Journal of Ambient Intelligence and Humanized Computing (2020) 11:675–691
https://doi.org/10.1007/s12652-019-01325-y

ORIGINAL RESEARCH

A comparative analysis of pooling strategies for convolutional neural network based Hindi ASR

Vishal Passricha[1] · Rajesh Kumar Aggarwal[1]

Received: 18 December 2018 / Accepted: 16 May 2019 / Published online: 24 May 2019

Abstract
State-of-the-art speech recognition is witnessing its golden era as convolutional neural network (CNN) becomes the leader in this domain. CNN based acoustic models have been shown significant improvement in speech recognition tasks. This improvement is achieved due to the special components of CNN, i.e., local filters, weight sharing, and pooling. However, lack of core understanding renders this powerful model as a black-box machine. Although, CNN is performing well in speech recognition still further investigation will help in achieving better recognition rate. Pooling is a very important component of CNN that reduces the dimensionality of the feature-map and offers compact feature representation. Various pooling methods like max pooling, average pooling, stochastic pooling, mixed pooling, L_p pooling, multi-scale orderless pooling, and spectral pooling have their own advantages and disadvantages. In this paper, we deeply explore the state-of-the-art pooling for speech recognition tasks. This paper also helps to investigate that which pooling method performs well in which condition. This work explores different pooling methods for different architectures on Hindi speech dataset. The experimental results show that max pooling performs well when tested for clean speech and stochastic pooling works well in the noisy environment.

Keywords CNN · Max-pooling · Pooling · Speech recognition

1 Introduction

Convolutional neural network (CNN), an advanced version of the deep neural network (DNN), has been replaced the traditional models from the speech recognition task. The capability of CNN to handle local spatial correlations and local translational variance present in speech signals makes it a powerful acoustic model. CNN is also capable to handle the slight position shift in frequency. Local filters, weight sharing, and pooling are the important components of CNN. Pooling helps in reducing the connections and parameters of network and down-sample the dimensionality of the feature map. It also reduces the problem of overfitting. It is employed by most of the CNNs to design a condensed form of features.

CNN-based systems use pooling step to extract the features from different regions. Hubel and Wiesel (1962) gave an idea of feature pooling. Fukushima and Miyake (1982) firstly proposed a biologically-inspired model of image recognition using feature pooling which is the predecessor of CNN. Lecun et al. (1990, 1998) proposed the initial concept of CNN and later, the modern framework of CNN. Koenderink and Van Doorn (1999) implemented this idea locally on orderless images. Various pooling strategies use different formulas (like maximum, sum, average, and some other commutative combination rule) to extract the features from pooling region. Initially, Boureau and Cun (2008) applied max pooling strategy in CNN. After that, average pooling, stochastic pooing, L_p pooling, and mixed pooling were successfully introduced and involved in various classification activities. Some advanced pooling techniques like multi-scale orderless pooling, spatial pyramid pooling, and spectral pooling were also introduced and offered good classification rate in vision tasks.

Pooling operator is applied to extract lower and higher level features. It offers more robustness to noise and clutter. Pooling operators profoundly affect the performance in computer vision. However, these claims have so far been purely

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s12652-019-01325-y) contains supplementary material, which is available to authorized users.

✉ Rajesh Kumar Aggarwal
rka15969@gmail.com

[1] National Institute of Technology, Kurukshetra, India

empirical (Ma et al. 2015; Sze et al. 2017). In the research, it is observed that several compounding factors make the pooling strategies obscure. It seems to be challenging to find the best pooling technique. In this paper, various pooling strategies are compared for ASR task on Hindi speech corpora and their detailed analysis is covered.

The motivation behind this study is that the activations in the pooled maps are less sensitive to the precise locations of structures. In CNN, this pooled map is taken as input, and its features are extracted. These features are increasingly invariant to local transformations of the input. The pooling layer transforms this joint feature representation into a new more usable one. It discards the useless details and keeps only the important information. In CNN, from initial discovery to an exploration of architecture is also considered as the part of the research. The high popularity of CNN in object classification and recognition task has revived this research domain.

In this work, first, we investigate the various pooling strategies for multiple convolutional layers, i.e., different number of convolutional layers. In the second experiment, the number of hidden units in each fully connected layer is taken as hyperparameter and results are evaluated for different values of it. In the third experiment, the optimal value of pool-size is investigated for different pooling strategies. After performing the experiments as mentioned above, the architecture that contains four convolutional layers with max pooling having pool-size 2 and 3 fully connected layers having 1024 hidden units each is found as the best performing architecture. It offers minimum word error rate (WER) 16.3% on clean Hindi speech. After that various pooling strategies are investigated for different activation functions of fully connected layers then to overcome the problem of overfitting, popular regularization techniques are applied to our architecture. In last, the best performing architecture is tested in noisy conditions for different pooling strategies. The minimum WER 19.3% is achieved by stochastic pooling in noisy conditions on Hindi speech dataset.

The rest of this paper is structured as follows: In Sect. 2, the related work is covered. Sections 3 and 4 explore the CNN as an acoustic model and language modeling, respectively. A detailed exploration of various pooling techniques is given in Sect. 5. In Sect. 6, different activation functions which are used in fully connected layers are elaborated. Section 7 briefly explains the different regularization techniques. The experimental setup used for experiments is described in Sect. 8. The detail about all the experiments conducted for various pooling strategies and the comparisons of their results are presented in Sect. 9. Finally, Sect. 10 concludes this paper with a brief summary.

2 Related work

2.1 Hindi ASR

A large amount of available training and testing data is the backbone of any successful speech recognition system. However, Indian languages like Hindi, Bengali, Punjabi, etc. are still lacking in the state-of-the-art datasets. Currently, limited work has been done for Hindi language ASRs.

Aggarwal and Dave (2011) firstly applied discriminative training methods, i.e. maximum mutual information (MMI), minimum classification error (MCE), and minimum phoneme error (MPE) for the training of the Hindi ASR. They used only 500 words limited vocabulary and showed MPE is superior over MMI and MCE for clean speech. In next year, Aggarwal and Dave (2012a) optimized the central and side frequencies of MFCC filterbank using genetic algorithm and particle swarm optimization methods and claimed that optimized features perform better than conventional MFCC features. In the same year, Aggarwal and Dave (2012b) combined conventional, hybrid, and segmental HMM using the ROVER system combination technique. This combined model is evaluated with bigram and trigram language modeling on Hindi speech dataset and claimed 4% reduction in WER as compared to the traditional method. Aggarwal and Dave (2013) integrated the conventional feature extraction methods like MFCC, PLP, and gravity centroids to enhance the recognition rate of Hindi ASR. Biswas et al. (2014) used wavelet sub-band perceptual wiener filtering to extract the acoustic features. These features effectively recognized Hindi digit dataset. Biswas et al. (2016a) used wiener filtered wavelet packet decomposed periodic and aperiodic features for Hindi speech recognition. These features were found more effective than baseline features in phoneme recognition. As an extension, Biswas et al. (2016b) used harmonic energy features using ANOVA fusion technique for Hindi phoneme recognition. They achieved promising improvement over wavelet packet cepstral coefficients (WPCC) features. In the same year, Pasricha and Aggarwal (2016) introduced a hybrid architecture for Hindi robust speech recognition system and achieved excellent results in the noisy environment.

Dua et al. (2018a) improved the performance of Hindi ASR by optimizing the filterbank. They used the differential evolution (DE) algorithm for optimizing the number and spacing of filters of various feature extraction techniques. By experimental results, they observed that gammatone frequency cepstral coefficients (GFCC) features are the best. In the extension of their work, they trained Hindi ASR with GFCC features using discriminative training methods (Dua et al. 2018b). In this, feature extraction methods are optimized by DE optimization, and the acoustic model is discriminatively trained using MMI and MPE. Passricha and Aggarwal (2019) sandwiched the bi-directional long short term memory (BLSTM) layers between CNN and fully connected layers to improve the recognition rate of Hindi speech recognition system.

2.2 Bengali ASR

If we talk about Bengali ASRs, Das et al. (2011) proposed the first continuous ASR system for Bengali speech in 2011 and claimed 85.3% word recognition rate. In 2016, Bhowmik and Mandal (2016) extracted DNN-based phonological features from Bengali continuous speech and claimed the word recognition rate of 89.17%. Nahid et al. (2016) worked on Bengali real number dataset using CMU Sphinx and claimed 100% accuracy. Note that they all worked on the very small dataset. Reza et al. (2017) performed a comparative analysis of the effects of data augmentation on HMM-GMM and DNN classifiers on Prodorshok-I dataset, i.e. Bengali isolated word dataset. Nahid et al. (2017) proposed a double layered LSTM-RNN approach for Bengali speech recognition, and achieved 13.2% and 28.7% WER and phoneme error rate (PER), respectively.

3 CNN acoustic model

CNN has become a popular feature extractor applying to biomedical image classification (Nguyen et al. 2019), Skelton estimation (Zavala-Mondragon et al. 2019), speech recognition (Singhal et al. 2018), seizure prediction (Hu et al. 2019), Lung nodule detection (Feng et al. 2019), human action recognition (Imran and Raman 2019) and many more. CNN has been replaced the traditional models from the speech recognition task (Abdel-Hamid et al. 2012; Passricha and Aggarwal 2018). CNN based acoustic models have performed better than state-of-the-art DNN systems in ASR by selecting more adequate features for different classes like speaker, gender, and phone (Abdel-Hamid et al. 2012, 2013; Sainath et al. 2013a, b; Toth 2014b). Toth (2015) argues that CNN improves the mean of the per-speaker recognition rate and reduces their variance by ≈ 5.7% when compared to DNN. CNN can utilize the long-time dependencies among the speech frames by exploiting prior knowledge of speech signal. According to Soltau et al. (2013), CNN does not take any stress for semi-clean data in the robust ASR systems. Hence the performance of the system deteriorates. The main aim of CNN is to discover the local structure in input data. CNN successfully reduces the spectral variations and effectively models the spectral correlations in acoustic features.

