Vellore institute of technology

Review 3 final document

NAME-Rajat gupta

REG NO-18BIT0006

SLOT -G1

REVIEW 3 (COMPARATIVE ANALYSIS)

link of video of integrated project

https://youtu.be/z0bK9h3Qz10

link of video of individual project

https://youtu.be/xm8JdADiL10

github link of all the coding

https://github.com/rajatgupta11jan2000/NASSCOM-PROJECT

MY MODEL IS CNN MODEL

In [21]:
```
from keras.layers import Dense, Dropout, Flatten, Conv1D, Input, MaxPooling1D
from keras.models import Model
from keras.callbacks import EarlyStopping, ModelCheckpoint
from keras import backend as K
K.clear_session()

def build_model():

    inputs = Input(shape=(8000,1))

    #First Conv1D Layer
    conv = Conv1D(8,13, padding='valid', activation='relu', strides=1)(inputs)
    conv = MaxPooling1D(3)(conv)
    conv = Dropout(0.3)(conv)

    #Second Conv1D Layer
    conv = Conv1D(16, 11, padding='valid', activation='relu', strides=1)(conv)
    conv = MaxPooling1D(3)(conv)
    conv = Dropout(0.3)(conv)

    #Third Conv1D Layer
    conv = Conv1D(32, 9, padding='valid', activation='relu', strides=1)(conv)
    conv = MaxPooling1D(3)(conv)
    conv = Dropout(0.3)(conv)

    #Fourth Conv1D Layer
    conv = Conv1D(64, 7, padding='valid', activation='relu', strides=1)(conv)
    conv = MaxPooling1D(3)(conv)
    conv = Dropout(0.3)(conv)

    #Flatten Layer
    conv = Flatten()(conv)

    #Dense Layer 1
    conv = Dense(256, activation='relu')(conv)
    conv = Dropout(0.3)(conv)

    #Dense Layer 2
    conv = Dense(128, activation='relu')(conv)
    conv = Dropout(0.3)(conv)

    outputs = Dense(len(labels), activation='softmax')(conv)

    model = Model(inputs, outputs)
    model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
    model.summary()
    return model
```

Comparative Results with other models in literature survey

In this model, a number of hyperparameters are used to specify the structure of the network. The number of hidden units in each hidden layer is very important; hence, it is taken as hyperparameter. win represents the time span of input speech signal. kW represents the kernel and temporal window width. dW represents the shift of temporal window. kWmp represents max-pooling kernel width and dWmp represents the shift of max-pooling kernel. The value of all hyperparameters is estimated during training based on frame-level classification accuracy on validation data. The range of hyperparameters after validation is shown in Table 1.

Hyperparameter	units	Range
Input window size (win)	ms	100-700
Kernel width of the first ConvNet layer (kW1)	samples	10-90
Kernel width of the n^{th} ConvNet layer (kWn)	samples	1-11
Number of filters per kernel (dout)	Filters	20-100
Max-pooling kernel width (kWmp)	Frames	2-6
Number of hidden units in the classifier	Units	100-1500

Table 1.

Range of hyperparameter for TIMIT dataset during validation.

The experiments are conducted for three convolutional layers. The speech window size (win is taken 250 ms with a shift of temporal window (dW) 10 ms. Table 2 shows the comparison of existing end-to-end speech recognition model in the context of PER. The results of the experiments conducted on TIMIT dataset for this model are compared with already existing techniques, and it is shown in Table 3. The main advantages of this model are that it uses only few parameters and offers better performance. It also increases the generalization capability of the classifiers.

