Performance Evaluation of Deep Autoencoder Network for Speech Emotion Recognition

Maria AndleebSiddiqui[1]
Computer Science and Information Technology
N.E.D University of Engineering & Technology
Karachi, Pakistan

Wajahat Hussain[2]
Deputy Manager
Karachi Shipyard and Engineering Works Ltd
Karachi, Pakistan

Syed Abbas Ali[3]
Computer & Information Systems Engineering
N.E.D University of Engineering and Technology
Karachi, Pakistan

Danish-ur-Rehman[4]
Electronics Engineering
N.E.D University of Engineering and Technology
Karachi, Pakistan

***Abstract*—The learning methods with multiple levels of representation is called deep learning methods. The composition of simple but now linear modules results in deep-learning model. Deep-learning in near future will have many more success, because it requires very little engineering in hands and it can easily take ample amount of data for computation. In this paper the deep learning network is used to recognize speech emotions. The deep Autoencoder is constructed to learn the speech emotions (Angry, Happy, Neutral, and Sad) of Normal and Autistic Children. Experimental results evident that the categorical classification accuracy of speech is 46.5% and 33.3% for Normal and Autistic children speech respectively. Whereas, Auto encoder shows a very low classification accuracy of 26.1% for only happy emotion and no classification accuracy for Angry, Neutral and Sad emotions.**

Keywords—Auto-encoder; emotions; DNN; classification accuracy; autism

I. Introduction

The composition of simple but nonlinear modules results in deep-learning model. The composition is started by representation at one level, which is usually raw input, and then this raw input is transformed into a representation of higher layers, which are slightly more abstract levels [1]. The records of machine learning techniques in many domains of science, especially in image recognition [2-5] and speech recognition [6-8] has been beaten up by deep learning modules. Deep learning has revolutionized the speech signal processing. Excellent results have been achieved using the deep learning networks [9-11]. An Autoencoder is constructed in this paper by stacking two layers; First layer is used to classify the category of speech that is normal or autistic and the second layer is used to classify the emotion of that category that is Angry, Happy, Neutral and Sad. Auto-encoder is a stack of building block. It contains multiple layers of representation [12]. Auto-encoder is also called auto-associator or Diabolo network. It is used to learn a representation of a set of data typically for dimensionality reduction [13]. Comprehensive review was presented in [14] about popular deep learning algorithms for speech emotion recognition. Experimental results shown that the performance of EMO-DB using Log Mel spectrograms on CNN+LSTM is highest that is 78.10%. To improve the Chinese speech emotion recognition, a novel speech emotion recognition algorithm based on stack autoencoder, denoise autoencoder and sparse autoencoder is proposed [15]. The experimental results revealed that the proposed algorithm with stack autoencoder performs 14.3% higher than SVM. The architecture of this algorithm can be studied in Fig. 1.

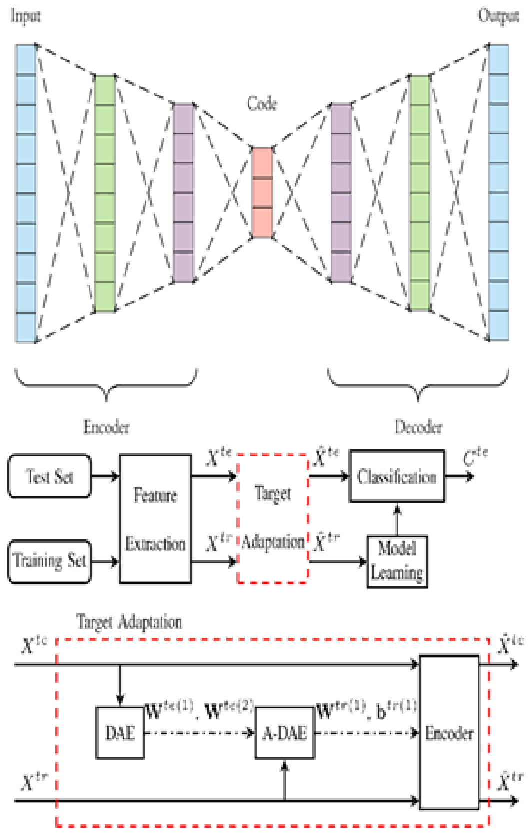

Fig. 1. Architecture of Auto-Encoder.