The architecture of CNN for speech recognition is illustrated in Fig. 1. The CNN consists of a few pairs of convolution and pooling layers. CNN introduces three new concepts over the DNN: local filters, weight sharing, and pooling. Each one of the concepts has the potential to boost speech recognition performance. Different phonemes have different energy concentrations in different local bands along the frequency axis. The process of distinguishing different phonemes becomes critical due to these local energy concentrations. These small local energy concentrations of the input are processed by a set of local filters that are applied by a convolutional layer. Local or convolutional filters are imposed on a subset region of the previous layer to capture a specific kind of local patterns known as structural locality. These filters are also effective against ambient noises. Earlier, the convolution was applied only on frequency axis which made the ASR system more robust against the variations generated from different speaking style and speaker (Abdel-Hamid et al. 2012, 2013; Sainath et al. 2013b). Later, Toth (2014a) applied convolution along the time axis. Unfortunately, the results did not meet the expected level due to the loss of locality. Convolution is generally applied only along the frequency axis because shift-invariance in

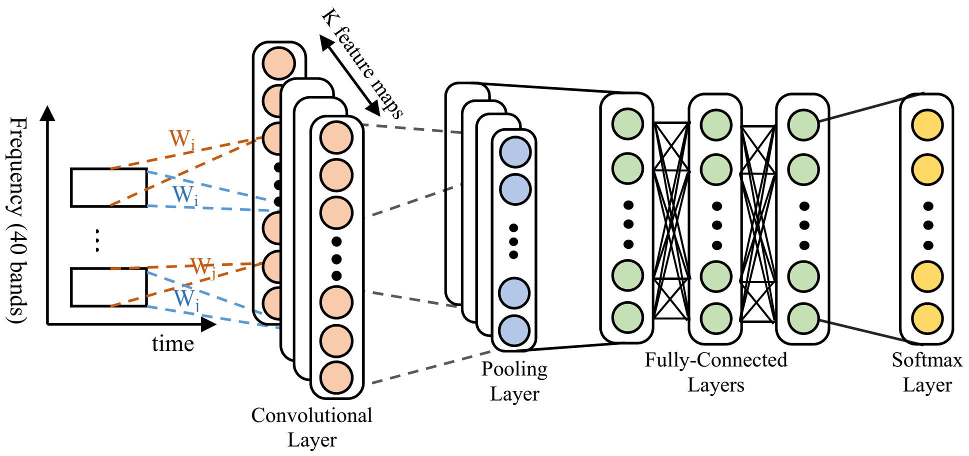

Fig. 1 Architecture of convolutional neural network for speech recognition

frequency is more important than shift-invariance in time (Abdel-Hamid et al. 2014).

Weight sharing is used with convolutional filters to reduce translational variance. Limited weight sharing (LWS) and full weight sharing (FWS) are convolutional weights that are applied to convolutional layers to utilize the characteristics of different spectral bands. They combine the information received from the neurons and this collected information is processed by activation function and passed to the next layer. Abdel-Hamid et al. (2012) favor LWS by presenting different spectral regions for different spectral phenomena. On the other hand, Sainath et al. (2013a) favor FWS by showing the same performance from FWS as LWS. FWS is generally applied on all neurons across all spectral regions which is much simpler. Pooling layer regularly follows the convolutional layer. Pooling provides additional translational and rotational invariances. It also provides a reduction in the number of parameters (Jarrett et al. 2009). It is generally applied along the frequency domain. Finally, fully connected layers combine inputs from all the positions into a 1-D feature vector. Therefore, high-level reasoning is applied to invariant features and then supplied to softmax, i.e., the final layer of the architecture. Finally, the softmax activation layer classifies the overall inputs. CNN classifies acoustic input features into corresponding classes. By using the training data, the prior probability of the considered state is calculated. CNN has the capabilities to model temporal correlations with adaptive content information. Hence it can perform any sequence-to-sequence mapping with high accuracy (Gehring et al. 2017).

4 Language modeling

Language model (LM) is a model of speech recognition system which predicts the next word based on the previous word. N-gram language model is widely used LM. The basic idea behind n-gram language modeling is to collect statistics about how frequent different n-gram are and use these to predict the next word. In simple words, an n-gram LM for the class token is generated, then probabilities for words in a class are distributed according to smoothed relative unigram frequencies of the words. However, n-gram LM have sparsity problem, in which we do not observe enough data to model the next output accurately (especially as n increases). Second, n-gram LM needs to keep track of all possible $n - 1$ histories, which is intractable in large vocabulary continuous speech recognition.

Recurrent neural network (RNN) language model is a good alternative of n-gram that solves the problem of data sparsity by representing words as vectors and uses them as inputs to the language model. Nowadays, it is highly used in ASR task. RNN remembers all the relationship while training itself means all the previous words helps to predict a better output. RNNLM represents the full, non-truncated history $h_1^{i-1} = \langle w_{i-1}, \ldots, w_1 \rangle$ for word w_i using the 1-of-k encoding of the most recent preceding word w_{i-1} and a continuous vector v_{i-2} is a vector of all ones. In this, RNN is used to compute LM probabilities $\mathrm{P}_{\mathrm{RNN}}(w_i | w_{i-1}, v_{i-2})$ consists of three layer.w_{i-1} and v_{i-2} are given as input to the input layer, by their concatenations the full history vector is predicted. These two inputs are compresses by hidden layer using sigmoid activation to compute a new feature representation v_{i-1}. This is then passed to the output layer to produce normalized RNNLM probabilities. The same output is recursively fed back into the input layer as the future remaining history to compute the LM probabilities for the next word $\mathrm{P}_{\mathrm{RNN}}(w_{i+1} | w_i, v_{i-1})$. As RNNLM use a vector representation of full histories, they are mostly used for N-best list rescoring. For more efficient lattice rescoring using RNNLMs, appropriate approximation schemes based on clustering among complete histories is used (Liu et al. 2014). The back-propagation through time (BPTT) is generally used to train RNNLM because the parameters are shared by all time steps in the network, the gradient at each output depends not only on the calculations of the current time step, but also the previous time steps. Here, we will use an LSTM-based RNN to construct a word-level language. State-of-the-art models of this type can require considerable computing resources and training time.

Besides using the WER to evaluate the quality of a language model, there is a more direct way to measure the probability of the testing word sequences through the language model, which is the perplexity. Perplexity is simply the cross-entropy between the model and the dataset:

$$PP(W) = 2^{H(W)}, \tag{1}$$

$$\text{where } H(v) = {}^{-1}\!/_{N_W} \log_2 P(W), \tag{2}$$

and N_W is the length of the testing word sequence. Perplexity can be roughly interpreted as the average branching factor of the testing data to the language model. Our aim should be minimizing the LM perplexity of the training data. As the lower the perplexity, the less branches the speech recognizer needs to consider during decoding. Best model is the one that best predicts an unseen test set, or assigns on average the probability to all sentences that is seen (Huang et al. 2001) (Fig. 2).

5 Type of pooling

CNN is successfully adopted in many speech recognition tasks and setting new records. Alternate convolution and pooling layers are followed in CNN architecture, and in

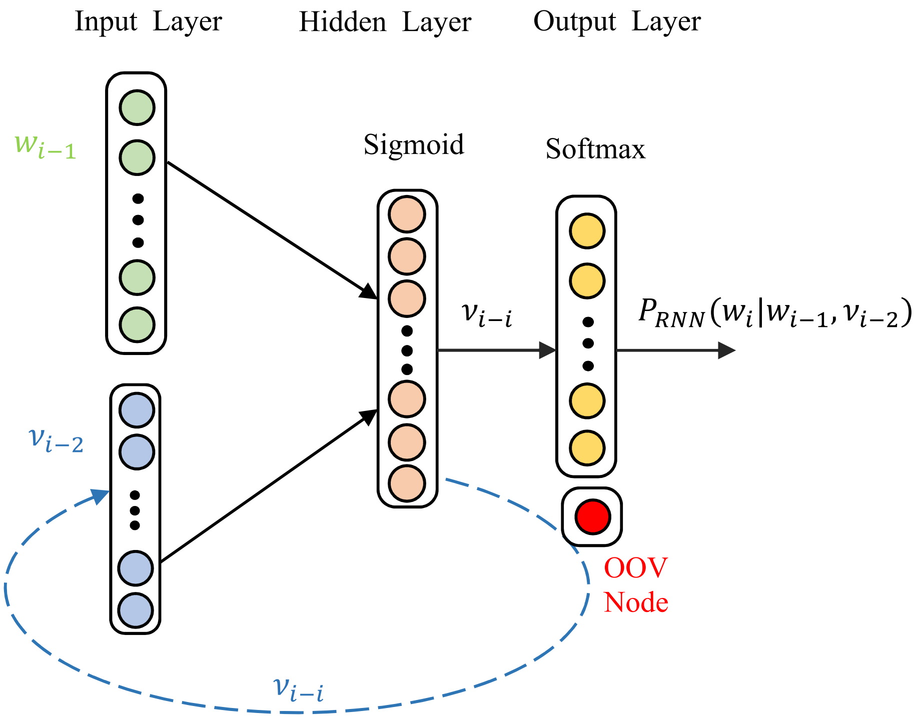

Fig. 2 An architecture of recurrent neural network language model for speech recognition

last fully connected layers are used. Pooling is a key-step in CNN-based ASR systems that reduces the dimensionality of the feature maps. It combines a set of values into a smaller number of values, i.e., the reduction in the dimensionality of the feature map. It transforms the joint feature representation into valuable information by keeping useful information and eliminating irrelevant 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. Stride represents the shifting unit of the filter that convolves around the input volume. In simple words, the stride is the amount by which the filter shifts.