Wer% and accuracy in my project

```
In [55]: accuracy=accuracy_score(ytest,ypred)
         accuracy

Out[55]: 0.7959183673469388
```

```
In [57]: wer=1-accuracy
         print(wer)

         0.20408163265306123
```

End-to-end speech recognition model	**WER (%)**
CNN-based speech recognition system using raw speech as input	33.2
Estimating phoneme class conditional probabilities from raw speech signal using convolutional neural networks .	32.4
Convolutional neural network-based continuous speech recognition using raw speech signal	33.2
End-to-end phoneme sequence recognition using convolutional neural networks	27.2
CNN-based direct raw speech model	20.4
End-to-end continuous speech recognition using attention-based recurrent NN: First results	18.57
Toward end-to-end speech recognition with deep convolutional neural networks	18.2
Attention-based models for speech recognition	17.6
Segmental recurrent neural networks for end-to-end speech recognition	17.3

Table 2.

Comparison of existing end-to-end speech model in the context of WER (%).

Bold value and text represent the performance of the CNN-based direct raw speech model.

Methods	**WER (%)**
GMM-/HMM-based ASR system	34
CNN-based direct raw speech model	20.4
Attention-based models for speech recognition	17.6
Segmental recurrent neural networks for end-to-end speech recognition	17.3
Combining time and frequency domain convolution in convolutional neural network-Based phone recognition	16.7
Phone recognition with hierarchical convolutional deep maxout networks	16.5

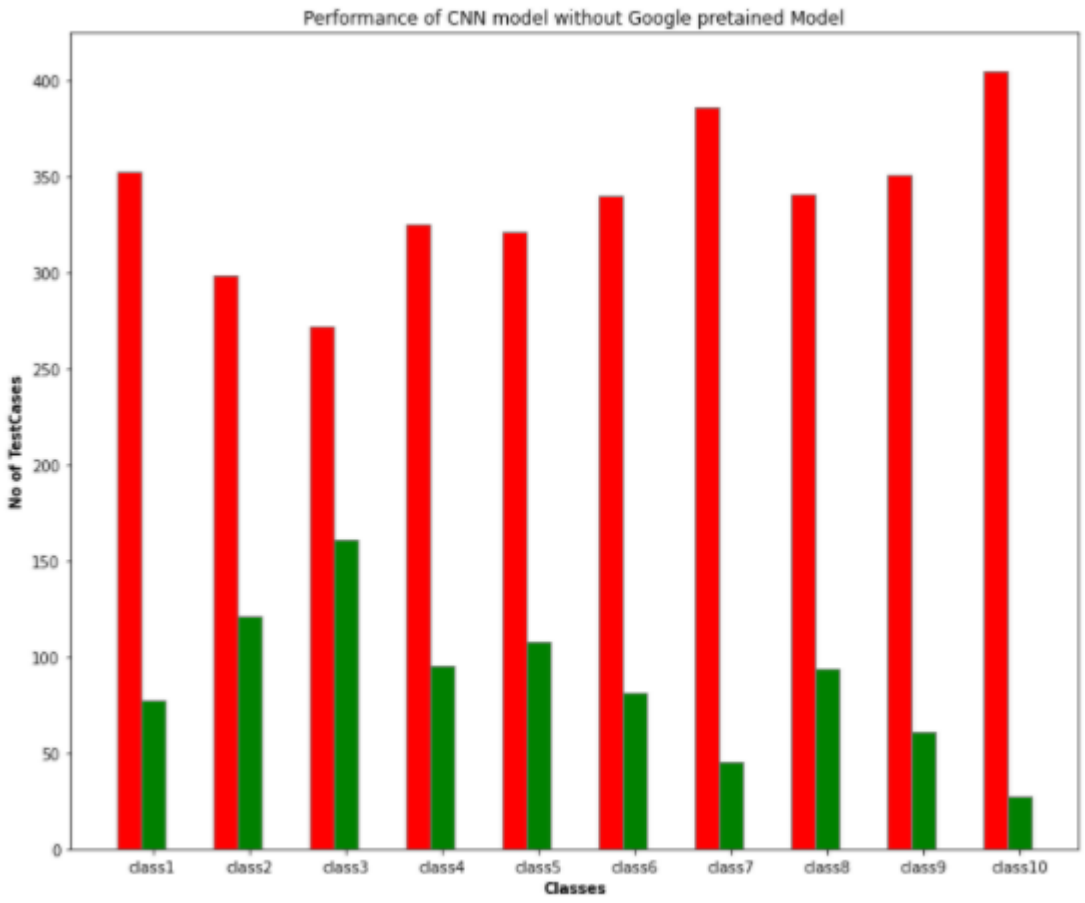
Table 3.