Here Xte are test samples, Xtr are training samples, W is weight, and b is bias

The representation of weight and bias in neural network is shown in Fig. 2. Architecture Neuron Model is an elementary neuron with "P" inputs as shown. Each input is weighted with an appropriate "w." The sum of the weighted inputs and the bias forms the input to the transfer function "f." Neurons can use any differentiable transfer function "f" to generate their output.

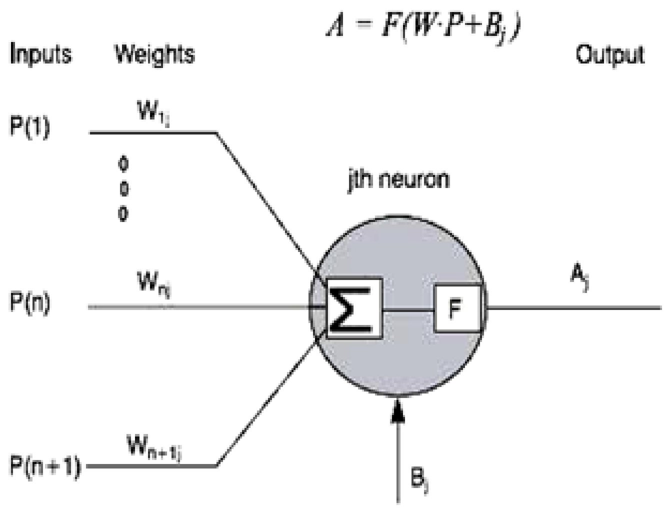

Fig. 2. Architecture of Auto-Encoder.

The basic Algorithm for auto encoder is as follows:

Step 1. Load the training speech data into memory

Step 2. Get the number of columns and rows in each sample

Step 3. Turn the training samples into vectors and put them in a matrix. As the training samples are saved into a matrix, training the network is ready to begin.

Rest of paper is organized into following three sections: Section II presents the methodology of experimental framework. Results of experimental framework with four different emotions are discussed in Section III. Conclusion is drawn in Section IV.

II. Methodology

A. Speech Data Set

The data evaluated in this study were collected from 94 normal and 94 autistic children of age group 10-13 years of both genders. Some Urdu language sentences with four different emotions (Angry, Happy, Neutral and Sad) are chosen for this study. The sentence which is suitable to utter and contain maximum phonetic information is used to implement the speech emotion corpus. The Emotion Corpus consists of 24 samples of each Angry, Happy and Neutral emotions and 22 samples of Sad Emotions. The following ITU recommendations have been used for corpus recording with specifications: SNR>= 45dB and bit rate 24120 bps. Windows 10 built in sound recorder and microphone has been used for recording the speaker's utterances with 48 kHz sampling rate and sensitivity of 56dB ± 25dB. The description of the available speech samples of both normal and autistic children for each emotion is shown in Table I.

B. Training and Configuration of Auto Encoder

The Flow diagram of auto encoder configuration, testing and training phases is presented in Fig. 3.

The training and testing of Autoencoder is discussed in subsequent sections.

1) Create and configure first auto encoder: Sparse auto encoder is trained by using speech training data set without labels. As auto encoder can replicate its input and output, so the size of the input and output will be same. The compressed representation of the input is learned by auto encoder when the size of the input is greater than the number of neurons in hidden layer. By modifying some of the settings of the feed forward neural network, the auto encoder is created.

Step 1. The size of the hidden layer, which is to be trained, is set. It is usually less than the input size.

Step 2. The number of training functions and training epochs are changed to create the network.

TABLE. I. Speech Emotion Data Set

Category	Emotions			
	Angry	***Happy***	***Neutral***	***Sad***
Normal	24	24	24	22
Autistic	24	24	24	22