Pooling operators provide a form of spatial transformation invariance as well as reduce the computational complexity for upper layers by eliminating some connections between convolutional layers. Pooling extracts the output from several regions and combines them. It maps the input from p adjacent units into the output by applying a unique function. This layer executes the down-sampling on the feature maps coming from the 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 it reduces the number of parameters or weights by 65%, thus lessening the computational cost. The second is that it controls the overfitting. An ideal pooling method is expected to extract only useful information and discards irrelevant details. This section covers earlier proposed pooling methods that have been successfully applied in CNN.

Researchers have proposed many pooling strategies. In the beginning, max and average pooling were popular choices (LeCun et al. 2004). Max and average pooling both are deterministic. Max pooling extracts the maximum activation in each pooling region, and average pooling takes the average of all activations in each pooling region. In average pooling, all the activations in a pooling region are taken into consideration with equal contributions. Average pooling may downplay the high activations as many low activations are also included. Stochastic pooling (Zeiler and Fergus 2013) and L_p pooling (Bruna et al. 2014) are alternative pooling strategies that address this problem. Bruna et al. (2014) claimed that the generalization offered by L_p pooling is better than max-pooling.

Stochastic pooling selects from a few specific locations (those with strong responses), rather than all possible locations in the region. It uses the stochastic process to select activation. It forms multinomial distribution by the activations and uses it to select the activation within the pooling region. Therefore, it is known as stochastic pooling. Mixed pooling (Yu et al. 2014) is a hybrid of max and average pooling. It switches between max-pooling and average pooling by parameter λ. It takes the advantages of both pooling but both are deterministic in nature hence it is also deterministic in nature. Multiscale orderless pooling is applied to improve the invariance of CNN (Gong et al. 2014). Spatial pyramid pooling extracts the features at the variable scales (He et al. 2014). It offers high scale invariances and reduces the overfitting. Spectral pooling is an advance pooling strategy that crops the features from the input stream (Rippel

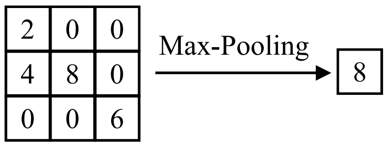

Fig. 3 An example of max-pooling

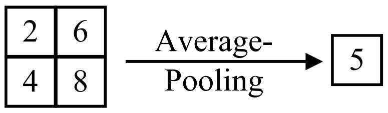

Fig. 4 An example of average pooling

et al. 2015). It overcomes the problem of a sharp reduction in dimensionality.

5.1 Max pooling

The most popular pooling strategy for CNN is max pooling which picks only the maximum activation and discards all other units from the pooling region (Boureau and Cun 2008). The function for max-pooling is given in Eq. (3):

$$s_j = \max_{i \in R_j} a_i, \tag{3}$$

where R_j is a pooling region and $\left\{ a_1, \ldots, a_{|R_j|} \right\}$ is a set of activations. Zeiler and Fergus (2013) have shown in their experiments that overfitting of the training data is a major problem with max pooling (Fig. 3).

5.2 Average pooling

The max pooling selects maximum activation of the filter template from each region. However other activations in the same pooling region are ignored, this should be taken into account. Instead of always extracting maximum activation from each pooling region as max pooling does, the average pooling takes the arithmetic mean of the elements in each pooling region. The function for average pooling is given in Eq. (4):

$$s_j = \frac{1}{|R_j|} \sum_{i \in R_j} a_i. \tag{4}$$

The drawback of average pooling is that all the elements in a pooling region are considered, even if many have low magnitude. It down-weighs the strong activations because it combines many sparse elements in average (Fig. 4).

5.3 Stochastic pooling

Inspired by the dropout (Hinton et al. 2012), Zeiler and Fergus (2013) proposed the idea of stochastic pooling. In max pooling, the maximum activation is picked from each pooling region. Whereas the areas of high activation are down-weighted by areas of low-activation in average pooling because all elements in the pooling region are examined, and their average is taken. It is a major problem with average pooling. The issues of max and average pooling are addressed using stochastic pooling. Stochastic pooling applies multinomial distribution to pick the value randomly. It includes the non-maximal activations of the feature map. In stochastic pooling, first, the probabilities p_i is computed for each region j by normalizing the activations within the regions, as given in Eq. (5):

$$p_i = a_i \Big/ \textstyle\sum_{k \in R_j} a_k. \tag{5}$$

These probabilities create a multinomial distribution that is used to select location l and corresponding pooled activation a_l based on p. Multinomial distribution selects a location l within the region:

$$s_j = a_l \, where \quad l \sim P\left(p_1, \ldots, p_{|R_j|}\right).$$

In simple words, the activations are selected based on the probabilities calculated by multinomial distribution. In this, all activations get the chances according to their probability proportionate. Stochastic pooling prohibits

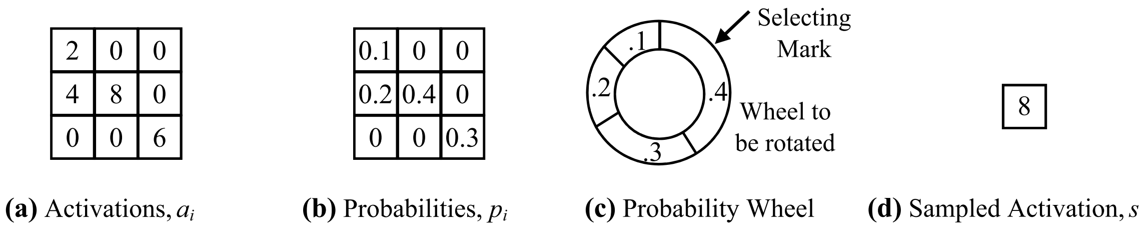

Fig. 5 Example of stochastic pooling **a** activations within a given pooling region, **b** probabilities based on activations, **c** probability wheel, **d** sampled activation

overfitting because of the stochastic component. Some advantages of max-pooling are also available in the stochastic pooling, and it also utilizes non-maximal activations. Its procedure is shown in Fig. 5.

It is to be noted that stochastic pooling represents the multinomial distribution of activations within the region; hence the selected element may or may not be the largest element. It gives high chances to stronger activations and suppresses the weaker activations.

5.4 L_p pooling

Sermanet et al. (2012) proposed the concept of L_p pooling and claimed that its generalization ability is better than max pooling. In this pooling, a weighted average of inputs is taken in pooling region. It is represented as given below:

$$s_j = \left(\frac{1}{\left|R_j\right|} \sum_{i \in R_j} a_i^p \right)^{1/p}, \tag{6}$$

where s_j represents the output of the pooling operator at location j, a_i is the feature value at location i within the pooling region R_j. The value of p varies between 1 and ∞. When $p = 1$, L_p operator behaves as average pooling and at $p = \infty$ it leads to max-pooling. For L_p pooling, $p > 1$ is examined as a trade-off between average and max pooling.

5.5 Mixed pooling

Max pooling extracts only the maximum activation whereas average pooling down-weighs the activation by combining the non-maximal activations. To overcome this problem, Yu et al. (2014) proposed a hybrid approach by combining the average pooling and max pooling. This approach is highly inspired by dropout (Hinton et al. 2012) and Dropconnect (Wan et al. 2013). Mixed pooling can be represented as:

$$s_j = \lambda \max_{i \in R_j} a_i + (1 - \lambda) \frac{1}{|R_j|} \sum_{i \in R_j} a_i, \tag{7}$$

where λ decides the choice of either using max pooling or average pooling. The value of λ is selected randomly either 0 or 1. When $\lambda = 0$, it behaves like average pooling and when $\lambda = 1$, it works like max pooling. The value of λ is recorded for forward-propagation order and it is used during backpropagation process. Yu et al. (2014) showed its superiority over max and average pooling by performing image classification on three different datasets.

5.6 Multi-scale orderless pooling

Multi-scale orderless pooling (MOP) was proposed by Gong et al. (2014). This pooling scheme improves the invariance of CNNs without affecting their discriminative power. MOP processes both the whole signal and local patches to extract the deep activation features. The activation features of the whole signal are captured for global spatial layout information and activation features of local patches are captured for more local, fine-grained details of the signal as well as enhancing invariance. Vectors of locally aggregated descriptors (VLAD) encoding (Jegou et al. 2012) is used to aggregate the activation features from local patches. Global spatial activation features and local patch activation features are concatenated using Karhunen–Loeve transformation (Dony 2001) to obtain the new signal representation.

5.7 Spatial pyramid pooling

He et al. (2014) proposed an idea of spatial pyramid pooling (SPP) which eliminates the problem of fixed-size input by generating a fixed-length representation regardless of input size. In this, feature maps of entire input are computed in one step and then feature from the arbitrary regions are pooled to generate fixed-length representation. This pooling layer is placed on top of the last convolutional layer. It feeds the fixed-length input to the fully connected layers. SPP extracts the feature at variable scales by offering the flexibility of input scales. It provides high scale-invariance and reduces overfitting.

5.8 Spectral pooling

Rippel et al. (2015) introduced a new pooling scheme by including an idea of dimensionality reduction by cropping the representation of the input in the frequency domain. Let $x \in R^{m \times m}$ be an input feature map and $h \times w$ be the desired dimensions of the output feature map. In this, firstly discrete Fourier transform (DFT) (Duhamel et al. 1988) is applied on the input feature map, then $h \times w$ size submatrix of frequency representation is cropped from the center. In last, inverse DFT is applied on $h \times w$ submatrix to convert it into spatial domain again. Spectral pooling preserves the more information for the same output dimensionality by applying linear low-pass filtering operation when compared to max pooling. It overcomes the problem of a sharp reduction in output map dimensionality. The key idea behind spectral pooling is matrix truncation, which reduces the computation cost in CNNs by employing fast Fourier transformation for convolutional kernels (Mathieu et al. 2014).

