


|  |   |                                    |
|--|---|------------------------------------|
| <b>ITC 4/50</b><br><b>Information Technology<br/>and Control</b><br><b>Vol. 50 / No. 4 / 2021</b><br><b>pp. 656-673</b><br><b>DOI 10.5755/j01.itc.50.4.27352</b> | <b>Low Latency Based Convolutional Recurrent Neural Network<br/>Model for Speech Command Recognition</b>  |                                    |
|  | Received 2020/07/18   | Accepted after revision 2021/07/09 |
|  |  <a href="http://dx.doi.org/10.5755/j01.itc.50.4.27352">http://dx.doi.org/10.5755/j01.itc.50.4.27352</a> |                                    |

**HOW TO CITE:** Kinkar, C. R., Jain, Y. K. (2021). Low Latency Based Convolutional Recurrent Neural Network Model for Speech Command Recognition. *Information Technology and Control*, 50(4), 656-673. <https://doi.org/10.5755/j01.itc.50.4.27352>

# Low Latency Based Convolutional Recurrent Neural Network Model for Speech Command Recognition

**Chhayarani Ram Kinkar**

Electronics & Communication Department, Smarat Ashok Technological Institute, Civil Lines, Vidisha, M.P. India; e-mail: chhayakinkar@gmail.com

**Yogendra Kumar Jain**

Electronics & Instrumentation Department, Smarat Ashok Technological Institute, Civil Lines, Vidisha, M.P. India; e-mail: ykjain\_p@yahoo.co.in

**Corresponding author:** Chhayakinkar@gmail.com

The presented paper proposes a new speech command recognition model for novel engineering applications with limited resources. We built the proposed model with the help of a Convolutional Recurrent Neural Network (CRNN). The use of CRNN instead of Convolutional Neural Network (CNN) helps us to reduce the model parameters and memory requirement as per resource constraints. Furthermore, we insert transmute and curtailment layer between the layers of CRNN. By doing this we further reduce model parameters and float number of operations to half of the CRNN requirement. The proposed model is tested on Google's speech command dataset. The obtained result shows that the proposed CRNN model requires 1/3 parameters as compared to the CNN model. The number of parameters of the CRNN model is further reduced by 45% and the float numbers of operations between 2% to 12 % in different recognition tasks. The recognition accuracy of the proposed model is 96% on Google's speech command dataset, and on laboratory recording, its recognition accuracy is 89%.

**KEYWORDS:** Convolutional Neural Network; Recurrent Neural Network; Gated Recurrent Unit; Low Latency; Speech command recognition.

## 1. Introduction

A natural language speech interaction system is beneficial for the user because there is no learning curve regarding operation of the system. User can operate the system by communicating with it as he/she communicates with other people. However, the development of a natural language speech interaction system is a complex task due to the ambiguous nature of natural language [3]. Scientists and researchers simplify the task by dividing it into two modules namely the speech command recognition module [31] and the natural language understanding module [11]. Moreover, carried out lots of research on both the modules as a result of their hard work today Microsoft- “Cortana”, Amazon- “Alexa”, Apple- “Siri”, Google- “Google assistant”, etc. user-friendly natural language speech interaction systems are available in the market [26, 36].

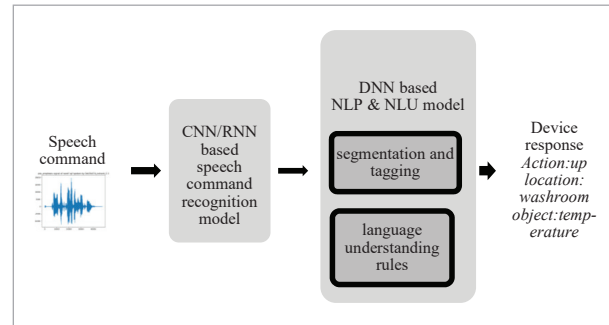
The mentioned natural language speech interaction systems rely on a powerful cloud-based neural network model that requires a broadband connection [36]. However, for novel engineering applications where memory and computational resources are limited, the use of a broadband-based speech interaction system is costly. It also compromises privacy, battery life [26] as well as it highly depends on external factors, for example, network quality [16], network speed [1], latency [27], network traffic [36], etc.

For novel engineering applications, a simple model running on the device and requiring less computational complexity compared to its cloud counterpart is more energy-efficient [25]. Scientists and researchers developed some locally running (standalone [31]) speech interaction system for novel engineering applications, for example, voice-controlled robots, daily required voice-controlled smart home appliances, smart industrial assistive devices, etc. [30]. However, modification in the developed system (Figure 1) is continued for maintaining a balance between the requirement of novel engineering applications and the state-of-the-art performance.

Within the mentioned framework, to achieve the performance as per the cloud-based system with constraints of limited memory and computational resources, the presented paper proposes a hybrid speech command recognition model for natural language speech interaction system used in novel engineering applications. The speech command recogni-

**Figure 1**

Block diagram of natural language speech interaction system for novel engineering applications (Time-domain representation is of speech command- Turn the temperature up in the washroom)



tion model is the heart of the natural language speech interaction system, and its recognition accuracy decides the overall performance of the natural language speech interaction system [36]. By achieving high recognition accuracy with limited model parameters, we try to find a new prospect for natural language speech interaction system used in novel engineering applications.

The main objectives of the presented work are:

- 1 Design of a new speech command recognition model that achieves state-of-the-art performance in spoken command recognition, and it is small in size to fulfill the constraints of novel engineering applications.
- 2 To introduce single word as well as continuous speech command recognition intelligence in the designed model.
- 3 Testing of the designed model on Google speech command dataset.

To fulfill mentioned objectives, we chose Convolution Recurrent Neural Network (CRNN) [29, 37] architecture instead of Deep Neural Network (DNN) [14, 38], Convolution Neural Network (CNN) [1, 13-14] and Recurrent Neural Network (RNN) [16, 23] architecture to build the model. The main reason behind choosing hybrid neural network architecture instead of pure neural network architecture is that we want to reduce the number of trainable parameters and the number of float operations that interns reduce

the computational complexity. The choice of CRNN architecture is motivated by the work of Wang and Zhang [37] on robust voice activity detection (a sub-field of speech recognition).

The presented paper is structured as follows. Section 1 is about the introduction of the topic. Section 2 discusses previous work on speech command recognition. Section 3 explains the processing of speech command for recognition purpose by layers of CRNN and the approach for reducing float operations, the number of trainable parameters in the proposed model. Section 4 discusses the dataset and implementation platform. Section 5 discusses the obtained results. Finally, the paper concluded with the conclusion.

## 2. Related Work

Natural language speech recognition is a challenging task. Leading commercial companies, scientists, researchers, work on speech recognition technology for achieving high recognition accuracy with less computational complexity. As background, this section will discuss approaches and models suggested by scientists, researchers for speech recognition.

Speech recognition technology begins with the recognition of a single phoneme instead of recognizing a continuous word [27]. The phoneme recognition in the state-of-the-art speech recognition model is done with the help of the Gaussian Mixer Model (GMM)-Hidden Markov Model (HMM)-Language Model (LM) paradigm [22, 27]. In the GMM-HMM-LM paradigm, GMM will process input speech feature vector (i.e. Mel Frequency Cepstral Coefficient (MFCC) [30]) and emits emission probability for HMM [5, 22, 27]. The HMM together with LM compute the most likely sequence of phoneme with the help of a decoder [6]. The main drawback of the GMM-HMM-LM based state-of-the-art speech recognition model is that it is unable to recognize the data present on the boundary line [22]. To solve the problem scientists, researchers replaced GMM with DNN [22, 25]. The new DNN-HMM-LM paradigm achieves high recognition accuracy as compared to the GMM-HMM-LM paradigm [22].

With further advancement in technology, the DNN-HMM-LM based speech recognition paradigm is modified into a single deep learning framework that directly recognized a continuous word instead of a sin-

gle phoneme [13]. Within a single deep learning framework, Bahdanau et al. [4] and Ueno et al. [35], proposed RNN based attention model which automatically learns the alignment between the input feature and the respective output sequence. The advantage of the attention deep learning framework is that Markov's assumption is not required for recognition [2]. The drawback of the attention deep learning framework is that in a noisy environment the estimated alignment is easily corrupted by noise and result in poor recognition accuracy [28]. As a solution, Miao et al. [28], Kim et al. [20], Li et al. [24] proposed Context Temporal Classification (CTC) speech recognition model. The CTC speech recognition model improves recognition accuracy in a noise environment. Both CTC and attention deep learning frameworks perform well and achieve an excellent result, but face challenges in incorporating the highly variable features of the natural language like accent style [22], various speaker attributes [25], speed of production of the speech signal [27], etc.

The advancement in deep learning technology continues and Hamid et al. [14], and Guiming et al. [12], replaced the attention-RNN speech recognition model and CTC-RNN speech recognition model with the CNN speech recognition model. Originally CNN is used in image processing applications [1]. To use CNN in speech recognition applications they arranged speech features in the form of a two-dimensional array and trained the CNN speech recognition model using the array. The CNN speech recognition model achieves high recognition accuracy by incorporating the highly variable feature of natural language. However, two main drawbacks of the CNN speech recognition model are- the large number of layers are required to get enough correlation between different frequency bands to achieve high recognition accuracy [32] and the CNN speech recognition model ignores the correlation between different frames [29].

The RNN or CNN employed speech recognition model performs well. However, the huge number of operations in Long Short-Term Memory (LSTM)/ Gated Recurrent Unit (GRU) cell [34, 37] or in the convolutional and max-pooling layer [14] result in a larger model size and limits the use of RNN/CNN employed speech recognition model in novel engineering applications with limited resources.