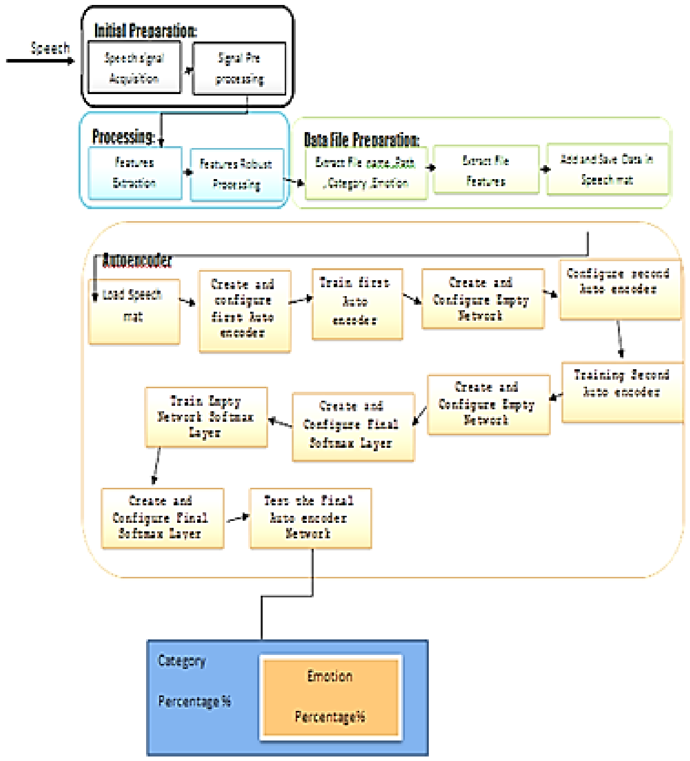
Speech
Initial Preparation:
Speech signal Acquisition
Signal Pre processing
Processing:
Features Extraction
Features Robust Processing
Data File Preparation:
Extract File name, Data, Category, Emotion
Extract File Features
Add and Save Data in Speech mat
Autoencoder
Load Speech mat
Create and configure first Auto encoder
Train first Auto encoder
Create and Configure Empty Network
Configure second Auto encoder
Training Second Auto encoder
Create and Configure Empty Network
Create and Configure Final Softmax Layer
Train Empty Network Softmax Layer
Create and Configure Final Softmax Layer
Test the Final Auto encoder Network
Category
Percentage %
Emotion
Percentage%

Fig. 3. Auto-Encoder Configuration Training and Testing Phases.

Step 3. Process function is not used at the input and output.

Step 4. The transfer function of logistic sigmoid is set for both layers.

Step 5. All the dataset is used for training.

Step 6. The first layer comprises of the sparse representation, which is learned by adding the regularizes as it encouraged the learning power of auto-encoder by using the following parameters:

a) |L2WeightRegularization|: It should be small enough to control the weighing of an L2 regularizers for the weights of the network (not the biases).

b) |sparsityRegularization|: It is used to prevent the large fraction of neurons in hidden layers from activating in response of input.

c) |sparsity|: This function is used to control the fraction of neurons.It is activated in response to the input layer in the 1st layer. Its range is between 0 and 1.

2) Train First Autoencoder

Step 1. Train the auto-encoder with the input data that should be identical to the target data.

Step 2. The auto-encoder diagram is viewed to show the size of hidden layer, input layer, output layer, as well as the transfer function for the two layers.

Step 3. From the first auto encoder the result can be visualized. Visualization helps to get the insight into the feature that can be gained. The hidden layer neurons have the weights vector associated with it in the input layer. Create and Configure empty network.

3) Create and configure empty network: Curls and Stroke patterns from the digit samples are represented by auto-encoder that are seen by the features learned. The compressed version of the input is the 100 dimensional outputs from the hidden layer of auto-encoder. Now the next auto-encoder is trained on the speech training dataset from which the set of the vectors are extracted. For training the next auto-encoder, the first version of the auto-encoder is created with the removed final layer. Removal of first layer is done by manually configuring the settings and creating an empty network object. The biases and weights can be copied from the trained auto-encoder.

Step 1. The empty network is created

Step 2. The number of inputs and outputs are set.

Step 3. The First and only layer is connected to the first input and also to the output.

Step 4. The connection for the bias term to the first layer is added.

Step 5. The size of the input and first layer is set.

Step 6. The first layer uses the logistic sigmoid transfer function.

Step 7. The first layer of trained auto encoder is used to copy the weights and biases.

Step 8. The empty network is seen by the |view| function which is equivalent to the first auto encoder with the first layer removed.

Step 9. To train the second auto-encoder, the features are now generated. This is achieved by evaluating the truncated auto-encoder on the speech training data set.

4) Configure second auto encoder: The second auto-encoder is trained in the similar way as the first auto-encoder. The main difference is that for training the second auto-encoder the speech training data set are the features obtained by hidden layer of the previous auto-encoder. The feed forward neural network is created once again and the settings are modified.

Step 1. The network is created. The number of training function, the size of the hidden layers and the training epochs are changed to conduct the experiment.

Step 2. The process function is not used at the input and output.

Step 3. The transfer function of the logistic sigmoid is set to both the layers

Step 4. All of the data is used for training.

Step 5. After creating the network, performance function is set to |msesparse|, the values of the performance function are set. The sparsity and the mean squared error with L2 weight are used to regularize the performance.

