than 80 articles in various journals and conference proceedings. His research interests mainly include deep learning, the Internet of Things, recommendation systems, data mining, knowledge discovery, and big data.



**KAI YE** is currently pursuing the master's degree with the School of Computer Science, South China Normal University, Guangzhou, China. His research focuses on deep learning and lightweight CNNs for face verification.



**ZHENG HUANG** received the Ph.D. degree in computer science from Shanghai Jiaotong University, in 2003. He is currently an Associate Professor of information security with Shanghai Jiaotong University. His current research interests include machine learning and information security.

...