On the other hand, with the growing demand for natural language speech interaction with devices and sufficient advancement in deep learning technology

Wang G. and Zhang W. [37], and Mukherjee et al. [29] tried CRNN for subfields of speech recognition, for example, keyword recognition, voice activity detection, categorizing sound, etc.

To fulfill the requirement of the current technological scenario in novel engineering applications, i.e. performance as per the cloud-based network, with constrain of limited resources, research work by researchers and commercial groups is continuing.

### 3. Material and Method

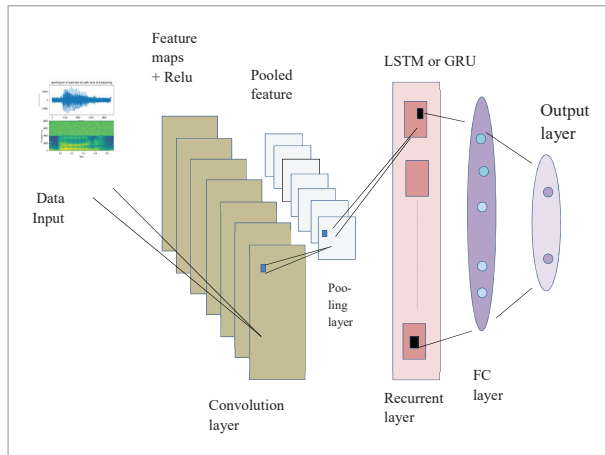
#### 3.1. Convolutional Recurrent Neural Network

The objective of the presented work is to develop a speech command recognition model that will achieve high recognition accuracy with limited model parameters and will suit the requirement of novel engineering applications. To fulfill this objective in the presented work hybrid neural network architecture CRNN (a combination of CNN and RNN as illustrated in Figure 2) is preferred to build the model.

The preference is given to CRNN because the CRNN captures the local spectral correlation in speech command by using convolutional layer and global spectral correlation in speech command by using recurrent layer [29, 37-38]. The simultaneous capturing of local and global spectral correlation of speech command by CRNN help in achieving high recognition accuracy

Figure 2

Basic CRNN for speech command recognition (FC is abbreviation of fully connected layer)



with few layers, that intern helps in reducing computational complexity and model size.

#### 3.2. Speech Command Recognition Using CRNN

The convolutional and recurrent layers of CRNN process the input speech command in the following manner to recognize it.

##### 3.2.1. Processing of Input Speech Command by Convolutional Layer

The first layer in the CRNN architecture is the convolutional layer and the speech command recognition process begin with the convolutional layer  $L_{c_1}$ .

The convolutional layer  $L_{c_1}$  of CRNN consists of two-dimensional convolution filter (number of filters -  $f_{c_1}$ , size of filter -  $s_{z_F} \times s_{z_T}$ , and strides-  $S_T, S_F$ ) [1]. The convolutional filter extract information either from the received spectrogram [22] of the speech command or from the speech feature vector i.e. a context window of  $F$  log Mel band energies  $E \in \mathbb{S}^{F \times T}$  [21] by performing convolution and pooling operations over it. Moreover, pass the extracted information to the next layer through the ReLU activation function [14]. Mathematically the convolution and pooling operation of convolutional layer  $L_{c_1}$  is expressed as:

$$O_j = R(\sum_{i=1}^I a_i * w_{i,j} + b_j) \tag{1}$$

$$P_{i,m} = \max_{n=1}^B O_{i,(m-1) \times s + n}, \tag{2}$$

where \* represent convolution operation,  $w_{i,j}$  represent weighted matrix,  $a_i (i = 1, \dots, I)$  is input feature map  $O_j (j = 1, \dots, J)$  is convolutional feature map,  $b_j$  is bias,  $I$  is the number of the input feature map,  $R$  is ReLU activation function,  $B$  represent pooling size,  $s$  represent shift size.

The output of the convolutional layer  $L_{c_1}$  is fed as an input to the next convolutional layer  $L_{c_2}$  and the process will continue till the last convolutional layer. In CRNN there are  $L_{c_c}$  number of convolutional layer stack together, and the output of the last convolutional layer  $L_{c_c}$  is a tensor and mathematically expressed as:

$$M \in \mathbb{R}^{\mathfrak{A} \times F' \times T}, \tag{3}$$

where  $\mathfrak{A}$  represent the number of feature maps of the last convolutional layer,  $F'$  is the number of frequen-

cy bands reaming after several pooling operations by convolutional layers,  $T$  represent the length of the sequence.

### 3.2.2. Processing of Input Data by Recurrent Layer

The output of the last convolutional layer  $L_{c_c}$  ( $M \in R^{3 \times F \times T}$ ) is feed as input to the second layer of CRNN i.e. recurrent layer as a sequence of frame  $h_{ct}^{L_{c_c}}$ .

The recurrent layer  $L_{r_1}$  consists of  $R_N$  numbers of hidden units in its GRU cell and scans the frame  $h_{ct}^{L_{c_c}}$ , after scanning the recurrent layer  $L_{r_1}$  computes a hidden vector  $h_{rt}$  for each frame as:

$$h_{rt}^{(L_{c_c}+1)} = \varpi(h_{ct}^{L_{c_c}} + h_{rt-1}^{(L_{c_c}+1)}), \quad (4)$$

where the symbol  $\varpi$  represent GRU [16, 23] with two inputs i.e. the output of the current frame of the previous layer  $h_{ct}^{L_{c_c}}$ , and the output of the previous frame of the current layer  $h_{rt-1}^{(L_{c_c}+1)}$ .

The output of the recurrent layer  $L_{r_1}$  is feed as input next recurrent layer  $L_{r_2}$  and the process will

continue till the last recurrent layer. In CRNN there are  $L_{r_r}$  number of recurrent layer stack together, and the last recurrent layer  $L_{r_r}$  computes input for succeeding fully connected layer as:

$$h_{rt}^{(L_{c_c}+L_{r_r})} = \varpi(h_{ct}^{(L_{c_c}+L_{r_r}-1)} + h_{rt-1}^{(L_{c_c}+L_{r_r})}). \quad (5)$$

### 3.2.3. Processing of Input Data by Fully Connected Layer

The output of the last recurrent layer i.e. Equation (5) is feed as an input to the third layer of CRNN i.e. fully connected layer [10, 38]. Each fully connected layer consists of  $FC_n$  numbers of units in it. In CRNN there is  $L_{FC_{FC}}$  number of fully connected layers stack together. The output of the last fully connected layer is computed as:

$$h_{FCt}^{(L_{c_c}+L_{r_r}+L_{FC_{FC}})} = R(h_{rt}^{(L_{c_c}+L_{r_r})}). \quad (6)$$

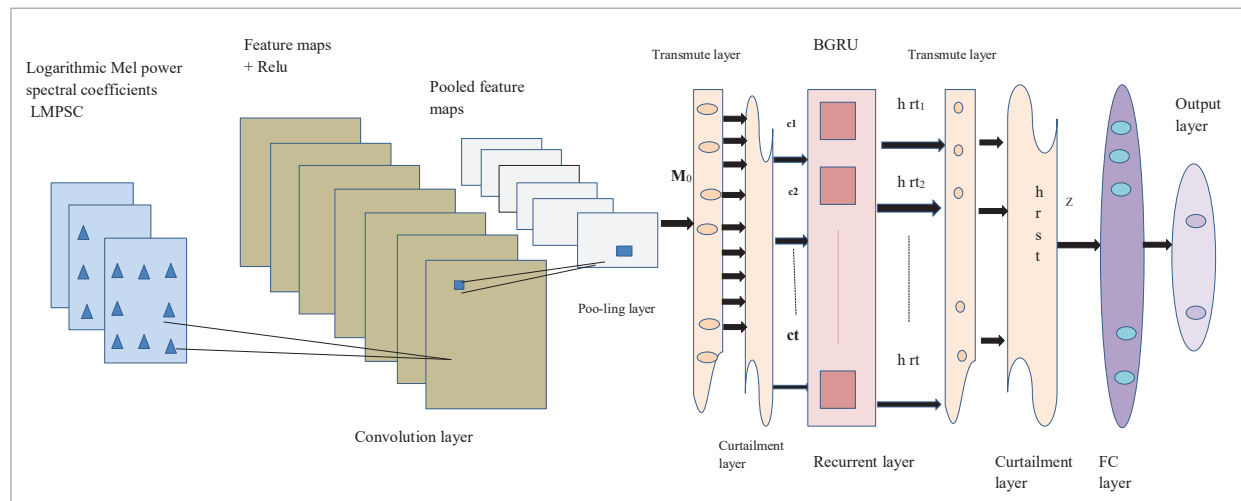
The last fully connected layer is followed by an output layer. The output layer computes the output word sequence from Equation (6) by using a soft-max activation function [15].

### 3.3. Method for Reducing the Number of Parameters and Float Operation in CRNN

In CRNN the number of parameters and the number of float operations is  $((tf_{c_n})/m \times FC_n)$  [29]. To reduce the number of parameters and the number of float operations for fulfilling the main objective of the presented work (i.e. to achieve high recognition accuracy with limited model parameters) the Equations (3) and (5) are processed in a divergent way with the help of a low latency method implemented between the layers of CRNN. The method is illustrated in Figure 3, and it is implemented with the help of curtailment and transmute layer.