Step 6. The parameters are altered to conduct the experiment.

5) Training second autoencoder: The features generated from the previous auto-encoder are used to train the second auto-encoder.

Step 1. The second auto-encoder is trained.

Step 2. The |view| command is called once again to view the diagram of the autoencoder. The first and Second auto-encoder are similar but the size of the layers is different.

6) Create and configure empty network: As before, a version of the second auto encoder is created with the final layer removed.

Step 1. The number of inputs and layers are set.

Step 2. The first and the only layer are connected to the first input and also connect to the output.

Step 3. A connection for bias term to the first layer is added.

Step 4. The size of the input and first layer is set.

Step 5. The first layer uses the logistic sigmoid function.

Step 6. The first layer of the trained auto-encoder copies the weights and biases.

Step 7. The diagram of the network can be seen by the |view| function. With the last layer removed, it is equivalent to the second auto encoder.

Step 8. The second truncated auto encoder passes the previous set which is used to extract the second set of features.

7) Create and configure final softmax layer: The original vectors in the speech training dataset had 784 dimensions. After passing them through the first auto encoder, this was reduced to 100 dimensions. After using the second auto encoder, this was reduced again to 50 dimensions. The 50 dimensional vectors are classified into different classes to carry out the training of the final layer. A softmax layer is created for the training of the final softmax layer. The output of hidden layer from the second auto encoder is used for its training. As the softmax layer only consists of one layer, it is created manually.

Step 1. Creation of the empty network.

Step 2. The number of inputs and layers are set.

Step 3. The first and the only layer is connected to the first input and also connected to the output.

Step 4. A connection for the bias term to the first layer is added.

Step 5. The size of the input and first layer is set.

Step 6. All of the data is used for training.

Step 7. The cross-entropy performance function is used.

Step 8. The number of training functions and training epochs is changed to conduct the experiment.

8) Train Empty Network Softmax Layer

Step 1. The training of the softmax layer is carried out. Supervised learning is used to train the softmax layer unlike the auto-encoders.

Step 2. |view| command is called to view the diagram of the softmax layer.

Step 3. A multilayer neural network is formed.

Step 4. In isolation, the training of the three separate components of the deep neural network is carried out. To view these three components are useful at these points. They are the networks |autoencHid1|, |autoencHid2|, and |finalSoftmax|.

9) Create and configure final softmax layer: Multilayer neural network is formed to join together these layers. The neural network is created manually, the weights and biases from the auto encoder and softmax layers are copied and the settings are configured after the creation of the network.

Step 1. An empty network is created.

Step 2. One input and three layers are specified.

Step 3. The 1st layer is connected to the input.

Step 4. The 2nd layer is connected to the 1st layer

Step 5. The 3rd layer is connected to the 2nd layer.

Step 6. The output is connected to the 3rd layer.

Step 7. A connection for the bias term to the first layer is added.

Step 8. The size of the input is set.

Step 9. Same as the layer in autoencHid1, the size of the first layer is set.

Step 10. Same as the layer in autoencHid2, the size of the second auto encoder is set.

Step 11. Same as the layer in final softmax layer, the size of the third layer is set.

Step 12. Same as in 1st auto encoders, the transfer function for the first layer is set.

Step 13. Same as in 2nd auto encoder, the transfer function for the second layer is set.

Step 14. Same as in Softmax layer, the transfer function for the third layer is set.

Step 15. Use all of the data for training

Step 16. Copy the weights and biases from the three networks that have already been trained

Step 17. Use the cross-entropy performance function

Step 18. The experiment can be conducted by changing the number of training epochs.

Step 19. |view| command can be used to see the diagram of the multi-layer network.

10)Test the final autoencoder network: The test set is used to compute the results with the full deep neural network. Now, the test samples have to be reshaped, as was done for the training set.

Step 1. The test samples feature set is loaded.

Step 2. Confusion matrix is used to visualized the results. Overall accuracy can be calculated by the numbers in the bottom right hand square of the matrix.

Step 3. The Deep Neural Network is tuned finely. Back propagation application on the whole multi-layer network is used to improve the results for the deep neural network. This process is called fine tuning. Finally the supervised training is used to tune this network by retaining it on the speech training data set.

III. Results

The experimental results in this section are based on Confusion Matrix of categorical classification as shown in Fig. 4.

