Speech recognition for simple speech commands

Stephy Benny (Assistant Professor), Mukesh Choudhary, Abhijeet Iyer, Ganesh S, Shreyas Mudaliar
Information Technology, SIES Graduate School of Technology
Nerul, Navi Mumbai

I. ***Abstract*— The paper proposes the development of a Speech Recognition system using Mel filter techniques. Speech recognition is used to recognize the speech words faster, efficiently and accurately. Mel Spectrogram is used to find characteristics from spoken word voice sample with respect to the word uttered by the speaker. Convolutional Neural Network (CNN) architecture with Batch Normalization is used and an accuracy of 92-94% for 12 labels is attained. This model can be further improved by EdgeSpeechNet or DeepSpeech and image recognition models.**

Keywords: Speech Recognition, CNN, MFCC, Mel Spectrogram, Batch normalization

II. INTRODUCTION

Human-computer interaction led to research in Speech Recognition. It one of the most active areas of research from the last few decades. Many researches have been dedicated to speech recognition research have been reported in the past few years. Automatic speech recognition (ASR) converts speech signals into list of words by means of an algorithm. Current speech understanding systems can understand input vocabularies of hundreds of words in optional environments. Speech signal has two important types of information: (i) speech content and (ii) The speaker identity. Speech content recognizers aim to extract the word information from the speech signal independent of the speaker by reducing the noise values and preparing a spectrogram image of the voice sample and identifying the appropriate label for the voice ample. [13]

III. SPEECH RECOGNITION

Speech recognition is a type of pattern recognition. Fig1 shows the stages for processing involved in speech recognition. The phases in supervised pattern recognition, are training phase and testing phase. The process of feature extraction is relevant for classification in both phases. During training phase, the classification model parameters are estimated using a large number of classes. During testing phase, the test pattern features are matched with the trained model of each individual class. The test pattern is concluded to belong to that class whose model has a higher probability ratio of matching with the best test pattern [13].

The goal of speech recognition is to generate the optimal word subject to linguistic constraints. The vocal evidence provided by the signal models of such units is combined with the rules of finding nearer features of the audio samples and returning an ideal word as output. In the case of speech recognition, two domains are present for the pattern matching stage namely: acoustic and symbolic. In the acoustic domain, a vectorized feature corresponds to a small segment of frame of speech, this is matched with the model of each and every class. The segment or frame is assigned a set of class labels along with their respective scores. This process of labeling is done for every feature vector in the feature vector sequence from the test data. The resultant label hypotheses is processed with the language model to yield the recognized sentence [14]

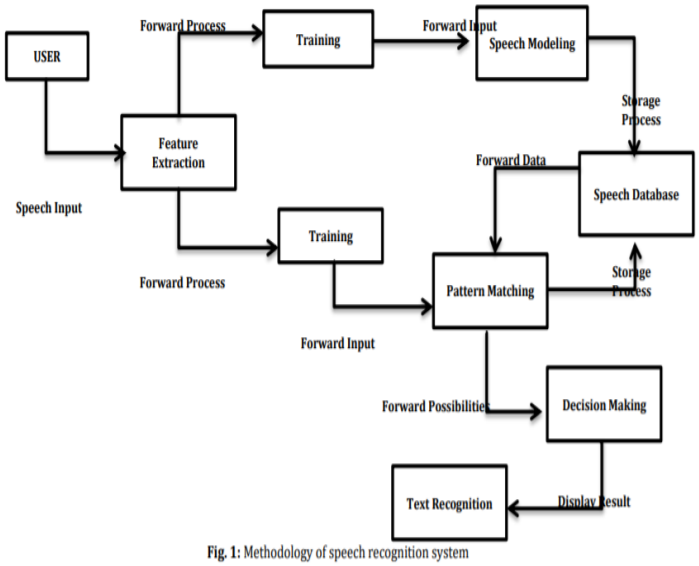


Fig. 1: Methodology of speech recognition system

Fig1: Methodology of System. [10]

IV. PREPROCESSING

A. Dataset

Speech Dataset from Google Commands Dataset or also referred as Tensor flow command dataset was used for building the model. The Dataset featured 30 labels namely yes, no, up, down, right, left, bed, bedroom, cat, dog, etc., and each label has more than 3000 voice samples, and in all 105829 voice samples.

B. *VAD* (Voice Activity Detection)

Although the words are short, there is silence in them. A decent *VAD* can reduce training size, accelerating training speed significantly. For voice files with sample rates below 1600 a random digit is added to them, in this manner of chopping audio files silence can be removed from the audio samples.

V. FEATURE EXTRACTION

A. Logfbank –

Log Mel-filter bank features energy from an audio signal. A numpy array of size numframes by nfilt containing features is returned. A feature vector is present in each row.