Figure3

Proposed hybrid CRNN speech command recognition method





As illustrated in Figure 3, the transmute and curtailment layer will compute the linear combination of the output states of the respective layers in such a way that the number of parameters and float operations in succeeding layers will be reduced on the other hand the succeeding layers will get enough features from coupled states to recognize speech command with high accuracy. Also, the arrangement makes the proposed model more robust to longer speech sequences with reduce number of parameters and float operations.

In the presented work, for the reduction in the number of parameters and float operation, two low latency methods for computing a linear combination of the output states in the transmute and curtailment layer are proposed and discussed in the following text.

**3.3.1. Mean-Weight-Reduction-Method**

In this method, the reduction in the number of parameters and float operations is obtained by computing a linear combination of each time step with the help of weight vector in the curtailment layer as follows:

First, the transmute layer  $L_{T_1}$  transforms the pooled feature maps of  $L_c$  convolutional layer into a feature vector. Next, for a sequence of length  $T$  of feature vector the curtailment layer  $L_{rs_1}$  will compute linear combination as:

$$c_t = R \sum_{i=0}^{T-1} MW_{i,t}, \tag{7}$$

where  $c_t$  represent input for the succeeding recurrent layers,  $W_{i,t}$  is the weight vector with the help of which curtailment layer  $L_{rs_1}$  will compute the input for the succeeding recurrent layers. The weight vector  $W_{i,t}$  of the curtailment layer  $L_{rs_1}$  is computed from the weight vector of Equation (1) and (2) as:

$$W_{i,t} = \sum_{j=0}^I w_{i,j} / ij. \tag{8}$$

Next, Equation (7) is feed as input to the succeeding recurrent layers and for the same sequence of length  $T$  the relationship between input and output of Bidirectional Gated Recurrent Unit (BGRU) cell (consideration of BGRU instead of Bidirectional Long Short-Term Memory (BLSTM) is based on Section 5.2) in the recurrent layer is:

$$Gr_t = R(W_r \cdot [h_{t-1}, c_t] + b_{r1}) \tag{9}$$

$$GU_t = R(W_u \cdot [h_{t-1}, c_t] + b_{u1}), \tag{10}$$

where  $Gr_t$  represent the output of the reset gate of BGRU cell [2],  $GU_t$  represent the output of the update gate of BGRU cell,  $b_{u1}, b_{r1}$  is bias,  $R$  is ReLU activation function.

Equation (9) and (10) shows that at any time instant  $t$ , the output of BGRU cell is the function of input sequence  $c_t$  at time  $t$  and BGRU cell output  $h_{t-1}$  at time  $t-1$  [37]. Now transmute layer  $L_{T_2}$  after recurrent layers will flatten the output of the BGRU cell from shape  $((t \times f_{cn}) / m)$  to shape  $((1 \times f_{cn}) / m)$  and curtailment layer  $L_{rs_2}$  will concatenate the output of the last recurrent layer  $L_{r_t}$  by using the formula:

$$Y_t = R(w_h \cdot [Gr_t * h_{t-1}, c_t]) \tag{11}$$

$$h_{rst} = GU_{r_t} + Y_t, \tag{12}$$

where  $Gr_t, GU_{r_t}$  represents update gate output and reset gate output of the last recurrent layer respectively,  $w_h$  is the weight vector for concatenating,  $h_{rst}$  is concatenated output.

Now  $(h_{rs_0}, \dots, h_{rs_{(G_r-1)}})$  is concatenated output of different BGRU cell of last recurrent layer computed using weight vector  $(w_{h_0}, \dots, w_{h_{(G_r-1)}})$ . By using Equation (12) the curtailment layer  $L_{rs_2}$  will compute a linear combination of concatenated output for succeeding fully connected layer as:

$$Z = R \sum_{i=0}^{G_r-1} h_{rst} (W_{2_i}), \tag{13}$$

where  $Z$  represent new input for the succeeding fully connected layer instead of Equation (5). The weight vector of Equation (13) is computed as:

$$W_{2_i} = (\sum_{h=0}^{G_r-1} w_h) / h. \tag{14}$$

**3.3.2. Mean-Max-Weight-Reduction-Method**

The mean-max-weight-reduction-method for the reduction in the number of parameters and float operation is in the neighborhood of the mean-weight-reduction method. The variability between the methods lies in the computation of the weight vector with the help

of which the curtailment layer computes a linear combination of the time steps. In this method, the computation of the weight vector  $W_{2_t}$  of the curtailment layer is based on the content of each state i.e. if at a time instant  $t$  a frame carries maximum information then the weight vector is longer for this frame as compared to another frame with the minimum information.

This method will follow the same procedure discussed above (Section 3.3.1). The equivalent equation for computing input for the succeeding recurrent layer by curtailment layer  $L_{rs1}$  is the same as Equation (7) and (8). The equation for computing input for succeeding fully connected layer by curtailment layer  $L_{rs2}$  is as follows:

$$Z = R \sum_{t=0}^{G_{r-1}} h_{rst} W_{2_t} \quad (15)$$

$$W_{2_t} = \max_{h=0}^{G_{r-1}} w_h. \quad (16)$$

In the CRNN architecture, the number of float operations is  $((f_{c_n})/m \times FC_n)$  and the number of parameters is  $((f_{c_n})/m \times FC_n)$  [18, 29]. With the help of Equation (13) or Equation (15), the proposed method will reduce the number of float operation to  $((f_{c_n} + t)/m \times FC_n)$  and the number of parameters to  $((f_{c_n})/m \times FC_n)$ .

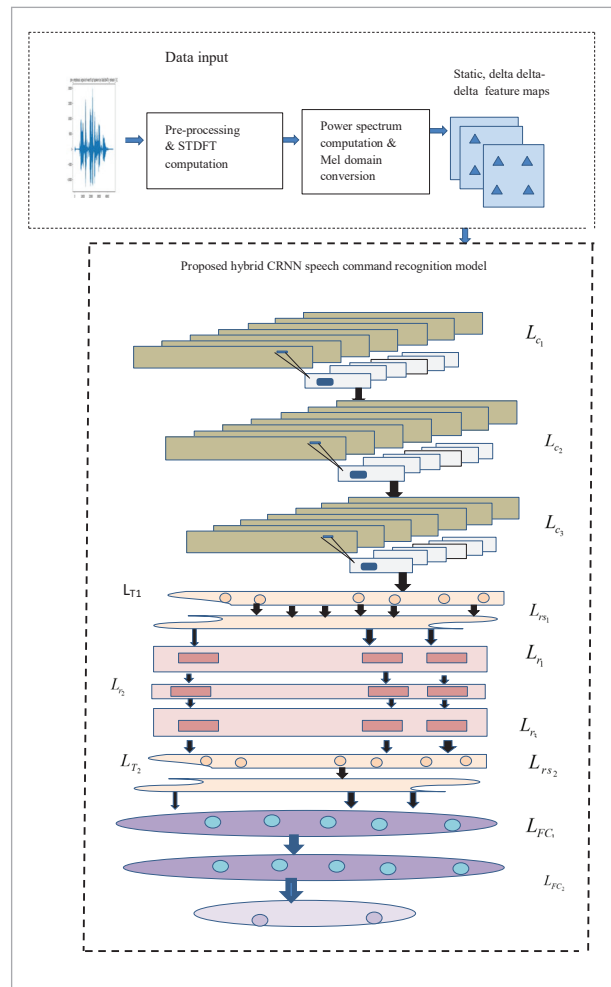
### 3.4. High-Level Description of the Proposed Model

The high-level architecture of the proposed hybrid CRNN speech command recognition model is illustrated in Figure 4. The high-level description of network structure starting from the data input method is as follows:

#### 3.4.1. Data Input

Speech commands are in the form of a one-dimensional vector [8]. To feed it to the 2D convolution layer of CRNN Himid et al. [14] suggest either to convert it into the spectrogram and feed the  $L_{c_1} \in L_{c_c}$  convolutional layer of CRNN with the spectrogram or extract Mel Frequency Spectral Coefficients (MFSCs) from the extremely long speech command and organized the extracted features in the form of a maps and feed  $L_{c_1} \in L_{c_c}$  convolutional layer of CRNN with organized feature maps i.e. with a context window of  $F \log$  Mel

**Figure 4**  
Proposed speech command recognition model



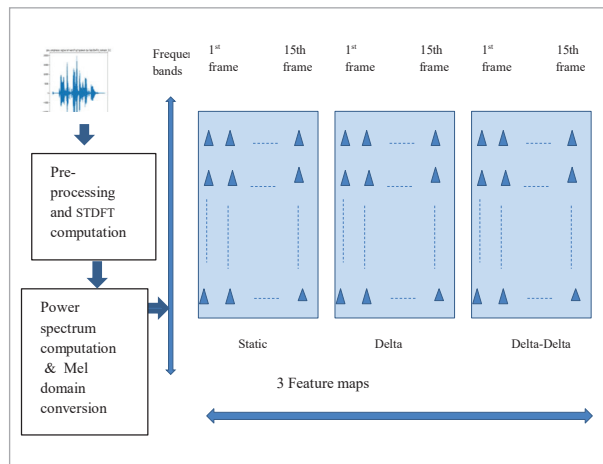
band energies over  $T$  frames [21]. In the presented work, the second method is preferred to feed the proposed model.

However, in the presented work, the proposed model is fed with Logarithmic Mel Power Spectral coefficients (LMPSC) i.e. without discrete cosine transform instead of traditional MFCC. The reason behind feeding the proposed model with LMPSCs instead of MFCC is that spectral features (LMPSC) carry more information as compare to cepstral (MFCC) features [5]. Moreover, the computation of features without discrete cosine transform helps in reducing the overall computational complexity of the proposed model.