Categories: Normal and Autistic

Total Samples: 188

Target Samples:

Normal 94 Autistic 94

Output Sample Classes:

Normal:175 Autistic:13

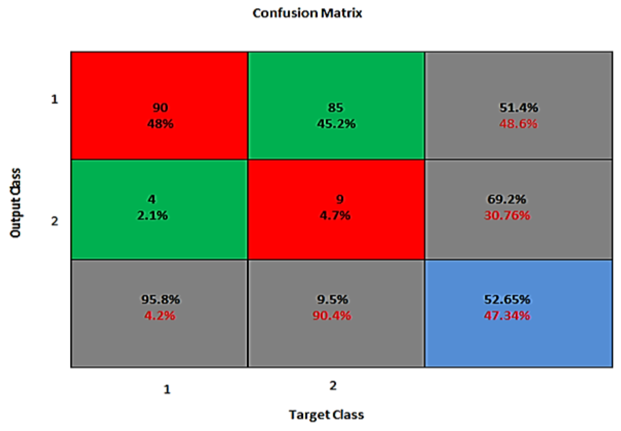

Fig. 4. Confusion Matrix of Categorical Classification.

Here 1= Normal , 2 = Autistic Results

Green are accurate hits per class, Red are error /miss per class.

Our results are 23 Normal hits, missed as Autistic and 20 Autistic hits, missed as Normal.

The Error Histogram is shown in Fig. 5.

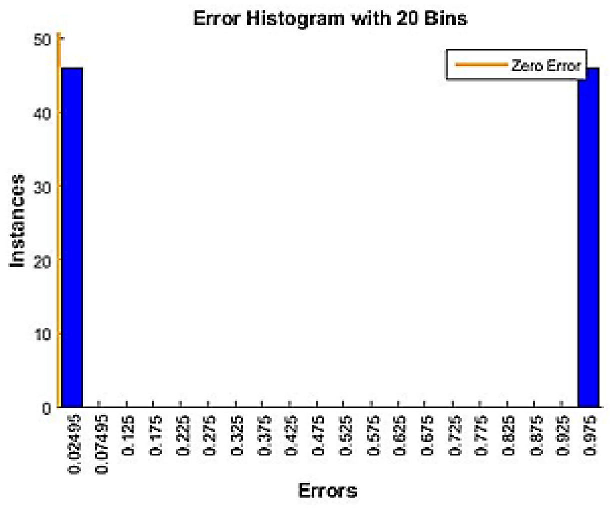

Fig. 5. Error Histogram of Categorical Classification.

Bins are sample for views. The algorithm has 45.7% error due to less number of dataset.

The emotional classification is presented in Fig. 6.

Categories: Normal and Autistic

Total Samples: 94

Target Samples:

Angry :24 Happy :24 Neutral: 24 Sad :22

Output Sample Classes:

Angry : 0 Happy :94 Neutral: 0 Sad: 0

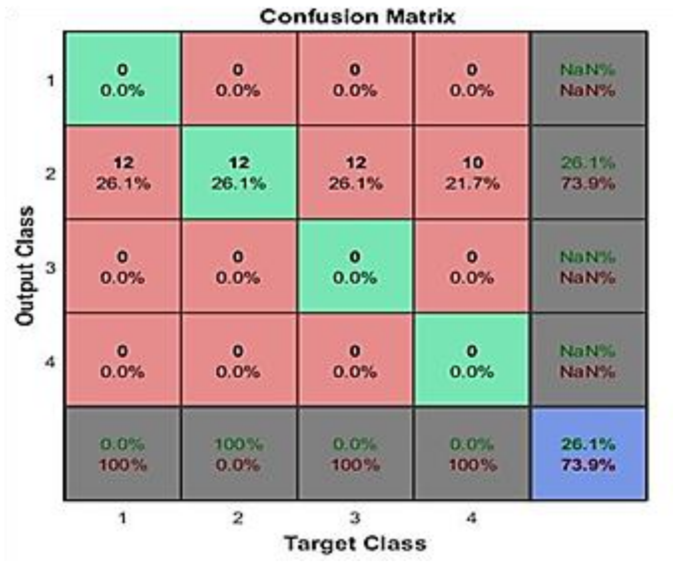

Fig. 6. Confusion Matrix Emotional Classification.

Here 1= Angry, 2 = Happy, 3 = Neutral, 4 =Sad

Green are accurate hits per class, Red are error /miss per class. The result is all Angry, Neutral and Sad are missed as Happy and all Happy are hits. The error histogram is shown in Fig. 7.