Comparison of existing techniques with CNN-based direct raw speech model in the context of WER (%).

Bold value and text represent the performance of the CNN-based direct raw speech model.

Classification report Comparison using Bar Graph

(Red bar-> total test cases, Green bar-> Correctly Classified test cases)



Screenshots of result I got with different performance matrix

In [37]:
```
from sklearn.metrics import confusion_matrix
cf=[]
cf=confusion_matrix(ytest,ypred)
cf
```

Out[37]:
```
array([[353,  26,   0,  18,   1,   2,   3,   5,   9,  13],
       [ 37, 299,   3,  36,   2,   3,   2,  10,  15,  13],
       [  1,   2, 272,   4,   1,   1,  21,   1,   8, 122],
       [ 15,  38,   2, 325,   2,   0,   9,   2,   7,  20],
       [  1,   8,   4,   0, 321,  21,   6,   5,  59,   4],
       [  4,   6,   0,   1,  23, 340,  17,   3,  25,   2],
       [  1,   0,  12,   2,   0,   1, 386,   1,  12,  16],
       [ 18,  31,   3,   1,   8,   1,   3, 341,  25,   4],
       [  1,  14,   3,   1,  20,   2,   8,   6, 351,   6],
       [  1,   3,   6,   3,   1,   0,   2,   1,  10, 405]], dtype=int64)
```

In [51]:
```
correct=[]
wrong=[]
for i in range(len(cf)):
    sum=0
    temp=cf[i][i]
    for j in range(len(cf[i])):
        sum=sum+cf[i][j]
    wrong.append(sum-temp)
    correct.append(temp)
correct
```

Out[51]:
```
[353, 299, 272, 325, 321, 340, 386, 341, 351, 405]
```

In [52]:
```
wrong
```

Out[52]:
```
[77, 121, 161, 95, 108, 81, 45, 94, 61, 27]
```

In [40]:
```
from sklearn.metrics import f1_score
f1_score(ytest,ypred,average='weighted')
```

Out[40]:
```
0.7964395847250307
```

In [41]:
```
from sklearn.metrics import precision_score,recall_score,confusion_matrix,classification_report,accuracy_score,f1_score
precision_score(ytest,ypred,average='weighted')
```

Out[41]:
```
0.8107543646617614
```

In [55]:
```
accuracy=accuracy_score(ytest,ypred)
accuracy
```

Out[55]:
```
0.7959183673469388
```

In [43]:
```
recall_score(ytest,ypred,average='weighted')
```

Out[43]:
```
0.7959183673469388
```

Comparision analysis on the basis of theory between all models

Related work

Traditional ASR system leveraged the GMM/HMM paradigm for acoustic modeling. GMM efficiently processes the vectors of input features and estimates emission probabilities for each HMM state. HMM efficiently normalizes the temporal variability present in speech signal. The combination of HMM and language model is used to estimate the most likely sequence of phones. The discriminative objective function is used to improve the recognition rate of the system by the discriminatively fine-tuned methods [8]. However, GMM has a shortcoming as it shows inability to model the data that is present on the boundary line. Artificial neural networks (ANNs) can learn much better models of data lying on the boundary condition. Deep neural networks (DNNs) as acoustic models tremendously improved the performance of ASR systems [9, 10, 11]. Generally, discriminative power of DNN is used for phoneme recognition and, for decoding task, HMM is preferred choice. DNNs have many hidden layers with a large number of nonlinear units and produce a very large number of outputs. The benefit of this large output layer is that it accommodates the large number of HMM states. DNN architectures have densely connected layers. Therefore, such architectures are more prone to overfitting. Secondly, features having the local correlations become difficult to learn for such architectures. In [12], speech frames are classified into clustered context-dependent states using DNNs. In [13, 14], GMM-free DNN training process is proposed by the researchers. However, GMM-free process demands iterative procedures like decision trees, generating forced alignments. DNN-based acoustic models are gaining much popularity in large vocabulary speech recognition task [10], but components like HMM and n-gram language model are same as in their predecessors.