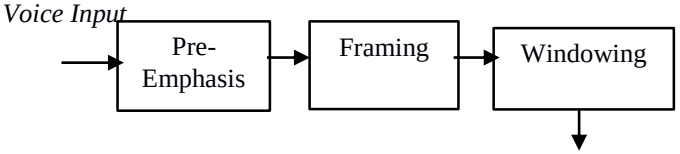
$$H_m(k) = \begin{cases} 0 & k < f(m-1) \\ \dfrac{k - f(m-1)}{f(m) - f(m-1)} & f(m-1) \le k < f(m) \\ 1 & k = f(m) \\ \dfrac{f(m+1) - k}{f(m+1) - f(m)} & f(m) < k \le f(m+1) \\ 0 & k > f(m-1) \end{cases}$$

Fig2: A base function for calculating Logfbank from Mels [1]

B. MFCC

Extracting features is the first step in an automatic speech recognition system (ASR) i.e. identify the components of the audio signal that is ideal for identifying the word content and discarding all the other irrelevant information which carries information like background noise, silence, audio noise. Sounds generated by humans varies with respect to the shape of the vocal tract. This shape or the signal waves of the tract determines what sound is produced. The job of FFT (Fast Fourier Transform) is to accurately represent the shape of the vocal tract. Mel Frequency Cepstral Coefficients (MFCC) is a popular Mel-spectral filter used in recognition approach. It has less complexity, from spoken word samples in the database 20 coefficients of MFCC of the Mel scale frequencies of speech Cepstrum are extracted. The MFCC samples are statistically analyzed for components. [7]

An important factor in ASR is the extraction and selection of the parametric representation i.e. preprocessing the acoustic signals, which affects the recognition performance. Mel-frequency Cepstrum coefficients (MFCC), is the result of a discrete cosine transform (DCT) step, the logarithm of the short-term energy spectrum expressed on a scale of Mel-frequency. The MFCCs are proven to be more efficient in machine learning algorithms but has lower accuracy in neural networks because of the DCT factor. The steps for the calculation of the MFCC include fft, logmel, dct. Linear scale for Mel-frequency wrapping frequency contents of sounds for speech signal is not followed. A frequency f, a pitch Hz is measured on a 'Mel' scale. The Mel frequency scale is a linear frequency spacing below 1000 Hz and has a logarithmic spacing above 1000Hz. Filter bank have response of a triangular band pass frequency, a constant Mel-frequency interval determines the spacing and bandwidth. The Mel scale filter bank is a series of l triangular band pass filters simulates the band pass filtering which occurs in the auditory system. [7] In final step, we convert the log Mel spectrum back to time, and Mel Frequency Cepstrum Coefficients (MFCC) is obtained. A representation of the local spectral properties of the signal provides a Cepstral representation of the speech spectrum for the given frame analysis. Mel spectrum
Coefficients are real numbers, so they are converted to the time domain using DCT. [7].



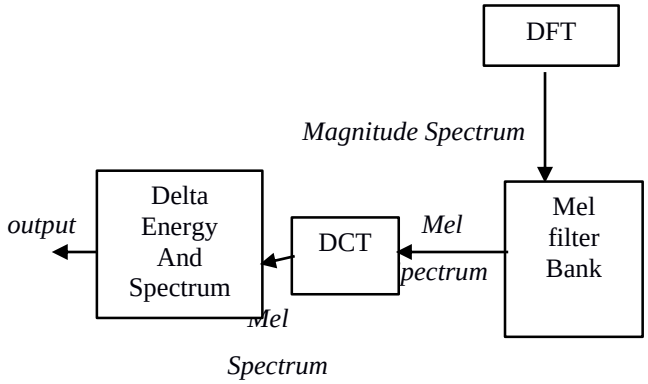


Fig3: MFCC Block Diagram [1]

Mel Scale

The Mel scale relates pitch of a tone to its measured frequency. Humans can discern small changes in pitch at low frequencies than at high frequencies. Incorporating this scale, the features match more closely to experimental or human hearing standards. [10]

The formula for converting from frequency to Mel scale is:

$$M(f) = 1125ln\left(1 + \frac{f}{700}\right) (1)$$

Equation for Mels back to frequency:

$$M - 1(m) = 700\left(exp\left(\frac{m}{1125}\right) - 1\right)$$

C. *Mel Spectrogram*

Sound is represented in Mel Spectrogram as an acoustic time-frequency as the power spectral density $P(f,t)$. It is sampled around equally spaced times into a number of points ,times ti and frequencies fj (on a Mel frequency scale). With Deep Neural Networks Mel-spectrogram performs better than MFCCs.

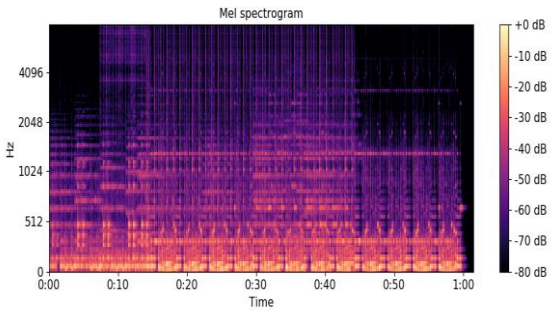


Fig 4: Output of Mel spectrogram [10]