6 Activation functions

For a certain task, the performance of a CNN also depends upon the specific selection of proper activation function. This section covers a brief discussion about various activation functions which are used in CNNs.

6.1 ReLU

Rectified linear unit is the most commonly used activation function in deep learning models. ReLU is a non-saturated linear activation function. Its output is zero for negative values and input itself otherwise. ReLU function is defined as:

$$h_l = \max(0, z_l), \tag{8}$$

6.2 Leaky ReLU

ReLU unit has zero gradients when the unit is not active which is considered as its major disadvantage. Due to zero gradient, units become inactive; hence gradient-based optimization could not adjust their weights. As a result, it slows down the training process. To overcome this issue, Maas et al. (2013) proposed an improved version of ReLU known as Leaky ReLU (LReLU). Mathematically LReLU is defined as:

$$h_l = \max(z_l, 0) + \lambda \min(z_l, 0), \tag{9}$$

where λ is a predefined parameter in range (0,1). The LReLU sacrifies hard-zero sparsity and compresses the negative part rather than mapping it to zero which makes it allow for a small, non-zero gradient when the unit is not active.

6.3 Parametric ReLU (PReLU)

The limitation of LReLU is predefined parameter λ. To resolve this issue, He et al. (2015) introduced an improved version, parametric ReLU which adaptively learns the parameters of the rectifiers in order to improve the accuracy. Mathematical equation to represent PReLU is:

$$h_l = \max(z_l, 0) + \lambda_l \min(z_l, 0), \tag{10}$$

where λ_l is the learned parameter for the lth channel. As PReLU only introduces a very small number of extra parameters, e.g., the number of extra parameters is the same as the number of channels of the whole network. Therefore, there is no extra risk of overfitting and the extra computational cost is also negligible.

6.4 Randomized ReLU

Xu et al. (2015) introduced randomized leaky rectified linear unit (RReLU) as another advanced version of LReLU which incorporates non-zero slope for the negative part in rectified units and improves the results. In this, the parameters of negative components are randomly sampled by a uniform distribution during training and then fixed during testing. The mathematical equation for RReLU function is defined as:

$$h_l^{(n)} = \max\left(z_l^{(n)}, 0\right) + \lambda_l^{(n)} \min\left(z_l^{(n)}, 0\right), \tag{11}$$

where $z_l^{(n)}$ denotes the input of activation function on the lth channel of nth example, $\lambda_l^{(n)}$ denotes its corresponding sampled parameter, and $h_l^{(n)}$ denotes its corresponding output. Due to its randomized nature, it helps in reducing overfitting.

6.5 ELU

Exponential linear unit was proposed by Clevert et al. (2016). It offers improved learning characteristics for deep learning models and provides higher classification accuracies. During positive part, ELU avoids the vanishing gradient problem by setting the positive part to identity whereas for negative part, in contrast to ReLU, It has negative values which allow them to push mean unit activations closer to zero. It lowers the computational complexities which are beneficial for fast learning.

When compared to LReLU, PReLU, and RReLU which have unsaturated negative parts, ELU engages a saturation function as negative part. As the saturation function will decrease the variation of the units if deactivated, it makes ELU more robust to noise. Mathematically, ELU is represented as:

$$h_l = \max(z_l, 0) + \min(\lambda(e^{z_l} - 1), 0), \tag{12}$$

where λ is a predefined parameter for controlling the value to which an ELU saturate the negative inputs.

6.6 Maxout

Maxout (Goodfellow et al. 2013) nonlinearity is a special activation function which selects the maximum input from multiple inputs at each spatial position. Mathematically, maxout activation is defined as:

$$h_l = max_{l \in [1,K]} z_l, \tag{13}$$

where z_l represents the lth channel of the feature map. Moreover, maxout is particularly well-suited for training with dropout.

6.7 Probout

Maxout always selects the maximum response among inputs, so the inputs having less values never get the chance. To overcome this issue, Springenberg and Riedmiller (2013) replaced the maximum operation in maxout with a stochastic generalization of maxout called probout. Initially, the probability for each of the k linear units is calculated as:

$$p_i = e^{\lambda z_i} \Big/ \sum_{j=1}^{k} e^{\lambda z_j}, \tag{14}$$

where λ represents the hyperparameter for controlling the variance distribution. Then, according to a multinomial distribution $\{p_1, p_2, \dots, p_k\}$, one unit among k units is picked and the picked unit value is set as activation value. In order to incorporate dropout, the probabilities are redefined as:

$$\hat{p}_0 = 0.5, \quad \hat{p}_i = e^{\lambda z_i} \Big/ \left(2 \times \sum_{j=1}^{k} e^{\lambda z_j} \right). \tag{15}$$

The activation function is then sampled as:

$$a_i = \begin{cases} 0 & if\, i = 0 \\ z_i & else \end{cases}, \tag{16}$$

where $i \sim multinomial\,\{\hat{p}_0, \dots, \hat{p}_k\}$.

The main advantage of probout is that it efficiently balances between preserving the desirable properties of maxout units and improving their invariance properties. Besides, this advantage, it is computationally expensive than maxout due to some additional probability calculations.

7 Regularization techniques

Maxout neurons expertly manage the problem of underfitting, so they have better optimization performance (Toth 2014b). However, deep CNNs face the problem of overfitting which cannot be neglected. Although, overfitting can be efficiently encountered by regularization methods. This section covers the brief history of some effective regularization techniques: L_p-norm, Dropout, and Dropconnect.

7.1 L_p-norm regularization

Regularization modifies the objective function by adding additional terms that penalize the model complexity and prevent overfitting. If the loss function is $\mathcal{L}(\theta, x, y)$, then the regularized loss will be:

$$\mathrm{E}(\theta, x, y) = \mathcal{L}(\theta, x, y) + \lambda R(\theta), \tag{17}$$

where $R(\theta)$ represents the regularization term, and λ represents the regularization strength.

The function used for L_p-norm regularization is $R(\theta) = \sum_j \theta_{jp}^{p}$ when $p \geq 1$, the $\mathrm{L_p}$-norm is convex, which makes the optimization easier and renders this function attractive (Hinton et al. 2012). If $p = 2$, the $\mathrm{L_2}$-norm regularization is generally referred to as weight decay. When $p < 1$, the L_p-norm regularization exploits the sparsity effect of the weights but conducts to non-convex function.

7.2 Dropout

Hinton et al. (2012) introduced the concept of dropout which is an effective method to reduce overfitting. In this method, half of the activations within a layer are stochastically set to zero for each training sample. By doing this, the hidden units cannot co-adapt to each other and learn better representation for the inputs. Goodfellow et al. (2013) showed that dropout is an effective way to control the overfitting for maxout networks because of better model averaging. Dropout ignores each hidden unit stochastically with probability p during the feed-forward operation in neural network training. The parameter p is known as dropout-rate. Initially, dropout is applied to fully connected layers and its output is represented as:

$$Y = r \times a(W^T X), \tag{18}$$

where $X = [x_1, x_2, \dots, x_n]^T$ represents the input to fully connected layer, $W \in \mathbb{R}^{n \times d}$ represents the weight matrix, and r represents the binary vector of size d. Its elements are independently selected from a Bernoulli distribution with parameter p. Dropout prevents the neural network from becoming too dependent on any one of the neurons. An improved method of the dropout was proposed by Wang and Manning (2013) that performed fast dropout training by sampling from or integrating a Gaussian approximation. An adaptive dropout method was proposed by Ba and Frey (2013). In this, the dropout probability for each hidden variable is computed using a binary belief network that shares the parameters with the DNN. Tompson et al. (2015) proposed a new dropout method called SpatialDropout. In this, standard dropout is applied before the 1×1 convolutional layer then dropout is extended across the entire feature map. As a result, the training time of the network increases but it does not prevent overfitting. This method works well especially when the training dataset size is small.

7.3 Dropconnect

Dropout and dropconnect (Wan et al. 2013) both methods intend to prevent co-adaptation of units in a neural network. The main idea is that units independently extract features from inputs instead of relying on other neurons to do so. Dropconnect works similar to dropout as it introduces dynamic sparsity within the model. Instead of randomly setting the outputs of neurons to zero, dropconnect removes the connections between the neurons, i.e., sets the elements of weight matrix W to zero. Thus each unit receives input from a random subset of units from the previous layer. The output of dropconnect is given by $y = a((R \times W)x)$, where $R_{ij} = Bernoulli(p)$ and R is a binary matrix encoding of the connection information. Additionally, R is drawn independently for each example, during the training process as well as biases are also masked out.

8 Experimental setup

The performance of ASR systems depends upon the availability of labeled speech data for training purpose. Indian languages like Hindi, Bengali, Punjabi are considered as under-resourced languages due to unavailability of large speech corpus, benchmarked data, and other resources. Some unique features distinguish them from the English. Indian languages have more vowel and consonants than the English language. They have more stop consonants as compared to English. Therefore, designing an ASR for any Indian language becomes a more challenging task. In brief, it can be said that they have a larger character set and more clear aspired sound. Length of vowels is also distinct. Earlier, only small datasets were available for them. Fortunately, some good datasets are presently available to research these languages ASR. In 2002, TIFR Mumbai developed a comparatively small corpus containing only 100 sentences due to which it cannot be suitable for discriminative models (Samudravijaya et al. 2000). For this experimental work, we use two new datasets 'Hindi as well as Bengali corpus'.