GMM or DNN-based ASR systems perform the task in three steps: feature extraction, classification, and decoding. It is shown in Figure 1. Firstly, the short-term signal stst is processed at time “t” to extract the features xtxt . These features are provided as input to GMM or DNN acoustic model which estimates the class conditional probabilities Pe(i|xi)Peixi for each phone class i∈{1,…,I}.i∈1…I. The emission probabilities are as follows:

$$pe(xt|i) \propto p(xt|i)p(xt) = P(i|xt)p(i) \forall i \in i,\ldots,I \qquad E1$$

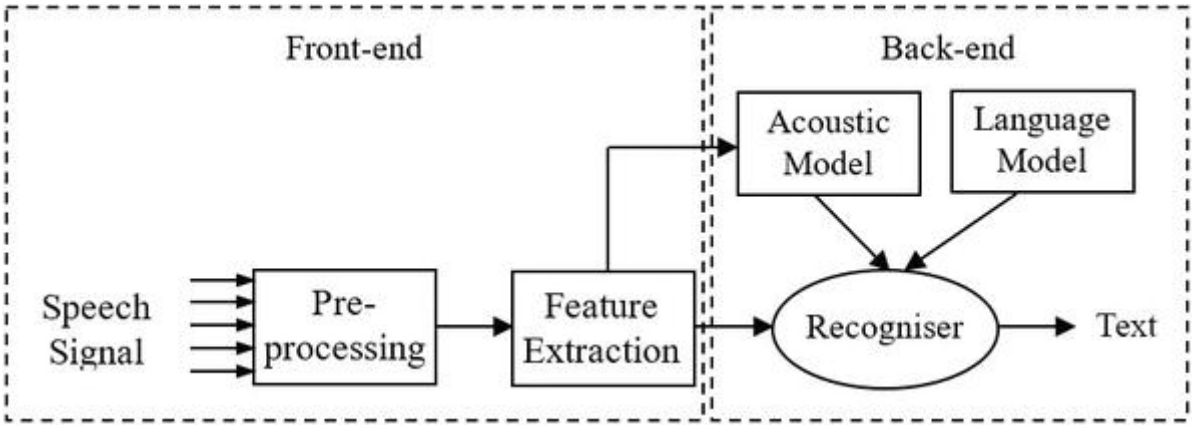


Figure 1.

General framework of automatic speech recognition system.

The prior class probability p(i)pi is computed by counting on the training set.

DNN is a feed-forward NN containing multiple hidden layers with a large number of hidden units. DNNs are trained using the back-propagation methods then discriminatively fine-tuned for reducing the gap between the desired output and actual output. DNN-/HMM-based hybrid systems are the effective models which use a tri-phone HMM model and an n-gram language model [10, 15]. Traditional DNN/HMM hybrid systems have several independent components that are trained separately like an acoustic model, pronunciation model, and language model. In the hybrid model, the speech recognition task is factorized into several independent subtasks. Each subtask is independently handled by a separate module which simplifies the objective. The classification task is much simpler in HMM-based models as compared to classifying the set of variable-length sequences directly. Figure 2 shows the hybrid DNN/HMM phoneme recognition model.