*Extractions of LMPSCs* – In the presented work, LMPSCs are extracted as per the European Telecommunication Standards Institute (ETSI) standard [7]. According to the ETSI standard first, the input speech command is pre-processed to convert it into a discrete form [8]. After pre-processing the discrete-time speech signal is converted into the Mel domain by computing Short-Time Discrete Fourier Transform (STDFT) [19] and passing the power spectrum of computed STDFT through a triangularly weighted Mel scale filter bank [8]. The resulting Mel power spectrum is compressed by taking its natural log. From the logarithmically compressed Mel power spectrum, LMPSCs are extracted and arranged in the form of feature vectors.

*Organization of extracted LMPSC feature vectors as a feature map* - The organization of extracted LMPSC feature vectors as feature maps for the convolutional layer  $L_{c_1}$  of the proposed model is illustrated in Figure 5.

**Figure 5**  
Organization of data input to proposed speech command recognition model



In this organization, the extracted LMPSC features are arranged as three 2-D feature maps representing static, delta, and delta-delta representation [14] of spectral features distributed along both i.e. frequency (by using frequency band index) and time (by using frame number within each context window) [12]. The presented organization is inspired by the work of Hamid et al. [14].

### 3.4.2. Convolutional Layer ( $L_c$ )

The proposed speech command recognition model goes ahead with two-dimensional convolutional layer followed by max-pooling layer, ReLU activation. In the proposed model three convolutional layers  $L_{c_1}, L_{c_2}, L_{c_3}$  with the number of filters  $f_{cn} = 64$  and size  $(sz_F \times sz_T = (20, 5))$ , stride  $(S_T \times S_F = (4, 1))$  are sequentially placed. The convolutional layer  $L_{c_1}, L_{c_2}, L_{c_3}$  will process the input speech features as discussed in Section 3.2.1.

### 3.4.3. Transmute Layer ( $L_T$ )

In the proposed model transmute layer is placed after the convolutional layer  $L_{c_3}$  and the recurrent layer  $L_{r_3}$  for the reduction in the number of parameters and float operation. This layer is implemented with the feed-forward concept with 32 units in it.

### 3.4.4. Curtailment Layer ( $L_{rs}$ )

In the proposed model, each transmute layer is followed by a curtailment layer to compute a linear combination of time steps with the help of equation (7) to equation (16) for the reduction in the number of parameters and float operation. This layer is implemented with 64 numbers of units in it.

### 3.4.5. Bidirectional Recurrent Layer ( $L_r$ )

In the proposed model, the curtailment layer  $L_{rs_1}$  is followed by bidirectional recurrent layers with  $R_N = 64$  numbers of hidden units in its BGRU cell. Bidirectional GRU cell is preferred in the proposed model because the bidirectional GRU cell considers present time steps as well as future time steps. This will help optimally incorporating the time dimensional features. In the proposed model, three bidirectional recurrent layers  $L_{r_1}, L_{r_2}, L_{r_3}$  are sequentially placed and will process the speech features as discussed in Section 3.2.2.

### 3.4.6. Fully Connected Layer ( $L_{FC}$ )

In the proposed model the curtailment layer  $L_{rs_2}$  is followed by the fully connected layer with the number of units  $FC_n = 64$  in it. In the proposed model two fully connected layers  $L_{FC_1}, L_{FC_2}$  are sequentially placed and process the received data as discussed in Section 3.2.3. The fully connected layer  $L_{FC_2}$  is followed by the output layer which generates the output word sequence using the soft-max activation function.



## 4. Experimental Setup

### 4.1. Dataset

The proposed model is tested on two datasets developed by the Google group for academic research. The first dataset is Google's single word speech command dataset developed by TensorFlow and AIY team [33]. This dataset consists of 65000 audio utterances spoken by male and female speakers. The duration of each audio utterance in this dataset is of one second. This dataset consists of 30 single word commands including digits from zero to nine. The second dataset is a fluent speech command (Flu. comm.) dataset developed by the Google group [9]. This dataset consists of 30,043 utterances, spoken by 97 male and female speakers. Each utterance of this dataset is a continuous sentence that is used to control smart home appliances or virtual assistants, for example, "put on the music" or "turn on the lights". The utterances of both the datasets are recorded with phone and laptop as a .wav file. The recorded utterances are sampled at 16 kHz as a single-channel signal.

### 4.2. Implementation

The proposed model is implemented on the python-Tensorflow platform using the Keras interface [15, 17-18]. Among available platforms, preference is given to this platform because of its data handling capacity and flexibility to model predictive modeling problems with few lines of codes [15, 17]. In Keras, the proposed model is implemented via sub-classing [10]. Transmute and curtailment layers are implemented via layer class definition of the customize layer using the build, call, add function, and setting the trainable weight argument [18]. The data input for the proposed model from the speech command dataset is computed according to the ETSI standard [7]. As per the ETSI standard, the speech command pre-processing specifications are tabulated in Table 1, and LMPSCs computation parameters are tabulated in Table 2.

**Table 1**

ETSI standard signal pre-processing parameters

| $\alpha$ | Frame size | Frame stride | Window type |
|----------|------------|--------------|-------------|
| 0.97     | 25ms       | 10ms         | Hamming     |

**Table 2**

ETSI standard feature vector computation parameters

| N-point STDFT | No. of triangular filters |
|---------------|---------------------------|
| 512           | 40                        |

## 5. Result and Discussion

To test the proposed model, series of experiments are performed on both the datasets (single word, and fluent command). The obtained results of different experiments are discussed in this section.

### 5.1. Impact of Convolutional Layer and Recurrent Layer on the Performance of the Proposed Model

The proposed model is built using CRNN. The CRNN is the combination of CNN and RNN [29]. Therefore the number of convolutional layers and the number of recurrent layers in CRNN will decide the recognition accuracy of the proposed model. In the first experiment, the impact of the number of convolutional layers, number of convolution filters, number of hidden units in the recurrent layer, the number of recurrent layers on the performance of the proposed model is analyzed. To analyze the impact, both the dataset (single word dataset and fluent command dataset) are split into training, validation, and testing sets with a ratio 6-1-1 [38]. The batch size is 45. The training is done until convergence though the different composition of CRNN requires a different number of epochs. The initial learning rate of each composition of CRNN is 0.001, with a decay of 0.5 after every ten epochs. The obtained recognition accuracies along with detailed specifications of the number of convolutional layers, size of convolution filter, the number of recurrent layers, number of hidden units in the recurrent layer, number of units in the fully connected layer are tabulated in Table 3.

The exploration of experimentally obtained results shows that CRNN composition with two convolution layers and a single recurrent layer requires the number of parameters around 178K to 266K with an average recognition accuracy of 86% in different recognition tasks. This recognition accuracy is below the threshold recognition accuracy of the cloud-based

**Table 3**

Impact of the number of convolutional and recurrent layers on the performance of the proposed model (hidden unit is abbreviated as hi. un. number of parameters is abbreviated as NOP, recognition accuracy is abbreviated as RA)

| Convolutional layer |                |                |              | Recurrent layer |               | FC layer     | Word        |               | Digit       |               | Flu. comm.  |               |
|---------------------|----------------|----------------|--------------|-----------------|---------------|--------------|-------------|---------------|-------------|---------------|-------------|---------------|
| No. of layer        | No. of filters | Size of filter | Strides      | No. of layer    | No. of hi.un. | No. of units | NOP         | RA            | NOP         | RA            | NOP         | RA            |
| 2                   | 32             | (20,5)         | (4,1)        | 1               | 32            | 32           | 189K        | 0.8836        | 178K        | 0.8644        | 266K        | 0.8677        |
| 2                   | 32             | (20,5)         | (4,1)        | 2               | 64            | 64           | 196K        | 0.9069        | 189K        | 0.8869        | 289K        | 0.8825        |
| 3                   | 64             | (20,5)         | (4,1)        | 1               | 32            | 32           | 225K        | 0.9128        | 214K        | 0.9005        | 303K        | 0.8966        |
| 3                   | 32             | (20,5)         | (4,1)        | 2               | 32            | 32           | 246K        | 0.9328        | 237K        | 0.9255        | 328K        | 0.9166        |
| <b>3</b>            | <b>64</b>      | <b>(20,5)</b>  | <b>(4,1)</b> | <b>3</b>        | <b>64</b>     | <b>64</b>    | <b>268K</b> | <b>0.9634</b> | <b>259K</b> | <b>0.9606</b> | <b>354K</b> | <b>0.9533</b> |
| 3                   | 32             | (20,5)         | (4,1)        | 3               | 32            | 32           | 250K        | 0.9542        | 243K        | 0.9513        | 345K        | 0.9421        |
| 3                   | 64             | (20,5)         | (4,1)        | 4               | 32            | 32           | 292K        | 0.9601        | 287K        | 0.9589        | 408K        | 0.9541        |

system (Microsoft Cortana-95%, Apple siri-95%). Whereas CRNN composition with three convolution layers and three recurrent layers require trainable parameters around 259K to 354K, but this composition achieves recognition accuracy of the cloud-based system i.e. around 95% in different recognition tasks. The CRNN composition with three convolution layers and four recurrent layers has a limited impact on improvement in recognition accuracy. By increasing one more recurrent layer, only 1% improvement in recognition accuracy is observed whereas the model parameters are increased as an average of around 35K in different recognition tasks.

The size of the CRNN model in terms of the number of parameters and recognition accuracy is directly proportional to the number of convolutional filters, the number of hidden units in the recurrent layers, and the number of units in the fully connected layer. By increasing them, recognition accuracy and the number of parameters increase.