Hindi corpus is a dataset of acoustic–phonetic labels consisting of total 48 h of speech from 20 speakers. This corpus was collected in India. The voice of 20 native speakers who were demographically balanced according to age distribution (16–30), (30–45), and (45–60), gender, and dialectical regions. There are total 23,984 audio files recorded at 44.1 kHz sampling rate with 4 channels which are saved as uncompressed PCM files. All the speech data are well transcribed and labeled. The training set contains 43 h of speech and the remaining 5 h of speech for testing purpose.

The well-known Bengali speech corpus 'Shruti' was developed in 2011 by IIT Kharagpur (Das et al. 2011). It contains 7500 sentences and nearly 20,000 unique words. It is not larger, so we prefer a new corpus 'Google Bengali dataset' which is freely available at openSLR. This Google Bengali dataset is used to evaluate the performance of different pooling techniques for Bengali speech. It is a well annotated and time-aligned speech dataset which contains transcribed audio data. It contains total 196,000 utterances recorded at 22 kHz sampling rate with mono voice. This corpus was collected in West Bengal, Assam, and Tripura state of India, and Bangladesh by Google India. The complete dataset is divided into a training set containing 176,000 utterances and testing set containing 20,000 utterances.

Our baseline architecture consists of 4 convolutional layers with 64, 256, 1024, and 1024 feature maps, respectively then it is connected with three fully connected layers containing 1024 hidden units each. Pooling layer always follows a convolutional layer. The filter-size is taken p for each pooling layer with the same stride value, i.e., p. All the neurons of CNN and fully connected layers are initially taken maxout except for the experiments to test the best activation function. The default value of dropout is taken 0.3 for all the experiments except for finding the best regularization technique and dropout rate. 40 dimensional mel frequency spectral coefficients (MFSC) features are used to generate the speech feature vector. In feature extraction, the hamming window size is taken 25 ms with a fixed shift of 10 ms. 123 observations of features are formed by adding 40 coefficients + energy and their first and second order derivatives. Some fundamental mismatches always exist between training and testing data which cause degraded recognition rate. Almost speech recognition systems adopt cepstral mean and variance normalization to increase the robustness (Viikki and Laurila 1998). The same practice, we follow at training and testing time.

The input is divided into 40 bands for CNN which classifies acoustic input features into corresponding classes. By using the training data, the prior probability of the considered state is calculated. The output of the pair of convolution and pooling layers are concatenated and processed by fully connected layers. Convolution is applied only along the frequency axis because shift-invariance in frequency is more important than shift-invariance in time (Abdel-Hamid et al. 2014; Sainath et al. 2013b).

Asynchronous stochastic gradient descent (ASGD) optimization strategy (Dean et al. 2012) is used to train the neural network with the cross-entropy (CE) criteria with a context window of 15 frames. Typically, CE is used as the objective function and optimization is performed through ASGD.

9 Experimental results and discussion

It is always challenging to find how many convolutional layers should be preferred, what should be the size of pool-filter (i.e., the width of the filter), what should be the value of stride, how many hidden units in fully connected layers will offer the best result. There is no standard value for these parameters. The main reason for that is the network highly depends on the type of input. Therefore, the number of convolutional layers, number of hidden units in each fully connected layer, size of pool-filter (width of filter-size), activation functions, and regularization are taken as hyperparameters.

Out-of-vocabulary (OOV) words are unknown words that appear in the test set but not in the train set. Most ASR system are closed vocabulary recognizer that only recognize words in fixed finite vocabulary. Open vocabulary recognizer allows new words to be introduced in the recognition vocabulary. However, due to different research direction all the experiments are performed on closed vocabulary test set. The OOV rate of the test set is 1.79% w.r.t. the baseline vocabulary.

In this section, various pooling strategies that have been successfully applied in computer vision tasks are investigated for various hyperparameters for speech recognition tasks. To check their superiority, all the pooling strategies are evaluated on the Hindi speech corpora. All the pooling layers use the same pool size (i.e., *p*) and the default pooling size is taken 2. Most of the experiments are conducted on clean speech except the model robustness evaluation which is conducted on noisy speech.

Initially, all pooling methods are evaluated for a different number of convolutional layers to get the optimal value of the number of convolutional layers that offers the best recognition rate. The result given in Table 1 shows that max pooling is offering the best result for clean speech and WER decreases by adding an extra convolutional layer. The reason for the reduction in WER is that convolutional layers effectively model the locality present in the speech signal. No doubt, by adding extra convolutional layer increases the recognition rate but simultaneously there is an increase in the network's size and the number of parameters also rises exponentially. Moreover, there is no much difference between the WER achieved by four and five convolutional layers' model whereas the number of parameters used in the five convolutional layer model is high. Second, locality presents in the speech signal is adequately modeled by four convolutional layers. If we see the results of six convolutional layer model, the WER starts increasing again. The reason is that the size of the feature map is decreasing by pooling layer and some important information are losing. So it is not looking much fruitful to use five convolutional layers' or higher model. Hence, we can conclude that four convolutional layers' model offers the optimal performance with max pooling.

Next, how many numbers of hidden units in each fully connected layers will offer optimal result is evaluated for different pooling methods. For this purpose, the architecture is tested for up to 2048 hidden units, and the minimum WER is achieved at 2048 hidden units in each fully connected layer. The result given in Table 2 indicates that max pooling is providing the best results for clean speech.

It is also observed that as the number of hidden units in each fully connected layer increases then corresponding WER decreases. The decrease in WER is achieved because higher number of hidden units offer better learning capability. Although, there is not much difference between the WER offered by 1024 units/layer and 2048 units/layer. If we consider the number of parameters in the 2048 units/layer model, then they are nearly thrice than 1024 units/layer model. Therefore, the model having 1024 units/layer is looking much optimal when compared with 2048 units/layer model.

The non-overlapped pooling is used throughout the experiments. The reason for avoiding overlapped pooling is that the performance of both pooling style is the same and overlapped pooling leads to a higher number of parameters (Sainath et al. 2013a). In non-overlapped pooling, the value of stride is always the same as the width of the filter. If the

Table 2 WER as a function of number of parameters in each fully connected layer

No. of hidden units	n = 256	n = 512	n = 768	n = 1024	n = 2048
Max	19.5	18.4	17.1	16.3	16.1
Average	19.8	18.8	17.4	16.6	16.3
Stochastic	20.0	18.9	17.3	16.6	16.3
L_p	20.0	19.0	17.4	16.5	16.2
Mixed	19.9	18.9	17.5	16.5	16.3
Multiscale orderless	20.4	19.3	17.8	17.0	16.7
Spatial	19.7	18.7	17.4	16.5	16.3
Spectral	20.5	19.4	18.0	17.2	16.9

Table 1 WER as a function of number of convolutional layers

Number of conv. layers/ pooling method	n = 1	n = 2	n = 3	n = 4	n = 5	n = 6
Max	20.0	18.5	17.4	16.3	16.3	16.8
Average	20.3	18.8	17.7	16.6	16.6	17.1
Stochastic	20.5	19.0	17.7	16.6	16.6	17.1
L_p	20.5	18.9	17.6	16.5	16.6	17.1
Mixed	20.8	18.9	17.6	16.5	16.5	17.0
Multiscale orderless	20.2	19.4	18.0	17.0	17.0	17.4
Spatial	20.8	18.8	17.6	16.5	16.5	17.1
Spectral	20.3	19.4	17.7	17.2	17.2	17.7

Table 3 WER as a function of pooling-size

Pool size	P = 1 (no pooling)	P = 2	P = 3	P = 4	P = 5
Max	17.4	16.3	17.1	18.2	20.0
Average	17.4	16.6	17.3	18.3	20.1
Stochastic	17.4	16.6	17.4	18.5	20.3
L_p	17.4	16.5	17.4	18.4	20.3
Mixed	17.4	16.5	17.3	18.4	20.2
Multiscale orderless	17.4	17.0	17.8	18.9	20.8
Spatial	17.4	16.5	17.3	18.5	20.2
Spectral	17.4	17.2	18.0	19.3	20.9

filter-size is 2 units, then the value of stride should be 2. It means that filter jumps only 2 units at a time. The pool-size value uncommonly 3 and rarely 4 is used still we evaluate this model for filter-size 1–5. Here filter-size 1 is the case of no pooling. The experimental results of this experiment are shown in Table 3.

After the analysis of the result given in Table 3, it can be concluded that filter-size 2 is providing the maximum recognition rate. The main reason for this performance is pool-size 2 which is less aggressive so generates less condense features. It covers only 4 units in a time. Whereas filter-size 3, 4, and 5 are more aggressive and maps 9, 16, and 25 units into a single unit at a time, respectively. They reduce the size of the feature map in a parabolic manner. By this, some important data points may be missed and as a result, the performance of system degrades. The best recognition rate is achieved for pool-size 2 for max-pooling. The reason is that clean features are either sparse or exact value which benefits to max pooling.

The performance of the spatial pyramid pooling is near to max pooling. The reason is that the spatial pyramid pooling layer is applied only on the top of the last convolutional layer and intermediate pooling layers are max pooling means main feature reduction is made using max pooling. Second, it is successful when variable representation is required to change into fixed-size, but in speech recognition, feature representation is already of fixed-size. So it is not much useful in speech. After analysis, we can say that max pooling offers the best result with pool-size 2.

In the next experiment, the performance of different pooling methods is evaluated for various activation functions, i.e., ReLU, LReLU, maxout, etc. in fully connected layers. The experimental results for this experiment are shown in Table 4.

In this experiment, again max pooling is performing well. After analyzing the results, we see that WER is same for ReLU, LReLU, PReLU, and RReLU. Generally, incorporating a non-zero slop for negative part in rectified activation units improve the performance. However, in clean speech, very less part of negative component is present in the feature map and for positive values, these all functions are the equivalent. ELU and probout do not show any significant decrease in WER. The maxout neurons show the best performance among others because they have better abilities to fit the training data. Therefore, we can conclude that the maxout activation function is best suitable for clean speech.