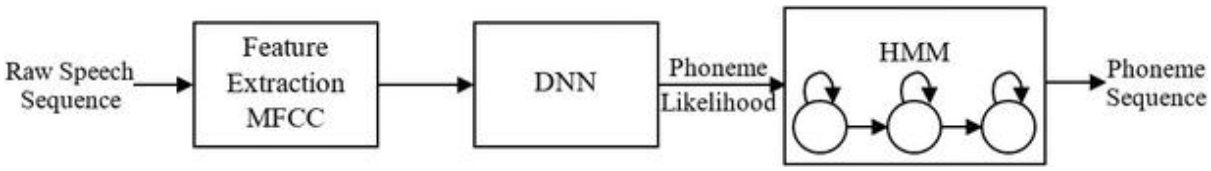


Figure 2.

Hybrid DNN/HMM phoneme recognition.

On the other side, researchers proposed end-to-end ASR systems that directly map the speech into labels without any intermediate components. As the advancements in deep learning, it has become possible to train the system in an end-to-end fashion. The high success rate of deep learning methods in vision task motivates the researchers to focus on classifier step for speech recognition. Such architectures are called deep because they are composed of many layers as compared to classical “shallow” systems. The main goal of end-to-end ASR system

is to simplify the conventional module-based ASR system into a single deep learning framework. In earlier systems, divide and conquer approaches are used to optimize each step independently, whereas deep learning approaches have a single architecture that leads to more optimal system. End-to-end speech recognition systems directly map the speech to text without requiring predefined alignment between acoustic frame and characters [16, 17, 18, 19, 20, 21, 22, 23, 24]. These systems are generally divided into three broad categories: attention-based model [19, 20, 21, 22], connectionist temporal classification [16, 17, 18, 25], and CNN-based direct raw speech method [5, 6, 7, 26]. All these models have a capability to address the problem of variable-length input and output sequences.

. This model generates a character based on the inputs and history of the target character. The attention-based models use encoder-decoder architecture to perform the sequence mapping from speech feature sequences to text as shown in Figure 3. Its extension, i.e., attention-based recurrent networks, has also been successfully applied to speech recognition. In the noisy environment, these models' results are poor because the estimated alignment is easily corrupted by noise. Another issue with this model is that it is hard to train from scratch due to misalignment on longer input sequences. Sequence-to-sequence networks have also achieved many breakthroughs in speech recognition [20, 21, 22]. They can be divided into three modules: an encoding module that transforms sequences, attention module that estimates the alignment between the hidden vector and targets, and decoding module that generates the output sequence. To develop successful sequence-to-sequence model, the understanding and preventing limitations are required. The discriminative training is a different way of training that raises the performance of the system. It allows the model to focus on most informative features with the risk of overfitting.