From the obtained result of the first experiment, (highlighted in Table 3) the high-level architecture of the proposed model in Figure 4 is drawn. The chosen size is approximately half of the depth wise separable CNN model [2].

## 5.2. Impact of RNN Variants on the Performance of the Proposed Model

In the first experiment, the recurrent layer is implemented with BGRU; However, BGRU and BLSTM

both are popular variants of RNN. Scientists and researchers use both to build RNN based speech recognition model [16, 23]. So the second experiment is performed to inspect which one BGRU or BLSTM is better for fulfilling the main objective of the present work.

In the second experiment, the convolutional layer is implemented with highlighted specifications of Table 3, and the recurrent layer is implemented with the help of BLSTM and BGRU one by one. Both the times the number of units of fully connected layer after recurrent layer is kept constant. The computational complexity (number of model parameters) and recognition accuracy of both the variants is computed to compare their impact on the proposed model.

The comparison is made on three tasks i.e. word, digit, and fluent command recognition task. For each task, the proposed model is trained for maximum 40 epochs [38]. The training of the proposed model will stop if no improvement is observed after ten consecutive epochs. The initial learning rate of the proposed model is 0.001 and decay of 0.5 after every ten epochs. The batch size is 45. The training, validation, and testing dataset in the second experiment is the same set as the first experiment.

The obtained result of the word, digit, and fluent command recognition task is tabulated in Table 4.

Analysis of obtained results of the second experiment shows that when the recurrent layer in the proposed

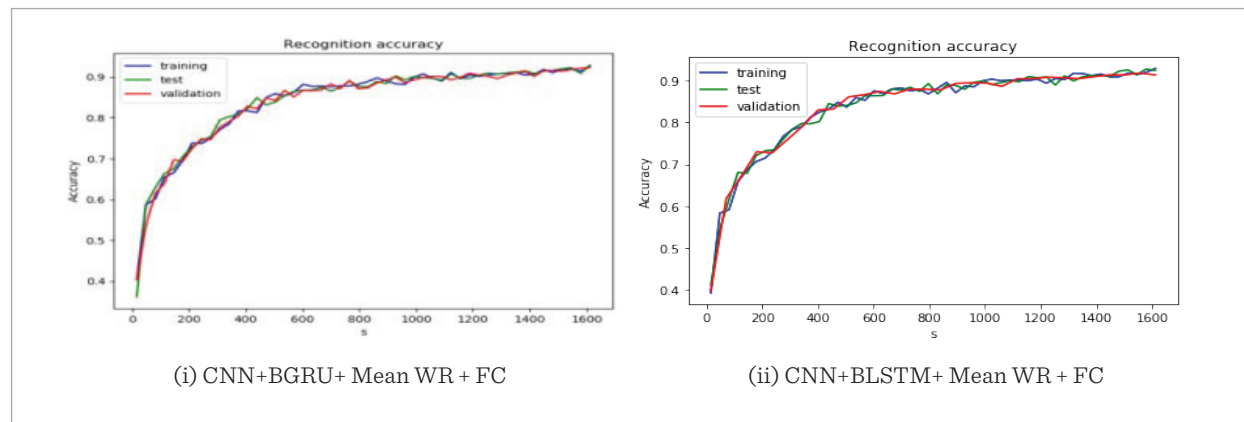
**Table 4**

Comparison of RNN variants in the proposed model when number of units of fully connected layer is 64 (NOP is abbreviation of number of parameters, TR is abbreviation of training, TE is abbreviation of testing, VA is abbreviation of validation, Mean WR is abbreviation of mean-weight-reduction-method, Max WR is abbreviation of mean-max-weight-reduction method)

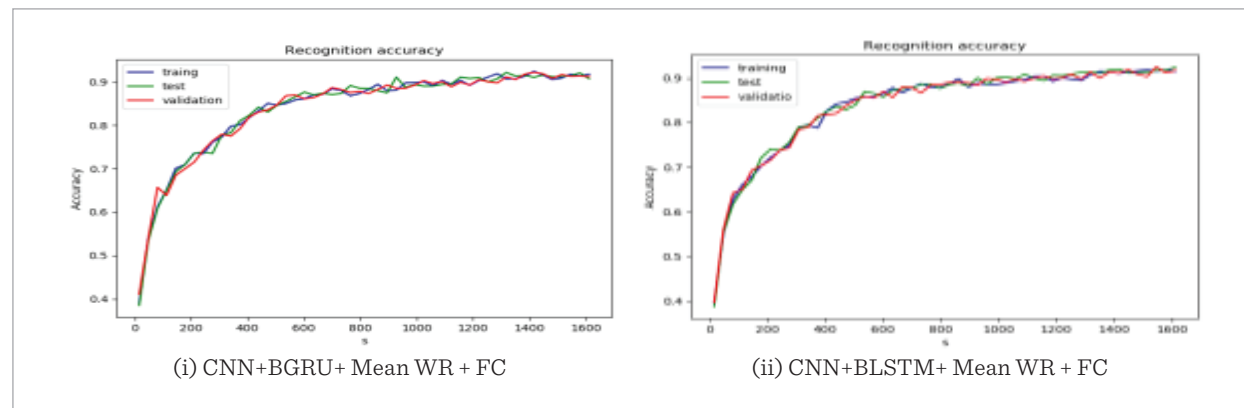
| Model                   | Word recognition accuracy |        |        |               | Digit recognition accuracy |        |        |               | Flu. comm. recognition accuracy |        |        |               |
|-------------------------|---------------------------|--------|--------|---------------|----------------------------|--------|--------|---------------|---------------------------------|--------|--------|---------------|
|                         | NOP                       | TR     | TE     | VA            | NOP                        | TR     | TE     | VA            | NOP                             | TR     | TE     | VA            |
| CNN+ BGRU+FC            | 268K                      | 0.9808 | 0.9583 | 0.9634        | 259K                       | 0.9815 | 0.9527 | 0.9606        | 372K                            | 0.9678 | 0.9361 | 0.9533        |
| CNN+BGRU+ Mean WR + FC  | 148K                      | 0.9823 | 0.9752 | <b>0.9648</b> | 136K                       | 0.9863 | 0.9579 | <b>0.9631</b> | 198K                            | 0.9699 | 0.9523 | <b>0.9585</b> |
| CNN+BGRU+ Max WR +FC    | 148K                      | 0.9818 | 0.9631 | 0.9641        | 136K                       | 0.9843 | 0.9608 | 0.9602        | 198K                            | 0.9718 | 0.9423 | 0.9543        |
| CNN+ BLSTM+FC           | 539K                      | 0.9801 | 0.9512 | 0.9512        | 492K                       | 0.98   | 0.9411 | 0.9567        | 713K                            | 0.9651 | 0.9334 | 0.9431        |
| CNN+BLSTM++ Mean WR+ FC | 273K                      | 0.9811 | 0.9651 | 0.9501        | 252K                       | 0.9833 | 0.9514 | 0.9579        | 372K                            | 0.9702 | 0.9461 | 0.9418        |
| CNN+BLSTM+ Max WR +FC   | 273K                      | 0.9803 | 0.9563 | 0.9504        | 252K                       | 0.9825 | 0.9516 | 0.9562        | 372K                            | 0.9701 | 0.9411 | 0.9413        |

**Figure 6 (a)**

Learning curve of the proposed model during word recognition task by implementing RNN layer with BGRU and BLSTM

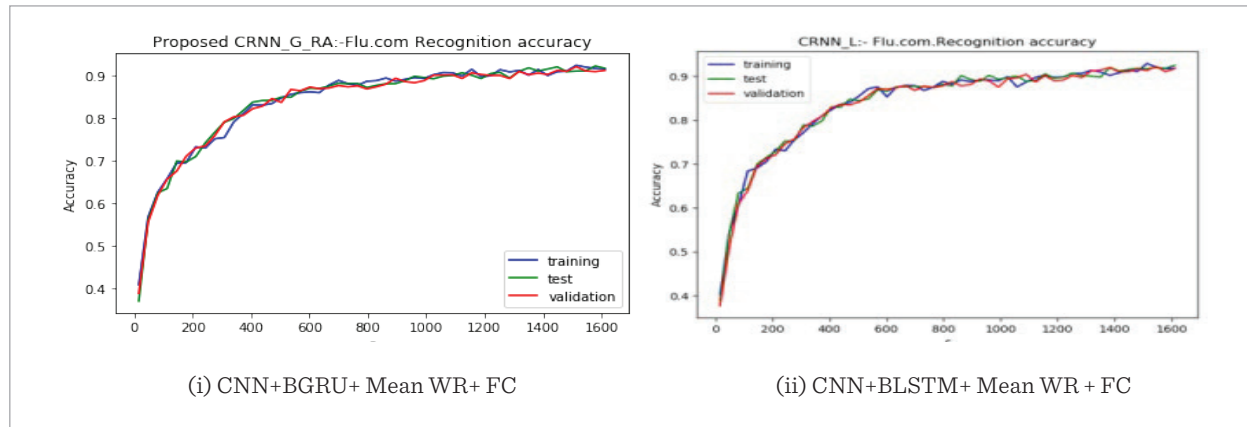
**Figure 6 (b)**

Learning curve of the proposed model during digit recognition task by implementing RNN layer with BGRU & BLSTM



**Figure 6 (c)**

Learning curve of the proposed model in fluent command recognition task by implementing RNN layer with BGRU and BLSTM



model is implemented with BLSTM, the model can recognize 95% word, 95% digit, and 94% fluent command correctly (Figure 6(a)-(ii) to Figure 6(c)-(ii)). Whereas when the recurrent layer is implemented with BGRU, the proposed model can recognize 96% word, 96% digit, and 95% fluent command correctly ((Figure 6(a)-(i) to Figure 6(c)-(i)). The recognition accuracy around 95% with the help of both variants i.e. BGRU and BLSTM will be as good as a cloud-based system.