In the fifth experiment, popular regularization methods i.e. L_p regularization, dropout, and dropconnect are tested for our architecture. First, experiments are performed for different versions of L_p regularization. Three variant L_0, L_1, and L_2 are tested for our architecture and their results are shown in shown in Table 5. L_0 regularization behaves as same as weight tying. It reduces the size of the network without affecting the performance. L_1 promotes sparsity hence produces the sparse output. It is computationally inefficient but

Table 5 WER as a function of L_p Regularization

L_p regularization	L_0	L_1	L_2
Max	17.1	16.6	16.5
Average	17.4	16.9	16.8
Stochastic	17.5	17.0	16.8
L_p	17.4	16.9	16.8
Mixed	17.3	16.8	16.7
Multiscale orderless	17.7	17.2	17.0
Spatial	17.3	16.7	16.7
Spectral	17.8	17.2	17.1

Table 4 WER as a function of activation method rate

Activation method	ReLu	LReLU	PReLU	RReLU	ELU	Maxout	Probout
Max	16.6	16.6	16.6	16.6	16.7	16.3	16.5
Average	16.9	17.0	17.0	17.0	17.0	16.6	16.7
Stochastic	16.9	17.0	17.1	16.9	17.0	16.6	16.7
L_p	16.9	17.0	17.1	17.0	17.1	16.6	16.7
Mixed	16.8	16.8	17.0	16.9	16.9	16.5	16.8
Multiscale orderless	17.3	17.3	17.4	17.2	17.5	17.0	17.2
Spatial	16.8	16.8	16.9	16.6	17.0	16.5	16.7
Spectral	17.5	17.5	17.5	17.4	17.7	17.2	17.5

offers the facility of built-in feature selection. L_2 promotes smoothness and computationally efficient due to having analytical solutions but does not have any features selection facility. In speech recognition experiments, L_2 regularization with max pooling is offering minimum WER but it is slightly better than L_1 regularization. The reason is that L_2 regularization uses the square slack so it is differentiable and produces the convex output.

The next experiments are performed for varying dropout rate from 0 to 0.5. Here, 0 means no regularization and 0.5 means 50% of the neurons are regularized. After analyzing the results, it is observed that as the dropout rate increases from 0 to 0.1 then the WER starts to decrease and it decreases up to 0.3 and after that WER starts to increase. The reason for the decrease in WER is that the regularization removes some connections. By this, overfitting of data is reduced. The reason for the increase in WER after dropout rate 0.3 is that high dropout rate removes too many connections and the network starts becoming shallow. As a result, some important features are not passed to the next layer and decrease in recognition rate. In this experiment, the best performance 16.3% is achieved at a dropout rate of 0.3 with max pooling and results are given in Table 6.

After that, the experiments are performed for varying dropconnect rate from 0 to 0.5. Here, 0 means no connection is removed and 0.5 means 50% of the connections are removed. After analyzing the results, it is observed that as the dropconnect rate increases from 0 to 0.1 then the WER starts to decrease and it decreases up to 0.4 and after that WER increases. The reason for the decrease in WER is that as the regularization removes some connections, then overfitting of data reduces. After dropconnect rate 0.4, WER starts increasing because too many connections are disabled and the network starts becoming shallow. Therefore, some important features are missed to pass to the next layer and the recognition rate deteriorates. In this experiment, the best performance 16.3% is achieved at a dropconnect rate of 0.4 with max pooling and results are shown in Table 7. In all the above mentioned experiments, we showed only WER for different architectures. Moreover, the perplexity of RNNLM is also observed at the same time and its value is found 109 for Hindi speech corpus.

The experiments carried on clean speech are not enough for a reliable conclusion. Simultaneously, to analyze the robustness to noise, the noisy dataset is composed by merging various degrees of noises and echo to artificially corrupt the clean signal so that the SNR exists between 10 and 25 dB. To construct the noisy dataset, noizeus dataset 'which is an extension of Aurora dataset' is used to artificially corrupt the clean speech. The noizeus dataset contains eight different noises. Out of eight, only four are selected for the experiments. Babble, restaurant, street, and train-station are different noises in which people's voice and a specific kind of noise are found, so they are the preferred choice. The best model is trained and tested using noisy dataset (Table 8).

Different noises show different effects on the spectrogram. Babble noise is generally caused by a crowd of people talking to each other. It is a stable background noise which

Table 6 WER as a function of dropout rate

Dropout rate	None	0.1	0.2	0.3	0.4	0.5
Max	18.3	17.3	16.6	16.3	16.5	17.1
Average	18.4	17.5	16.7	16.6	16.6	17.2
Stochastic	18.4	17.4	16.8	16.6	16.7	17.3
L_p	18.3	17.4	16.7	16.5	16.6	17.2
Mixed	18.3	17.5	16.8	16.5	16.6	17.3
Multiscale orderless	18.7	17.7	17.0	17.0	17.2	17.7
Spatial	18.3	17.4	16.7	16.5	16.6	17.2
Spectral	18.9	17.7	17.0	17.2	17.4	17.9

Table 7 WER as a function of dropconnect rate

Dropconnect rate	None	0.1	0.2	0.3	0.4	0.5
Max	18.3	17.3	16.6	16.5	16.3	17.1
Average	18.4	17.5	16.7	16.6	16.4	17.2
Stochastic	18.4	17.5	16.9	16.6	16.4	17.2
L_p	18.3	17.4	16.7	16.7	16.4	17.2
Mixed	18.3	17.4	16.8	16.6	16.5	17.2
Multiscale orderless	18.7	17.7	17.0	16.8	16.6	17.5
Spatial	18.3	17.4	16.7	16.6	16.4	17.3
Spectral	18.9	17.7	17.0	17.0	16.7	17.6

Table 8 WER as a function of different noises

Noise	Babble				Restaurant				Street				Train-station				Average
SNR (dB)	10	15	20	25	10	15	20	25	10	15	20	25	10	15	20	25	
Max	21.8	18.4	17.2	16.6	25.3	21.4	18.2	16.9	22.3	19.2	17.8	16.8	23.9	20.1	18.1	16.8	19.4
Average	21.7	18.4	17.3	16.9	25.4	21.7	18.5	17.2	22.4	19.4	18.0	17.0	**23.6**	**19.7**	**18.0**	**16.8**	19.5
Stochastic	**21.6**	**18.3**	**17.1**	**16.7**	**25.0**	**21.2**	**18.2**	**17.0**	22.2	19.3	17.9	16.9	23.7	19.9	18.2	17.0	**19.3**
L_p	21.8	18.6	17.4	16.8	25.6	21.6	18.4	17.1	**22.1**	**19.1**	**17.8**	**16.8**	24.0	20.4	18.3	17.0	19.5
Mixed	21.9	18.6	17.4	16.8	25.5	21.6	18.4	17.1	22.4	19.4	18.0	17.0	23.9	20.4	18.3	17.0	19.6
Multiscale orderless	22.1	18.8	17.8	17.2	25.8	22.0	18.8	17.5	22.6	19.8	18.4	17.4	24.1	20.8	18.7	17.4	19.9
Spatial	21.8	18.4	17.4	16.8	25.4	21.6	18.4	17.1	22.3	19.4	18.0	17.0	23.9	20.4	18.3	17.0	19.6
Spectral	22.3	19.1	17.9	17.3	26.0	22.1	18.9	17.6	22.8	19.9	18.5	17.5	24.4	20.9	18.8	17.5	20.0

The stochastic pooling performs well for babble & restaurant noise. L_p and average pooling show the best results for street and train-station noise respectively. Overall, stochastic pooling is doing well in noisy environment

forces the speaker to raise their volume. For this noise, stochastic pooling performs well because all values are raised by small fixed values on noise. Restaurant noise is similar to babble noise, but it contains the voice of the group of people with light background music. The difference is that this background noise is like whispering which does not highly affect the speaker's volume etc. Stochastic pooling also performs well for this noise because it does not affect the higher units. Street noise is different from the above noises because it is a vehicular noise. In this, the curve of noise raises and downs very rapidly, so it is easy to isolate. As any noise causing vehicle go through the street, the noise level will immediately rise, and as it passes away, the noise wave falls down. L_p pooling offered minimum WER for this noise because of the short-time effect of noise. The last one is the train-station noise. It is a mixture of different kind of background noises, like hawker's voice, a train arriving noise, announcements etc. These all are variable noises, so its curve is in a zigzag form i.e., not stable. Average pooling is performing well for this because in this values of spectrogram changed rapidly. After the experiments conducted on noisy speech dataset, it can be concluded that overall stochastic pooling is the best in noisy conditions because the average WER for various noises at different SNRs is minimum for stochastic pooling. The reason for good recognition rate in noisy environment is that stochastic pooling gives high chances to higher valued units.

For clean speech, max pooling shows improvements over other pooling strategies. Max pooling performs well when features are sparse, i.e., clean speech. The pooling strategies like L_p pooling and mixed pooling does not offer any significant improvement in both environments. It is to be noted that MOP and spectral pooling also degrade the performance of ASR systems. However, the gains on speech tasks are not much high when compared to gains achieved in vision task (Ren and Xu 2015). The results are approximately the same, i.e., there is no much difference in the recognition rate for all pooling strategies for training and testing set. In the results, multi-scale orderless pooling and spectral pooling shows a huge performance drop. In the noisy environment, stochastic pooling performed well, but this gain is not much remarkable. The experiments concluded that there is not a high change in recognition rate by changing the pooling strategy in speech recognition than it is for other domains such as image classification (Liu et al. 2017). The comparison of our best performing architecture with existing techniques in context of WER is given in Table 9.