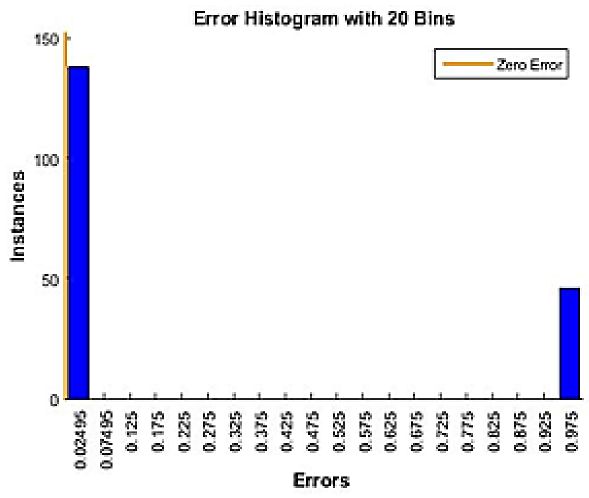

Fig. 7. Error Histogram of Emotional Classification.

IV. Conclusion

This paper evaluated the performance of deep auto encoder for four different emotions of normal and autism children. Experimental frame work were comprised on total 94 speech emotions sample in four different emotions and make used of confusion matrix to demonstrate results in term of classification accuracy. Experimental framework of categorical classification produced overall accuracy of 52.65% and the overall emotional classification of speech produces 26.1% accuracy which shows very low percentage of classification accuracy of emotions. Authors are focusing on improving classification accuracy by increasing the emotions corpus of the children.

References

[1] Y.L.Cun , Y. Bengio, G.Hinton, “Review on Deep Learning”, Nature, vol. 521,pp.436-444.2015.

[2] A. Krizhevsky, I. Sutskever, G. Hinton, “ImageNet classification with deep convolutional neural networks”,, In Proc. Advances in Neural Information Processing Systems, vol. 25, pp. 1090–1098, 2012.

[3] C. Farabet, C.Couprie, L.Najman,Y.LeCun, "Learning Hierarchical Features for Scene Labeling," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.35, No.8,pp.1915-1929.2013.

[4] J. Tompson., A.Jain, Y. LeCun, C. Bregler, “Joint training of a convolutional network and a graphical model for human pose estimation”, In Proc. Advances in Neural Information Processing Systems , vol. 27,pp. 1799–1807.2014.

[5] C.Szegedy et al.,“Going deeper with convolutions”, Computer Vision Foundation,pp.1-9.2015.

[6] T. Mikolov ,A. Deoras, D. Povey,L. Burget, J.Cernock, “Strategies for training large scale neural network language models”, In Proc. Automatic Speech Recognition and Understanding,pp.196–201,2011.

[7] G. Hinton, et al., “Deep neural networks for acoustic modeling in speech recognition”. IEEE Signal Processing Magazine, vol. 29,pp.82–97.2012.

[8] T.Sainath, A.R.Mohamed ,B. Kingsbury, B.Ramabhadran, “Deep convolutional neural networks for LVCSR”, In Proc. Acoustics, Speech and Signal Processing , pp.8614–8618, 2013.

[9] A. Graves, N. Jaitley, A.R.Mohamed. “Hybrid Speech Recognition with Deep Bidirsctional LSTM”, in proc. of IEEE workshop on Automatic Speech Recognition and Understanding, pp 273-278, 2013.

[10] K. Han, D. Yu, I.Tashev,“Speech Emotion Recognition using deep neural network and extreme learning machine”, in Proc. of the international speech communication and association conference, pp.223-227, 2014.

[11] H. Lee, L.Yan , P. Dham, A.Y.Ng, “Unsupervised feature learning for audio classification using convolution deep belief networks, Neural Information Processing System, pp.1096-1104.2009.

[12] S.S. Mousavi, M.Schukat, E.Howlay, “Deep Reinforcement Learning: An overview”, In Proc. of SAI Intelligent System Conference (Intellisys), pp. 426-440.2018.

[13] Y.Bengio, “Learning deep architectures for Artificial Intelligence”, Foundations and trends in Machine Learning, vol.2, No.1, pp.1–127. 2009.

[14] S.K.Pandey, H.S Shekhawat, S.R.M Prasanna, "Deep LEarning Techniques for speech emotion recognition: A Review", IEEE Proc. on International Conference on Radioelectronika, Czech Republic, pp:1-5, July 2019.

[15] P.Wei, Y.Zhao, "A novel speech emotion recognition algorithm based on wavelet kernel sparse classifier in stacked deep auto-encoder model", Springer: Pers Ubiquit Comput 23, pp. 521–529, 2019.