The comparison of the required number of parameters by BLSTM and BGRU shows that BGRU requires around forty-seven percent fewer parameters as compared to BLSTM (specifically fifty percent less for word recognition and forty-six percent less for digit recognition and forty-seven percent for fluent command recognition). The use of the mean/mean-max weight-reduction method in BLSTM makes it compatible with BGRU in terms of the number of parameters. Whereas the use of mean/mean-max-weight-reduction method in BGRU further reduces the model parameters.

From the analysis of the obtained result of the second experiment, it is confirmed that BGRU is more suitable in the recurrent layer of the proposed model to reduce complexity and memory requirement as per the requirement of novel engineering applications.

### 5.3. Impact of Proposed Reduction Method on the Performance of the Proposed Model

To make the proposed model more robust to longer speech sequences with the reduced number of pa-

rameters and float operations, transmute and curtailment layers are inserted. The transmute and curtailment layers will compute linear combination of states of CNN layer and RNN layer with the help of mean-weight-reduction method or mean-max-weight-reduction method. In the third experiment, both methods are tested for different numbers of units in the last fully connected layer.

The batch size, learning rate in the third experiment is the same as the second experiment, but this time to get more accurate results, we train the proposed model for maximum 50 epochs [38]. The obtained recognition accuracies of the word, digit, and fluent commands recognition task along with the number of units of fully connected layer, the number of parameters, and float number of operations are tabulated in Table 5(a), 5(b), 5(c), 5(d).

Analysis of the obtained result shows that for word recognition accuracy around 96%, digit recognition, and fluent command recognition accuracy around 95% (which is as good as a cloud-based system) the proposed reduction approach with the help of transmute and curtailment layers reduces float number of operations which in turn reduces the number of parameters. Specifically, if 32 units of fully connected layers are used to implement the proposed model, then the parameters are reduced by 43% (compared with the basic CNN+ BGRU+FC model) as illustrated in figure 7(a). If 64/128/256 units of fully connected layers are used to implement the proposed model, then the reduction is 45%, 53%, 64% respectively (figure 7(b)

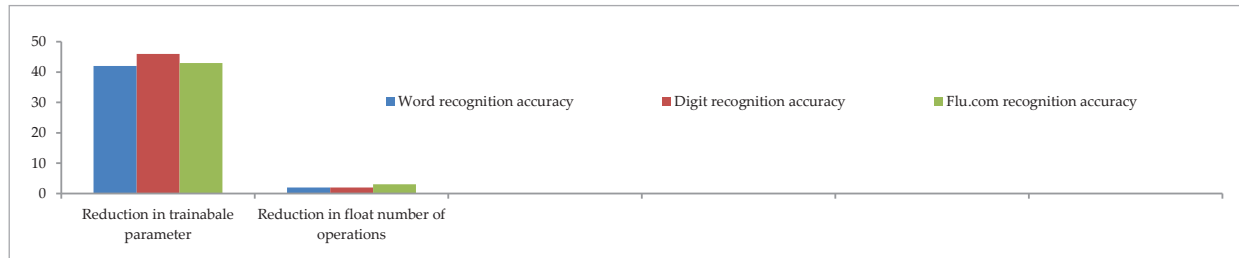
**Table 5 (a)**

Comparison between number of units of fully connected layer and number of parameters and float number of operations in the proposed model (NOP is abbreviation of number of parameters, NFO is abbreviation of number of float operations, RA is abbreviation of recognition accuracy)

| No. of unit of FC layer | Model                 | Word recognition accuracy |        |        | Digit recognition accuracy |        |        | Flu. comm. recognition accuracy |        |        |
|-------------------------|-----------------------|---------------------------|--------|--------|----------------------------|--------|--------|---------------------------------|--------|--------|
|                         |                       | NOP                       | NFO    | RA     | NOP                        | NFO    | RA     | NOP                             | NFO    | RA     |
| 32                      | CNN+ BGRU+FC          | 250K                      | 4.471M | 0.9542 | 243K                       | 4.471M | 0.9513 | 345K                            | 6.379M | 0.9421 |
|                         | CNN+BGRU+ Mean WR+ FC | 144K                      | 4.349M | 0.9634 | 130K                       | 4.349M | 0.9589 | 195K                            | 6.282M | 0.9523 |
|                         | CNN+BGRU+ Max WR+FC   | 144K                      | 4.349M | 0.9631 | 130K                       | 4.349M | 0.9571 | 195K                            | 6.282M | 0.9499 |

**Figure 7 (a)**

For 32 unit of fully connected layer obtained reduction in number of parameters and float number of operations

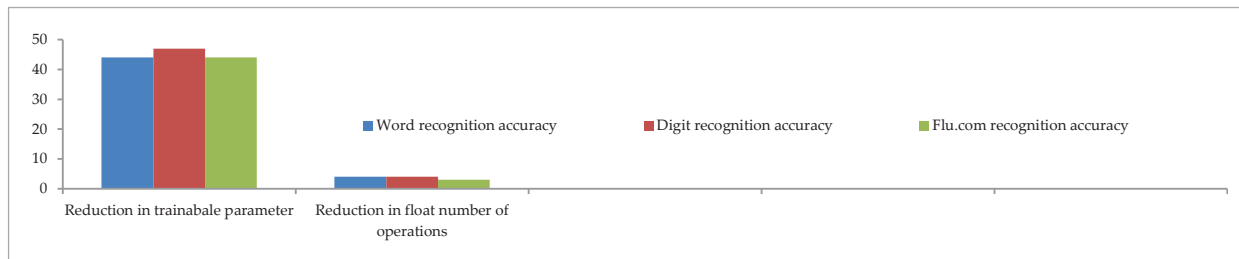
**Table 5 (b)**

Comparison between number of units of fully connected layer and number of parameters and float number of operations in proposed model (NOP is abbreviation of number of parameters, NFO is abbreviation of number of float operations, RA is abbreviation of recognition accuracy)

| No. of unit of FC layer | Model                 | Word recognition accuracy |        |        | Digit recognition accuracy |        |        | Flu. comm. recognition accuracy |        |        |
|-------------------------|-----------------------|---------------------------|--------|--------|----------------------------|--------|--------|---------------------------------|--------|--------|
|                         |                       | NOP                       | NFO    | RA     | NOP                        | NFO    | RA     | NOP                             | NFO    | RA     |
| 64                      | CNN+ BGRU+FC          | 268K                      | 4.552M | 0.9634 | 259K                       | 4.552M | 0.9606 | 354K                            | 6.428M | 0.9533 |
|                         | CNN+BGRU+ Mean WR+ FC | 148K                      | 4.392M | 0.9648 | 136K                       | 4.392M | 0.9631 | 198K                            | 6.259M | 0.9585 |
|                         | CNN+BGRU+ Max WR+FC   | 148K                      | 4.392M | 0.9641 | 136K                       | 4.392M | 0.9602 | 198K                            | 6.259M | 0.9543 |

**Figure 7 (b)**

For 64 unit of fully connected layer obtained reduction in number of parameters and float number of operations





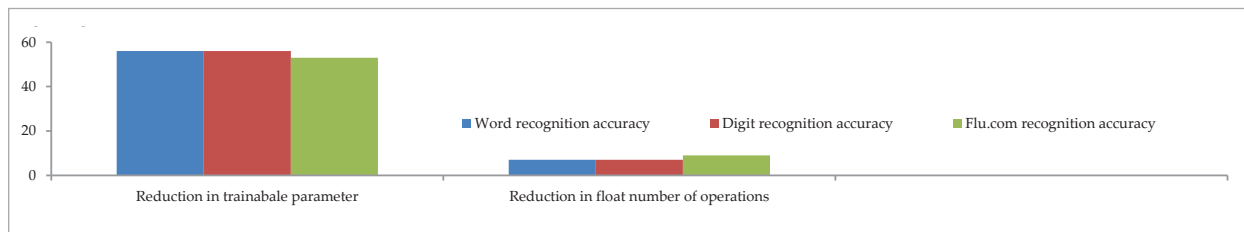
**Table 5 (c)**

Comparison between number of units of fully connected layer and number of parameters and float number of operations in proposed model (NOP is abbreviation of number of parameters, NFO is abbreviation of number of float operations, RA is abbreviation of recognition accuracy)

| No. of unit of FC layer | Model                 | Word recognition accuracy |        |        | Digit recognition accuracy |        |        | Flu. comm. recognition accuracy |         |        |
|-------------------------|-----------------------|---------------------------|--------|--------|----------------------------|--------|--------|---------------------------------|---------|--------|
|                         |                       | NOP                       | NFO    | RA     | NOP                        | NFO    | RA     | NOP                             | NFO     | RA     |
| 128                     | CNN+BGRU+FC           | 392K                      | 4.764M | 0.9652 | 375K                       | 4.764M | 0.9492 | 507K                            | 6.8172M | 0.9562 |
|                         | CNN+BGRU+ Mean WR+ FC | 171K                      | 4.432M | 0.9679 | 164K                       | 4.432M | 0.9464 | 238K                            | 6.2982M | 0.9589 |
|                         | CNN+BGRU+ Max WR+FC   | 171K                      | 4.432M | 0.9662 | 164K                       | 4.432M | 0.9491 | 238K                            | 6.2982M | 0.9563 |