10 Conclusion

The use of CNN has seen explosive growth in ASR systems. Components like pooling, weight-sharing, etc. provide more strength to CNN. Therefore, it offers a high recognition rate. Pooling is an essential component of CNN. To get a good recognition rate, it is often useful to select pooling techniques according to conditions. This paper mainly addresses the problem of understanding and uses of pooling techniques in the speech recognition task. For this purpose, we explored and investigated the appropriate pooling strategies for CNN on ASR tasks and compared them by examining their behavior from easier to harder subsequent classification. By considering experiments on Hindi speech corpus, the conclusion is that the max-pooling is the best technique in a clean environment and stochastic pooling works well

Table 9 The comparison of existing techniques in the context of word error rate

Approach	Feature extraction	Acoustic modeling	Data-set	Performance rate (accuracy)	Concluding remarks
Aggarwal and Dave (2011)	MFCC	Conventional HMM HMM + MMI, HMM + MCE, HMM + MPE	500 words	Conventional HMM (86.9%) HMM + MMI (88%) HMM + MCE (89.2%) HMM + MPE (90%)	Speaker independent, word level and Phone level small vocabulary size Hindi ASR system. Testing in clean conditions only
Mishra et al. (2011)	MFCC PLP MF-PLP BFCC	HMM	400 words (10 digits spoken by 40 speakers)	97.9% for MFCC 98.5% for PLP 99% for MF-PLP 98.1% for BFCC	Speaker independent, Hindi digits recognition for small vocabulary
Adiga et al. (2013)	MFCC, GWCC, GFCC	HMM + GMM (16 Gaussian mixtures)	8440 utterances	Set A-GFCC (71.41%) GFCC (72.57%) Set B- GWCC (73.86%) GFCC (74.45%)	Word level medium vocabulary size Hindi ASR system. Testing in clean and noisy conditions
Mandal et al. (2015)	Spectral analysis	GMM and DNN	78 h recordings	25.8% WER for GMM 18.7% WER for DNN	Word-level medium vocabulary Hindi speech recognition
Biswas et al. (2016a)	WERBC	HMM	1000 sentences of 100 speakers	WERBC-clean (78.99%)	Phoneme level medium vocabulary size Hindi ASR system. Testing in clean and noisy conditions
Dua et al. (2018b)	MFCC + DE, PLP + DE, MFPLP + DE, GFCC + DE	HMM + GMM (256 Gaussian Mixtures), HMM + MMI HMM + MPE	1000 sentences of 100 speakers	GFCC + DE + MPE 86.9 (clean) GFCC + DE + MPE 86.2 (noisy)	Sentence level medium size Hindi ASR system. Testing in clean and noisy conditions
Our Best performing Architecture	MFSC	CNN	48 h Hindi dataset	Hindi: 16.3% (WER)	Hindi large vocabulary continuous speech recognition

in noisy conditions. The results indicate that high gains are not achieved with any pooling scheme as gains achieved on vision tasks. Although much work has been done, CNN remains an important area of research with many opportunities for innovation. In the future, local filters or weight sharing may be investigated to understand the performance achieved by CNN in ASR.

Compliance with ethical standards

Conflict of interest There is no conflict of interest for this paper.

References

Abdel-Hamid O, Mohamed A, Jiang H, Penn G (2012) Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition. Paper presented at the 2012 IEEE international conference on acoustics, speech and signal processing (ICASSP). https://doi.org/10.1109/ICASSP.2012.6288864

Abdel-Hamid O, Deng L, Yu D (2013) Exploring convolutional neural network structures and optimization techniques for speech recognition. Paper presented at the interspeech. In: Bimbot F, Cerisara C, Fougeron C, Gravier G, Lamel L, Pellegrino F, Perrier P (eds) Interspeech, pp 3366–3370

Abdel-Hamid O, Mohamed AR, Jiang H, Deng L, Penn G, Yu D (2014) Convolutional neural networks for speech recognition. IEEE ACM Trans Audio Speech Lang Process 22(10):1533–1545. https://doi.org/10.1109/TASLP.2014.2339736

Adiga A, Magimai M, Seelamantula CS (2013) Gammatone wavelet cepstral coefficients for robust speech recognition. Paper presented at the TENCON 2013–2013 IEEE region 10 conference (31194). https://doi.org/10.1109/TENCON.2013.6718948

Aggarwal RK, Dave M (2011) Discriminative techniques for Hindi speech recognition system information systems for Indian languages. Springer, Berlin, pp 261–266. https://doi.org/10.1007/978-3-642-19403-0_45

Aggarwal RK, Dave M (2012a) Filterbank optimization for robust ASR using GA and PSO. Int J Speech Technol 15(2):191–201. https://doi.org/10.1007/s10772-012-9133-9

Aggarwal RK, Dave M (2012b) Integration of multiple acoustic and language models for improved Hindi speech recognition system. Int J Speech Technol 15(2):165–180. https://doi.org/10.1007/s10772-012-9131-y

Aggarwal RK, Dave M (2013) Performance evaluation of sequentially combined heterogeneous feature streams for Hindi speech recognition system. Telecommun Syst 52(3):1457–1466. https://doi.org/10.1007/s11235-011-9623-0

Ba J, Frey B (2013) Adaptive dropout for training deep neural networks. In: Proceedings of the 26th international conference on neural information processing systems (NIPS'13), vol 2, pp 3084–3092

Bhowmik T, Mandal SKD (2016) Deep neural network based phonological feature extraction for Bengali continuous speech. In: 2016 international conference on signal and information processing (IConSIP), pp 1–5. https://doi.org/10.1109/ICONSIP.2016.7857491

Biswas A, Sahu PK, Chandra M (2014) Admissible wavelet packet features based on human inner ear frequency response for Hindi consonant recognition. Comput Electr Eng 40(4):1111–1122. https://doi.org/10.1016/j.compeleceng.2014.01.008

Biswas A, Sahu P, Bhowmick A, Chandra M (2016a) Speech recognition using ERB-like admissible wavelet packet decomposition based on perceptual sub-band weighting. IETE J Res 62(2):129–139. https://doi.org/10.1080/03772063.2015.1056844

Biswas A, Sahu P, Chandra M (2016b) Admissible wavelet packet sub-band based harmonic energy features using ANOVA fusion techniques for Hindi phoneme recognition. IET Signal Proc 10(8):902–911. https://doi.org/10.1049/iet-spr.2015.0488

Boureau Y-L, Cun YL (2008) Sparse feature learning for deep belief networks. In: Proceedings of the 20th international conference on neural information processing systems (NIPS'07), pp 1185–1192

Bruna J, Szlam A, LeCun Y (2014) Signal recovery from pooling representations. In: Proceedings of the 31st international conference on machine learning, ICML 2014 Beijing, China

Clevert D-A, Unterthiner T, Hochreiter S (2016) Fast and accurate deep network learning by exponential linear units (elus). Paper presented at the international conference on learning representations (ICLR)

Das B, Mandal S, Mitra P (2011) Bengali speech corpus for continuous automatic speech recognition system. In: Paper presented at the 2011 international conference on speech database and assessments (Oriental COCOSDA), Hsinchu, 2011, pp 51–55. https://doi.org/10.1109/ICSDA.2011.6085979

Dean J, Corrado G, Monga R, Chen K, Devin M, Le QV, Mao M, Ranzato M, Senior A, Tucker P, Yang K, Ng A (2012) Large scale distributed deep networks. In: Proceedings of the 25th international conference on neural information processing systems (NIPS'12), pp 1223–1231

Dony R (2001) Karhunen–Loeve transform. In: The transform and data compression handbook, vol 1. CRC Press, Boca Raton, pp 1–34

Dua M, Aggarwal RK, Biswas M (2018a) Performance evaluation of Hindi speech recognition system using optimized filterbanks. Eng Sci Technol Int J 21(3):389–398. https://doi.org/10.1016/j.jestch.2018.04.005

Dua M, Aggarwal RK, Biswas M (2018b) GFCC based discriminatively trained noise robust continuous ASR system for Hindi language. J Ambient Intell Humaniz Comput. https://doi.org/10.1007/s12652-018-0828-x

Duhamel P, Piron B, Etcheto JM (1988) On computing the inverse DFT. IEEE Trans Acoust Speech Signal Process 36(2):285–286. https://doi.org/10.1109/TASSP.1986.1164811

Feng Y, Hao P, Zhang P, Liu X, Wu F, Wang H (2019) Supervoxel based weakly-supervised multi-level 3D CNNs for lung nodule detection and segmentation. J Ambient Intell Humaniz Comput. https://doi.org/10.1007/s12652-018-01170-5

Fukushima K, Miyake S (1982) Neocognitron: a new algorithm for pattern recognition tolerant of deformations and shifts in position. Pattern Recogn 15(6):455–469. https://doi.org/10.1016/0031-3203(82)90024-3

Gehring J, Auli M, Grangier D, Yarats D, Dauphin YN (2017) Convolutional sequence to sequence learning. In: Proceedings of the 34th international conference on machine learning (ICML'17), pp 1243–1252

Gong Y, Wang L, Guo R, Lazebnik S (2014) Multi-scale orderless pooling of deep convolutional activation features. In: Proceedings of 13th European conference on computer vision, pp 392–407. https://doi.org/10.1007/978-3-319-10584-0_26

Goodfellow IJ, Warde-Farley D, Mirza M, Courville A, Bengio Y (2013) Maxout networks. In: Proceedings of the 30th international conferenceon machine learning (ICML'13), pp 1319–1327

He K, Zhang X, Ren S, Sun J (2014) Spatial pyramid pooling in deep convolutional networks for visual recognition. In: Proceedings of the 13th European conference on computer vision (ECCV 2014), pp 346–361. https://doi.org/10.1007/978-3-319-10578-9_23

He K, Zhang X, Ren S, Sun J (2015) Delving deep into rectifiers: surpassing human-level performance on ImageNet classification. In: Proceedings of the 2015 IEEE international conference on computer vision (ICCV'15), pp 1026–1034. https://doi.org/10.1109/ICCV.2015.123

Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov RR (2012) Improving neural networks by preventing co-adaptation of feature detectors. CoRR abs/1207.0580

Hu W, Cao J, Lai X, Liu J (2019) Mean amplitude spectrum based epileptic state classification for seizure prediction using convolutional neural networks. J Ambient Intell Humaniz Comput. https://doi.org/10.1007/s12652-019-01220-6

Huang X, Acero A, Hon H-W (2001) Spoken language processing: a guide to theory, algorithm, and system development. Prentice Hall PTR, Upper Saddle River

Hubel DH, Wiesel TN (1962) Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. J Physiol 160(1):106–154. https://doi.org/10.1113/jphysiol.1962.sp006837

Imran J, Raman B (2019) Evaluating fusion of RGB-D and inertial sensors for multimodal human action recognition. J Ambient Intell Humaniz Comput. https://doi.org/10.1007/s12652-019-01239-9

Jarrett K, Kavukcuoglu K, Ranzato MA, LeCun Y (2009) What is the best multi-stage architecture for object recognition? Paper presented at the 2009 IEEE 12th international conference on computer vision. https://doi.org/10.1109/ICCV.2009.5459469

Jegou H, Perronnin F, Douze M, Sanchez J, Perez P, Schmid C (2012) Aggregating local image descriptors into compact codes. IEEE Trans Pattern Anal Mach Intell 34(9):1704–1716. https://doi.org/10.1109/TPAMI.2011.235

Koenderink JJ, Van Doorn AJ (1999) The structure of locally orderless images. Int J Comput Vis 31(2–3):159–168. https://doi.org/10.1023/A:1008065931878

LeCun Y, Boser BE, Denker JS, Henderson D, Howard RE, Hubbard WE, Jackel LD (1990) Handwritten digit recognition with a back-propagation network. In: Advances in Neural Information Processing Systems (NIPS 1989)

LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11):2278–2324. https://doi.org/10.1109/5.726791

LeCun Y, Huang FJ, Bottou L (2004) Learning methods for generic object recognition with invariance to pose and lighting. In: Proceedings of the 2004 IEEE computer society conference on computer vision and pattern recognition (CVPR 2004)

Liu X, Wang Y, Chen X, Gales MJ, Woodland PC (2014) Efficient lattice rescoring using recurrent neural network language models. Paper presented at the 2014 IEEE international conference on acoustics, speech and signal processing (ICASSP). https://doi.org/10.1109/ICASSP.2014.6854535

Liu L, Shen C, van den Hengel A (2017) Cross-convolutional-layer pooling for image recognition. IEEE Trans Pattern Anal Mach

Intell 39(11):2305–2313. https://doi.org/10.1109/TPAMI.2016.2637921

Ma M, Huang L, Xiang B, Zhou B (2015) Dependency-based convolutional neural networks for sentence embedding. In: Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing, vol 2, pp 174–179

Maas AL, Hannun AY, Ng AY (2013) Rectifier nonlinearities improve neural network acoustic models. In: ICML workshop on deep learning for audio, speech and language processing

Mandal P, Jain S, Ojha G, Shukla A (2015) Development of Hindi speech recognition system of agricultural commodities using deep neural network. In: INTERSPEECH-2015, pp 1241–1245

Mathieu M, Henaff M, LeCun Y (2014) Fast training of convolutional networks through FFTS. In: International conference on learning representations (ICLR2014), CBLS, April 2014. arXiv:1312.5851

Mishra A, Chandra M, Biswas A, Sharan S (2011) Robust features for connected Hindi digits recognition. Int J Signal Process Image Process Pattern Recogn 4(2):79–90

Nahid MMH, Islam MA, Islam MS (2016) A noble approach for recognizing Bangla real number automatically using CMU Sphinx4. In: 5th international conference on informatics, electronics and vision (ICIEV 2016). IEEE, pp 844–849. https://doi.org/10.1109/ICIEV.2016.7760121

Nahid MMH, Purkaystha B, Islam MS (2017) Bengali speech recognition: a double layered LSTM-RNN approach. In: 20th international conference of computer and information technology (ICCIT 2017), pp 1–6. https://doi.org/10.1109/ICCITECHN.2017.8281848

Nguyen LD, Gao R, Lin D, Lin Z (2019) Biomedical image classification based on a feature concatenation and ensemble of deep CNNs. J Ambient Intell Humaniz Comput. https://doi.org/10.1007/s12652-019-01276-4

Pasricha V, Aggarwal R (2016) Hybrid architecture for robust speech recognition system. In: 2016 international conference on recent advances and innovations in engineering (ICRAIE). IEEE, pp 1–7. https://doi.org/10.1109/ICRAIE.2016.7939586

Passricha V, Aggarwal RK (2018) Convolutional support vector machines for speech recognition. Int J Speech Technol 1:1. https://doi.org/10.1007/s10772-018-09584-4

Passricha V, Aggarwal RK (2019) A hybrid of deep CNN and bidirectional LSTM for automatic speech recognition. J Intell Syst. https://doi.org/10.1515/jisys-2018-0372

Ren JS, Xu L (2015) On vectorization of deep convolutional neural networks for vision tasks. Paper presented at the Proceedings of the twenty-ninth AAAI conference on artificial intelligence, Austin, Texas

Reza M, Rashid W, Mostakim M (2017) Prodorshok I: a Bengali isolated speech dataset for voice-based assistive technologies: a comparative analysis of the effects of data augmentation on HMM-GMM and DNN classifiers. In: 2017 IEEE region 10 humanitarian technology conference (R10-HTC). IEEE, pp 396–399. https://doi.org/10.1109/R10-HTC.2017.8288983

Rippel O, Snoek J, Adams RP (2015) Spectral representations for convolutional neural networks. In: Proceedings of the 28th international conference on neural information processing systems (NIPS'15), vol 2, pp 2449–2457

Sainath TN, Kingsbury B, Mohamed AR, Dahl GE, Saon G, Soltau H, Beran T, Aravkin AY, Ramabhadran B (2013a) Improvements to deep convolutional neural networks for LVCSR. In: 2013 IEEE workshop on automatic speech recognition and understanding (ASRU). IEEE, pp 315–320. https://doi.org/10.1109/ASRU.2013.6707749

Sainath TN, Mohamed AR, Kingsbury B, Ramabhadran B (2013b) Deep convolutional neural networks for LVCSR. In: 2013 IEEE international conference on acoustics, speech and signal processing. IEEE, pp 8614–8618. https://doi.org/10.1109/ICASSP.2013.6639347

Samudravijaya K, Rao PVS, Agrawal S (2000). Hindi speech database. In: Sixth international conference on spoken language processing (ICSLP 2000), Beijing, China

Sermanet P, Chintala S, LeCun Y (2012) Convolutional neural networks applied to house numbers digit classification. In: 21st international conference on pattern recognition (ICPR 2012), pp 3288–3291

Singhal S, Passricha V, Sharma P, Aggarwal RK (2018) Multi-level region-of-interest CNNs for end to end speech recognition. J Ambient Intell Humaniz Comput. https://doi.org/10.1007/s12652-018-1146-z

Soltau H, Kuo HK, Mangu L, Saon G, Beran T (2013) Neural network acoustic models for the DARPA RATS program. In: Interspeech, pp 3092–3096

Springenberg JT, Riedmiller M (2013) Improving deep neural networks with probabilistic maxout units. CoRR:1312.6116

Sze V, Chen Y-H, Yang T-J, Emer JS (2017) Efficient processing of deep neural networks: a tutorial and survey. Proc IEEE 105(12):2295–2329. https://doi.org/10.1109/JPROC.2017.2761740

Tompson J, Goroshin R, Jain A, LeCun Y, Bregler C (2015) Efficient object localization using convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), pp 648–656

Toth L (2014a) Combining time- and frequency-domain convolution in convolutional neural network-based phone recognition. In: 2014 IEEE international conference on acoustics, speech and signal processing (ICASSP). https://doi.org/10.1109/ICASSP.2014.6853584

Toth L (2014b) Convolutional deep maxout networks for phone recognition. In: Fifteenth annual conference of the international speech communication association (INTERSPEECH)

Toth L (2015) Phone recognition with hierarchical convolutional deep maxout networks. Eurasip J Audio Speech Music Process. https://doi.org/10.1186/s13636-015-0068-3

Viikki O, Laurila K (1998) Cepstral domain segmental feature vector normalization for noise robust speech recognition. Speech Commun 25(1–3):133–147. https://doi.org/10.1016/S0167-6393(98)00033-8

Wan L, Zeiler M, Zhang S, LeCun Y, Fergus R (2013) Regularization of neural networks using dropconnect. In: Proceedings of the 30th international conference on machine learning (ICML), pp 1058–1066

Wang S, Manning C (2013) Fast dropout training. In: Proceedings of the 30th international conference on machine learning (ICML), pp 118–126

Xu B, Wang N, Chen T, Li M (2015) Empirical evaluation of rectified activations in convolutional network. arXiv preprint arXiv:1505.00853

Yu D, Wang H, Chen P, Wei Z (2014) Mixed pooling for convolutional neural networks rough sets and knowledge technology. Springer International Publishing, Cham, pp 364–375. https://doi.org/10.1007/978-3-319-11740-9_34

Zavala-Mondragon LA, Lamichhane B, Zhang L, Haan GD (2019) CNN-SkelPose: a CNN-based skeleton estimation algorithm for clinical applications. J Ambient Intell Humaniz Comput. https://doi.org/10.1007/s12652-019-01259-5

Zeiler MD, Fergus R (2013) Stochastic pooling for regularization of deep convolutional neural networks. In: Proceedings of the international conference on learning representation (ICLR)