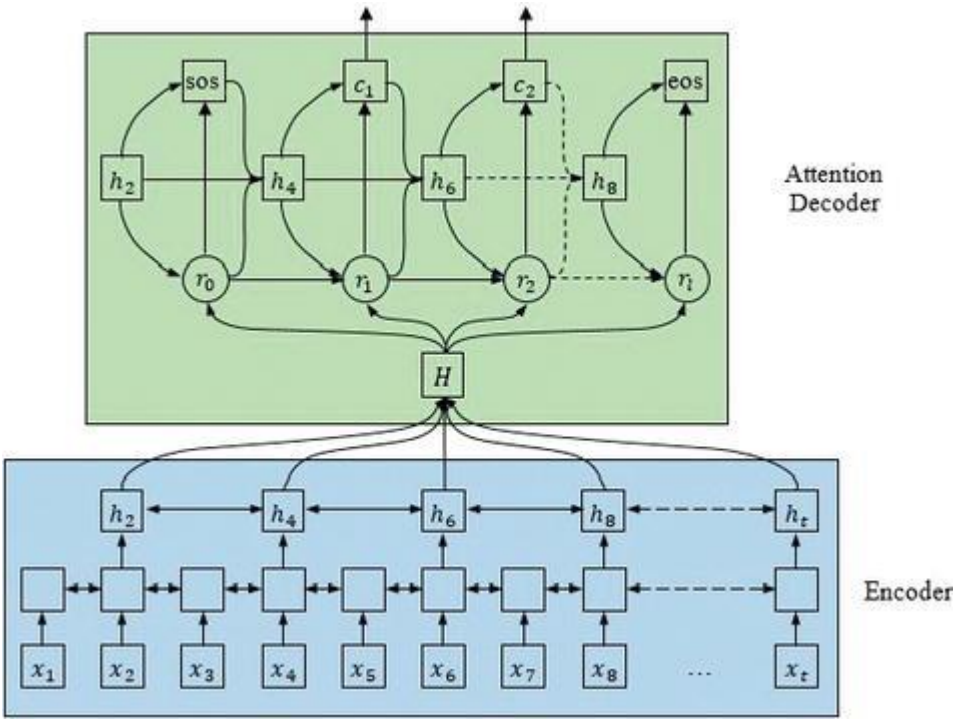


Figure 3.

Attention-based ASR model.

End-to-end trainable speech recognition systems are an important application of attention-based models. The decoder network computes a matching score between hidden states generated by the acoustic encoder network at each input time. It processes its hidden states to form a temporal alignment distribution. This matching score is used to estimate the corresponding encoder states. The difficulty of attention-based mechanism in speech recognition is that the feature inputs and corresponding letter outputs generally proceed in the same order with only small deviations within word. However, the different length of input and output sequences makes it more difficult to track the alignment. The advantage of attention-based mechanism is that any conditional independence assumptions (Markov assumption) are not required in this mechanism. Attention-based approach replaces the HMM with RNN to perform the sequence prediction. Attention mechanism automatically learns alignment between the input features and desired character sequence.

CTC techniques infer the speech-label alignment automatically. CTC [25] was developed for decoding the language. Firstly, Hannun et al. [17] used it for decoding purpose in Baidu's deep speech network. CTC uses dynamic programming [16] for efficient computation of a strictly monotonic alignment. However, graph-based decoding and language model are required for it. CTC approaches use RNN for feature extraction [28]. Graves et al. [30] used its objective function in deep bidirectional long short-term memory (LSTM) system. This model successfully arranges all possible alignments between input and output sequences during model training, not on the prior.

Two different versions of beam search are adopted by [16, 31] for decoding CTC models. Figure 4 shows the working architecture of the CTC model. In this, noisy and not informative frames are discarded by the introduction of the blank label which results in the optimal output sequence. CTC uses intermediate label representation to identify the blank labels, i.e., no output labels. CTC-based NN model shows high recognition rate for both phoneme recognition [32] and LVCSR [16, 31]. CTC-trained neural network with language model offers excellent results [17].