**Figure 7 (c)**

For 128 unit of fully connected layer obtained reduction in number of parameters and float number of operations



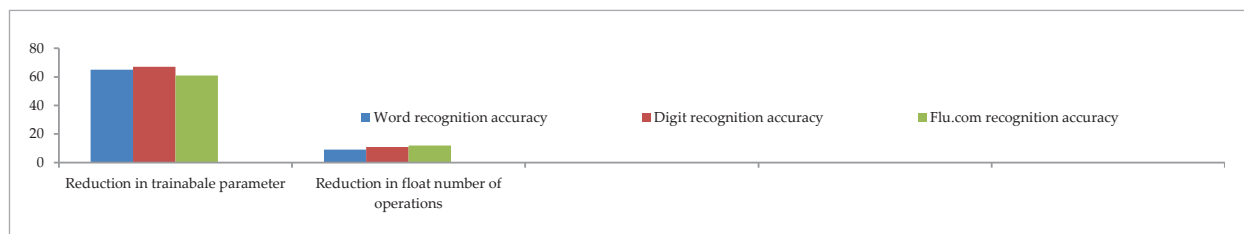
**Table 5 (d)**

Comparison between number of units of fully connected layer and number of parameters and float number of operations in proposed model (NOP is abbreviation of model parameters, NFO is abbreviation of number of float operations, RA is abbreviation of recognition accuracy)

| No. of unit of FC layer | Model                 | Word recognition accuracy |        |        | Digit recognition accuracy |        |        | Flu. Comm. recognition accuracy |        |        |
|-------------------------|-----------------------|---------------------------|--------|--------|----------------------------|--------|--------|---------------------------------|--------|--------|
|                         |                       | NOP                       | NFO    | RA     | NOP                        | NFO    | RA     | NOP                             | NFO    | RA     |
| 256                     | CNN+ BGRU+FC          | 580K                      | 5.198M | 0.9682 | 564K                       | 5.198M | 0.9473 | 804K                            | 7.208M | 0.9575 |
|                         | CNN+BGRU+ Mean WR+ FC | 198K                      | 4.721M | 0.9695 | 181K                       | 4.721M | 0.9514 | 311K                            | 6.382M | 0.9596 |
|                         | CNN+BGRU+ Max WR+FC   | 198K                      | 4.721M | 0.9686 | 181K                       | 4.721M | 0.9518 | 311K                            | 6.382M | 0.9594 |

**Figure 7(d)**

For 256 unit of fully connected layer obtained reduction in model parameters and float number of operations



to 7(d)). However, for a higher number of units of the fully connected layer (256 unit), only model parameters increases (from 144K -198K); the recognition accuracy remains almost constant.

The comparison between the mean-weight-reduction method and the mean-max-weight-reduction method shows that both methods are nearly equal to each other and reduce the number of float operations and the number of model parameters approximately by an equal amount. However, the recognition accuracy by mean-weight-reduction method is slightly better than the mean-max-weight-reduction method. Specifically, the mean-weight-reduction method achieves the highest recognition accuracy for 64 units of the fully connected layer with a moderated number of parameters during testing of the proposed model. From the comparison, we can say that the mean-weight-reduction method represents better performance to get an acceptable threshold (by reducing the number of parameters to 1/3 of the depth wise separable CNN model [2]) for the natural language speech interaction system used in novel engineering applications.

Analysis of obtained result in concern with float number of operation shows that both the method reduces the float number of operation between 2% to 12% in different recognition tasks (figure 7(a) to 7(d)).

Investigations about the number of units of the fully connected layers and its impact on the recognition accuracy of the proposed model show that the different number of units of fully connected layers has little impact on recognition accuracy. The convolutional and recurrent part has more influence on the recognition accuracy of the proposed model.

#### 5.4. Comparison with Other Models

The proposed model is compared with the attention RNN model and depth wise separable CNN model [2]. To make the comparison with both the models the testing dataset of the proposed model contains the word of Google's single word speech commands dataset. Digit and fluent command recognition tasks are not compared because the published article contains the result of the word recognition task. The comparison is shown in Table 6.

The comparison table shows that the proposed model is compatible with other models. With the proposed model we get high recognition accuracy with the low-

**Table 6**

Comparison with other models

| Model                    | Accuracy (%) | Model parameters |
|--------------------------|--------------|------------------|
| Attention RNN            | 96.9         | 202K             |
| Depth wise separable CNN | 95.4         | 498K             |
| CNN+RNN+MeanWR+FC        | 96.5         | 148K             |

est number of trainable parameters. The recognition accuracy of the attention RNN model is higher than the proposed model but for 0.4% higher recognition accuracy this model compromises with 26% more model parameters.

#### 5.5. Performance of the Proposed Model on Realistic Recording

In the fourth experiment, the proposed model was tested on realistic recordings [22]. The main aim behind this testing is to check the performance of the proposed model, which is trained on lots of data recorded in a clean and control environment on field data [5]. For this, the same commands of both the dataset are recorded in a laboratory. The details of laboratory recordings are tabulated in the appendix section. The testing dataset consists of 563 utterances of the word, 254 utterances of digits, and 384 utterances of fluent commands. All other setting is as per experiment three. The obtained recognition accuracies are tabulated in Table 7.

**Table 7**

Recognition accuracy of the proposed model on field data

| Flu.comm. | Word | Digit |
|-----------|------|-------|
| 89%       | 90%  | 87%   |

The recognition accuracy of the proposed model on field data is low as compared to data recorded in a clean and control environment. This may be due to the mismatch between accent styles, or mixing of echoes with the recorded signal, or due to the different recording protocol, etc. but recognition accuracy of the proposed model is within the acceptable range to use it, in the natural language speech interaction system of novel engineering applications.

To improve the recognition accuracy on field data some modification are needed in the feature extraction process so that clean speech features are extracted from recorded speech (in the realistic environment echoes from the surrounding environment mix with recordings).

## 6. Conclusion

By implementing the proposed speech command recognition model in a hybrid way i.e. by using CRNN, transmute, and curtailment layers instead of a conventional way, we try to find a new solution for natural language speech interaction system used in novel engineering applications. The main constrains of these applications (limited resources) is fulfilled by re-

ducing the computational complexity in terms of the number of parameters and the number of float operation which in turn reduces the memory requirement.

The experimental investigation on Google's speech command dataset shows that the proposed model can recognize 96% word, 95% digit, and 95% fluent commands correctly with 2%-12% less number of float operations and around 45% fewer trainable parameters in different recognition tasks. Testing of the proposed model on field data shows that the proposed model can achieve recognition accuracy around 89% on field data.

The future work will include testing of different training strategies (for example data augmented method) on the proposed model to overcome its performance limitation on realistic recordings.

## Appendix

The details of laboratory recording are tabulated in Table 8. The audio recordings are captured by using Py-Audio package 0.20.11 [36] and sampled at 16 kHz

**Table 8**

Details of laboratory recordings

| Laboratory                | Age group | No. of person participated in recording |    | Total Recordings |
|---------------------------|-----------|---|----|------------------|
|                           |           | F                                       | M  |                  |
| Lab1<br>(H*W*L= 10*8*8)   | 10 to 15  | 11                                      | 8  | 429              |
|                           | 20 to 25  | 6                                       | 8  |                  |
|                           | 40 to 50  | 3                                       | 4  |                  |
| Lab2<br>(H*W*L= 10*20*10) | 10 to 15  | 9                                       | 7  | 652              |
|                           | 20 to 25  | 8                                       | 10 |                  |
|                           | 40 to 50  | 3                                       | 2  |                  |
| Lab3<br>(H*W*L= 10*15*20) | 10 to 15  | 9                                       | 7  | 479              |
|                           | 20 to 25  | 9                                       | 10 |                  |
|                           | 40 to 50  | 2                                       | 3  |                  |

## References

1. Albawi, S., Mohammed, T. A., Al-Zawi, S. Understanding of a Convolutional Neural Network International Conference on Engineering and Technology (ICET), Antalya, 2017, 1-6. <https://doi.org/10.1109/ICEngTechnol.2017.8308186>
2. Andrade, D. C., Leo, S., Viana, M., L., D., Bernkopf, C. A Neural Attention Model for Speech Command Recognition. Audio & speech processing (arXiv:[eess.AS]), 2018. <https://arxiv.org/pdf/1808.08929.pdf>.

3. Bahar, P., Zeyer, A., Schlüter, R., Ney, H. On using 2D Sequence-to-Sequence Models for Speech Recognition. *IEEE International Conference on Acoustic Speech and signal Processing (ICASSP)*, Brighton, United Kingdom, 2019, 5671-5675. <https://doi.org/10.1109/ICASSP.2019.8682155>
4. Bahdanau, D., Chorowski, J., Serdyuk, D., Brakel, P., Bengio, Y. End-to-End Attention-Based Large Vocabulary Speech Recognition. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Shanghai, 2016, 4945-4949. <https://doi.org/10.1109/ICASSP.2016.7472618>
5. Deng, L. Front-End, Back-End, and Hybrid Techniques for Noise-Robust Speech Recognition. In: Kolossa D., Hab-Umbach R. (Eds.) *Robust Speech Recognition of Uncertain or Missing Data*, Springer, Berlin, Heidelberg, 2011, ch. 4. [https://doi.org/10.1007/978-3-642-21317-5\\_4](https://doi.org/10.1007/978-3-642-21317-5_4)
6. Deng, L., Li, X. Machine Learning Paradigms for Speech Recognition: An Overview. *IEEE Transactions on Audio Speech and Language Processing*, 2013, 21(5), 1060-1089. <https://doi.org/10.1109/TASL.2013.2244083>
7. ETSI Standard Document, Speech Processing, Transmission and Quality Aspects (STQ); Distributed Speech Recognition, [https://www.etsi.org/deliver/etsi\\_es/202000\\_202099/202050/01.01.05\\_60/es\\_202050v010105p.pdf](https://www.etsi.org/deliver/etsi_es/202000_202099/202050/01.01.05_60/es_202050v010105p.pdf).
8. Fan, R., Liu, G. CNN-Based Audio Front End Processing on Speech Recognition. *International Conference on Audio, Language and Image Processing (ICALIP)*, Shanghai, 2018, 349-354, <https://doi.org/10.1109/ICALIP.2018.8455731>
9. Fluent Speech Commands Dataset: <https://fluent.ai/fluent-speech-cmmands-a-dataset-for-spoken-language-understanding-research/>
10. Foster, D. *Generative Deep Learning*. O'Reilly media Inc., 2019.
11. Gangi, M. A. D., Negri, M., Cattoni, R., Dessi, R., Turchi, M. Enhancing Transformer for End-to-end Speech-to-Text Translation. *Proceedings of Machine Translation Summit XVII, Research Track*, Dublin, Ireland, 2019, 1, 23-31. <https://www.aclweb.org/anthology/W19-6603.pdf>.
12. Guiming, D., Xia, W., Guangyan, W., Yan, Z., Dan, L. Speech Recognition Based on Convolutional Neural Networks. *IEEE International Conference on Signal and Image Processing (ICSIP)*, Beijing, China, 2016, 708-711. <https://doi.org/10.1109/SIPROCESS.2016.7888355>
13. Hamid, O. A., Deng, L., Yu, D. Exploring Convolutional Neural Network Structures and Optimization Techniques for Speech Recognition. *Conference Interspeech-2013*, France, 2013, 10-22. <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.703.648&rep=rep1&type=pdf>.
14. Hamid, O. A., Mohamed, A., Jiang, H., Deng, L., Penn, G., Yu, D. Convolutional Neural Network for Speech Recognition. *IEEE/ACM Transactions on audio speech and language Processing*, 2014, 22(10), 1533-1545. <https://doi.org/10.1109/TASLP.2014.2339736>
15. Hatcher, W. G., Yu, W. A Survey of Deep learning Platforms, Applications and Emerging Research Trends. *IEEE Access*, 2018, 6, 24411-24432. <https://doi.org/10.1109/ACCESS.2018.2830661>
16. Hori, T., Kubo, Y., Nakamura, A. Real-Time One-Pass Decoding with Recurrent Neural Network Language Model for Speech Recognition. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Florence, 2014, 6364-6368, <https://doi.org/10.1109/ICASSP.2014.6854829>
17. Huei, Y. C. Benefits and Introduction to Python Programming for Fresh More Students Using Inexpensive Robots. *IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE)*, Wellington, 2014, 12-17. <https://doi.org/10.1109/TALE.2014.7062611>
18. Jakhar, K., Hooda, N. Big Data Deep Learning Framework using Keras: A Case Study of Pneumonia Prediction. *4th International Conference on Computing Communication and Automation (ICCCA)*, Greater Noida, India, 2018, 1-5. <https://doi.org/10.1109/CCAA.2018.8777571>
19. Khan, M. A., Ashraf, I., Alhaisoni, M., Damaševičius, R., Scherer, R., Rehman, A., Bukhari, S. A. C. Multimodal Brain Tumor Classification Using Deep Learning and Robust Feature Selection: A Machine Learning Application for Radiologists. *Diagnostics*, 2020, 10, 565. <https://doi.org/10.3390/diagnostics10080565>
20. Kim, S., Hori T., Watanabe, S. Joint CTC- Attention Based End-to-End Speech Recognition Using Multi-Task Learning. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, New Orleans, LA, 2017, 4835-4839. <https://doi.org/10.1109/ICASSP.2017.7953075>
21. Lauraitis, A., Maskeliūnas, R., Damaševičius, R., Krilavičius, T. Detection of Speech Impairments Using Cepstrum Auditory Spectrogram and Wavelet Time Scattering Domain Features. *IEEE Access*, 2020, 8, 96162-96172. <https://doi.org/10.1109/ACCESS.2020.2995737>

22. Lawrence, R. R., Juang, B. H. *Fundamentals of Speech Recognition*. Englewood Cliffs: PTR Prentice Hall, 1993.
23. Li, J., Zhao, R., Hu, H., Gong, Y. Improving RNN Transducer Modeling For End-to-End Speech Recognition. *Speech and Language Group, Microsoft*. <https://www.microsoft.com/en-us/research/uploads/prod/2019/10/RNNT.pdf>. <https://doi.org/10.1109/ASRU46091.2019.9003906>
24. Li, K., Li, J., Ye, G., Zhao, R., Gong, Y. Towards Code-Switching ASR for End-to-End CTC Models. *IEEE International Conference on Acoustic Speech and Signal Processing (ICASSP), Brighton, United Kingdom, 2019, 6076-6080*. <https://doi.org/10.1109/ICASSP.2019.8683223>
25. Ling, Z. An Acoustic Model for English Speech Recognition Based on Deep Learning. *11th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA), Qiqihar, China, 2019, 610-614*. <https://doi.org/10.1109/ICMTMA.2019.00140>
26. Lugosch, L., Ravanelli, M., Ignoto, P., Tomar, V. S., Bengio, Y. Speech Model Pre-training for End-to-End Spoken Language Understanding. *Conference Interspeech-2019, 2019, Graz, Australia, 45-56*. <https://arxiv.org/pdf/1904.03670.pdf>. <https://doi.org/10.21437/Interspeech.2019-2396>
27. Manasa, C. S., Priya, K. J., Gupta, D. Comparison of Acoustical Models of GMM-HMM Based for Speech Recognition in Hindi using PocketSphinx. *3rd International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 2019, 534-539*. <https://doi.org/10.1109/ICCMC.2019.8819747>
28. Miao, Y., Gowayyed, M., Na, X., Ko, T., Metze, F., Waibel, A. An Empirical Exploration of CTC Acoustic Models. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Shanghai, 2016, 2623-2627*. <https://doi.org/10.1109/ICASSP.2016.7472152>
29. Mukherjee, R., Goyal, P., Banerjee, D., Dey, K., Goyal, P. Convolutional Recurrent Neural Network Based Approach for Making Sense of Sounds Data Challenge. *Technical Report Published in Making Sense of Sounds Data Challenge 2018, 54-61*. [https://cvssp.org/projects/making\\_sense\\_of\\_sounds/site/assets/challenge\\_abstracts\\_and\\_figures/Rajdeep\\_Mukherjee.pdf](https://cvssp.org/projects/making_sense_of_sounds/site/assets/challenge_abstracts_and_figures/Rajdeep_Mukherjee.pdf)
30. Sahlol, A. T., Abdelaziz, M., Tariq Jamal, A., Damaševićus, R., Farouk Hassan, O. A Novel Method for Detection of Tuberculosis in Chest Radiographs Using Artificial Ecosystem-Based Optimization of Deep Neural Network Features. *Symmetry, 2020, 12(7), 1146*. <https://doi.org/10.3390/sym12071146>
31. Sainath, T. N., Pang, R., Rvbach, D., He, Y., Prabhakar, R., Li, W., Visontai, M., Liang, Q., Strohmaier, T., Wu, Y., Mcgraw, I., Chiu, C. C. Two Pass End to End Speech Recognition. *Interspeech, Graz, Australia, 2019, 23-45*. <https://doi.org/10.21437/Interspeech.2019-1341>
32. Sainath, T.N., Kingsbury, B., Soltau, H., Ramabhadran, B. Optimization Techniques to Improve Training Speed of Deep Neural Networks for Large Speech Tasks. *IEEE Transactions on Audio, Speech, and Language Processing, 2013, 21(11), 2267-2276*. <https://doi.org/10.1109/TASL.2013.2284378>
33. Speech Command Data Set: <https://ai.googleblog.com/2017/08/launching-speech-commands-dataset.html>
34. Takashima, R., Sheng, L., Kawai, H. Investigation of Sequence-Level Knowledge Distillation Methods for CTC Acoustic Models. *IEEE International Conference on Acoustic Speech and Signal Processing (ICASSP), Brighton, United Kingdom, 2019, 6156-6160*. <https://doi.org/10.1109/ICASSP.2019.8682671>
35. Ueno, S., Inaguma, H., Mimura, M., Kawahara, T. Acoustic-to-Word Attention-Based model Complemented with Character-Level CTC-Based Model. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, 2018, 5804-5808*. <https://doi.org/10.1109/ICASSP.2018.8462576>
36. Wang, D., Gong, C., Liu, Q. Improving Neural Language Modeling via Adversarial Training. *Proceedings of the 36th International Conference on Machine Learning, CA, USA, 2019, 367-375*. <https://arxiv.org/pdf/1906.03805.pdf>
37. Wang, G., Zhang, W. An RNN and CRNN Based Approach to Robust Voice Activity Detection. *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPAASC), Lanzhou, China, 2019, 1347-1350*. <https://doi.org/10.1109/APSIPAASC47483.2019.9023320>
38. Woo, H., Kim, N., Lee, J. Deep Neural Network Based Self-training Based on Unsupervised Learning and Dropout. *International Journal of Fuzzy Logic and Intelligent Systems, 2017, 17(1), 1-9*. <https://doi.org/10.5391/IJFIS.2017.17.1.1>