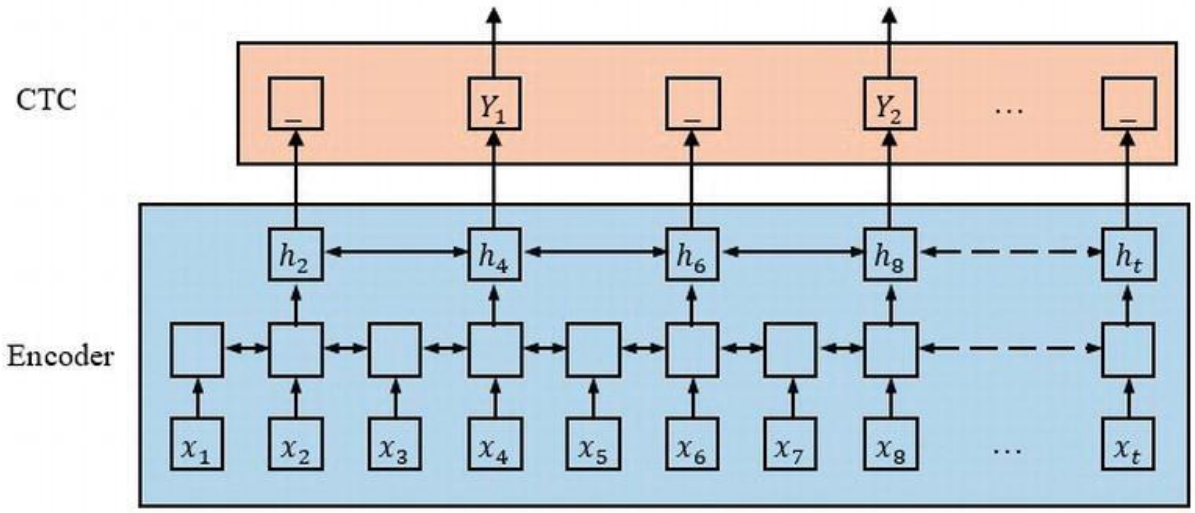


Figure 4.

CTC model for speech recognition.

End-to-end ASR systems perform well and achieve good results, yet they face two major challenges. First is how to incorporate lexicons and language models into decoding. However, [16, 31, 33] have incorporated lexicons for searching paths. Second, there is no shared experimental platform for the purpose of benchmark. End-to-end systems differ from the traditional system in both aspects: model architecture and decoding methods. Some efforts were also made to model the raw speech signal with little or no preprocessing [34]. Palaz et al. [6] showed in his study that CNN [35] can calculate the class conditional probabilities from raw speech signal as direct input. Therefore, CNNs are the preferred choice to learn features from the raw speech. Two stages of learned feature process are as follows: initially, features are learned by the filters at first convolutional layer, and then learned features are modeled by second and higher-level convolutional layers. An end-to-end phoneme sequence recognizer directly processes the raw speech signal as inputs and produces a phoneme sequence. The end-to-end system is composed of two parts: convolutional neural networks and conditional random field (CRF). CNN is used to perform the feature learning and classification, and CRFs are used for the decoding stage. CRF, ANN, multilayer perceptron, etc. have been successfully used as decoder. The results on TIMIT phone recognition task also confirm that the system effectively learns the features from raw speech and performs better than traditional systems that take cepstral features as input [36]. This model also produces good results for LVCSR [7].

4. Convolutional neural networks

CNNs are the popular variants of deep learning that are widely adopted in ASR systems. CNNs have many attractive advancements, i.e., weight sharing, convolutional filters, and pooling. Therefore, CNNs have achieved an impressive performance in ASR. CNNs are

composed of multiple convolutional layers. Figure 5 shows the block diagram of CNN. LeCun and Bengio [41] describe the three states of convolutional layer, i.e., convolution, pooling, and nonlinearity.

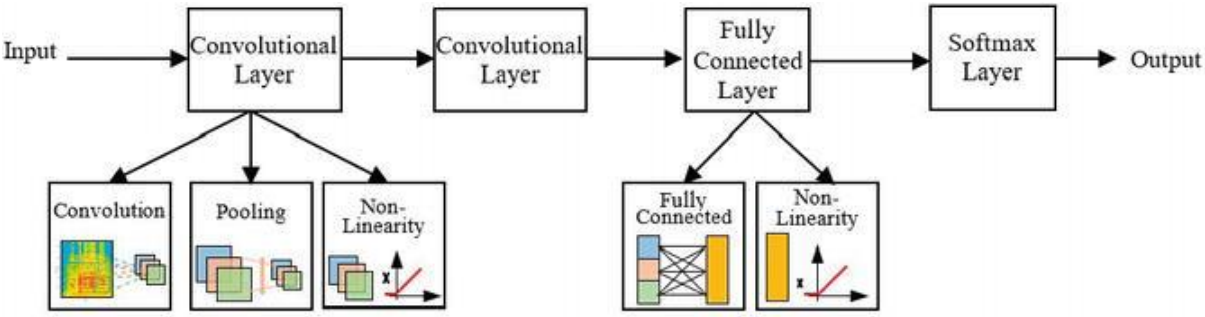


Figure 5.

Block diagram of convolutional neural network.

Deep CNNs set a new milestone by achieving approximate human level performance through advanced architectures and optimized training [42]. CNNs use nonlinear function to directly process the low-level data. CNNs are capable of learning high-level features with high complexity and abstraction. Pooling is the heart of CNNs that reduces the dimensionality of a feature map. Maxout is widely used nonlinearity and has shown its effectiveness in ASR tasks [43, 44].

Pooling is an important concept that transforms the joint feature representation into the valuable information by keeping the useful information and eliminating insignificant information. Small frequency shifts that are common in speech signal are efficiently handled using pooling. Pooling also helps in reducing the spectral variance present in the input speech. It maps the input from p adjacent units into the output by applying a special function. After the element-wise nonlinearities, the features are passed through pooling layer. This layer executes the downsampling on the feature maps coming from previous layer and produces the new feature maps with a condensed resolution. This layer drastically reduces the spatial dimension of input. It serves the two main purposes. The first is that the amount of parameters or weight is reduced by 65%, thus lessening the computational cost. The second is that it controls the overfitting. This term refers to when a model is so tuned to the training examples.

5. CNN-based end-to-end approach

A novel acoustic model based on CNN is proposed by Palaz et al. [5] which is shown in Figure 6. In this, raw speech signal is segmented into input speech signal sct={st−c,…,st,…,st+c}stc=st−c…st…st+c in the context of 2c frames having spanning window winwin milliseconds. First convolutional layer learns the useful features from the raw speech signal, and remaining convolutional layers further process these features into the useful information. After processing the speech signal, CNN estimates the class conditional probability, i.e., P(i/sct)Pi/stc , which is used to calculate emission scaled-likelihood

P(sct/i)Pstc/i . Several filter stages are present in the network before the classification stage. A filter stage is a combination of convolutional layer, pooling layer, and a nonlinearity. The joint training of feature stage and classifier stage is performed using the back-propagation algorithm.

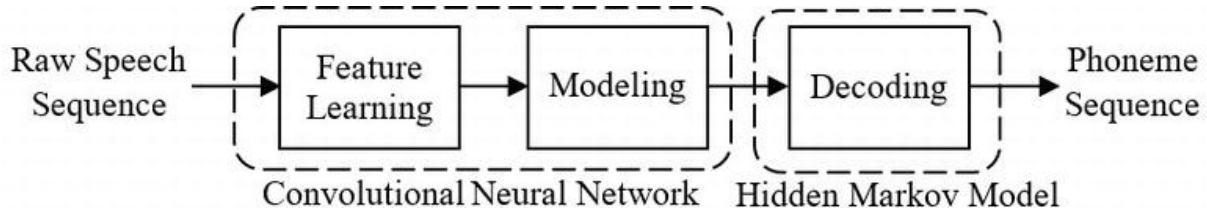


Figure 6.

CNN-based raw speech phoneme recognition system.

The end-to-end approach employs the following understanding:

1. Speech signals are non-stationary in nature. Therefore, they are processed in a short-term manner. Traditional feature extraction methods generally use 20–40 ms sliding window size. Although in the end-to-end approach, short-term processing of signal is required. Therefore, the size of the short-term window is taken as hyperparameter which is automatically determined during training.

2. Feature extraction is a filter operation because its components like Fourier transform, discrete cosine transform, etc. are filtering operations. In traditional systems, filtering is applied on both frequency and time. So, this factor is also considered in building convolutional layer in end-to-end system. Therefore, the number of filter banks and their parameters are taken as hyperparameters that are automatically determined during training.

3. The short-term processing of speech signal spread the information across time. In traditional systems, this spread information is modeled by calculating temporal derivatives and contextual information. Therefore, intermediate representation is supplied to classifier and calculated by taking long time span of input speech signal. Therefore, winwin , the size of input window, is taken as hyperparameter, which is estimated during training.

The end-to-end model estimates P(i/sct)Pi/stc by processing the speech signal with minimal assumptions or prior knowledge.