In MIR tasks mostly, the inputs are obtained as short fragments, which is of about 2 to 4 seconds of sound. Considering a sampling rate of 6001 Hz, a window size of 2047 samples and a time shift parameter of 512 samples, i.e., 23 ms, the resulting

spectrogram obtained contains only positive frequencies for convenience), where the hindmost is the time dimension. Here we can see that the frequency dimension is or has been over-sampled. The singular bins present in the higher frequency regions have much lower energy and little information than in lower regions. Calculating the Mel-spectrogram for these cases is way too easy and simply uncomplicated method of bringing down the information to merely 80 frequency channels by averaging them over increasingly many frequency bins. The number of 80 channels has been determined with prefatory experiments as a breakpoint for optimal CNN performance, obviously because of a adequate resolution along the frequency dimension.[15]

We now define the following building blocks of a typical CNN:

•Convolution:

$$S * w(m,n) := \sum_{m'} \sum_{n'} S(m',n')w(m-m',n-n')$$

•Pooling:

For $1 \le p \le \infty$, we define A×B pooling as the operator mapping an M × N array S_0 to a $\frac{M}{A} \times \frac{N}{B}$ array S_1 by

$$S_1(m,n) = P_p{}^{A,B}(m,n) = \left|v_{S_0}{}^{m,n}\right| \vee \quad _p where v_{S_0}{}^{m,n}$$

For $m = 1,\dots,\frac{M}{A}$ and $n = 1,\dots,\frac{N}{B}$, is the vector consisting of the array entries S_0; $(n-1)\cdot B+1,\dots,n\cdot B$. In this work, we use max-pooling, which has been the most successful choice, corresponding $p = \infty$ in the above formula.[9]

The window function for generating the spectrogram by $g \in CN$ and the mel-filters, typically given by simple triangular functions, by $\Lambda_v \in CN$ for $v \in I = \{1,\dots,K\}$, where K is the chosen number of filters. We can then write the mel-spectrogram as

$$MSg(f)(b,v) = \sum_k \quad ⍰(f.T_b g)(k) \vee \quad ^2.\Lambda_v(k).$$

We then compare two different settings which lead to a time-frequency feature map which is then used as input to the deeper layers of the CNN:1. STFT-based: Compute spectrogram and take weighted averages over certain regions in frequency; for the classical Mel scale this leads to the Mel-spectrogram coefficients, but other choices of Λ_v are possible. Taking time- and frequency-sampling parametersα,βinto account, the resulting time-frequency feature map is computed for $b = \alpha l_0$ as follows.

$$MS_g(f)(b,v) = \sum_k \quad ⍰(f.T_b g)(\beta k) \vee \quad ^2.\Lambda_v(\beta k). (5)$$

2. Filter bank-based: compute filtered version off with respect to some, possibly adaptive, filter bank h_v, v ∈ I and apply subsequent time-averaging using a time-averaging functionϖ_v:

$$FB_{h_v}(f)(b,v) = \sum_l \quad (f * h_v)(\alpha l) \vee \quad ^2.\varpi_v(\alpha l - b). [15]$$

The figure 5 shows the working of Mel spectrogram

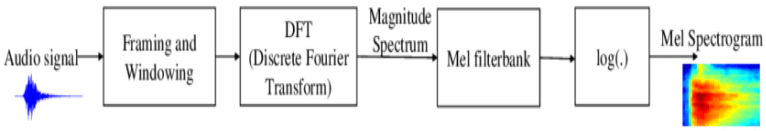


Fig 5: Mel Spectrogram Block Diagram [1]

VI. MODEL

A. Layers

- Sequential Model

The Sequential model is consists of a stack of layers that are arranged in a linear manner. Before training a model, we need to configure the input shape of the layers,hidden layers if they are present and their dimensions. This layer is usually the default layer for any Keras model.

- Batch Normalization

The dispensation of each layer's inputs changes during training of the model because the parameters of the preceding layers change. Thus, training our model using Deep Neural Networks is an intricate process to do with. This reduces the speed of training the model as it acquires for itself a lower learning rates, needs careful parameter initialization thus making it reputedly hard to train models with drenching nonlinearities. We mention to this circumstance as the internal covariate nshift. The solution we use to eliminate this problem is by normalizing layer input.[15] .

The process of batch normalizing draws its source from making normalization a part of the model architecture. It is performed by doing the normalization for mini-batch size due to which it is called as Batch Normalization. [3]
It is a technique used for improving the performance and stability of neural networks, and also makes more complicated and advanced deep learning models work in practice. The idea behind is to normalize the inputs of each layer in such a way that they have a mean output after applying activation function tends to zero or is zero and a standard deviation of the same nears one or is equal to one. This is analogous to how the inputs to networks are standardized. The main theory used here is to normalizing the inputs given to layers within the network and not normalizing the input to the network directly. We normalize the activations of the previous layer for each batch during the training phase of the model, so thus we call this mini-group of layers taken for normalizing as "batch".[3] We apply a transformation that maintains the mean activation value close to 0 and the activation standard deviation value close to 1. Other than some of the innate reasons, there are some good mathematical basis why batch normalization helps the network or the model to learn better and faster. By doing the batch normalization helps combat what the authors term it as internal covariate shift. As Batch Normalization is faster to train and learn, it permits us to use much higher and effective learning rates of the model and be less worried and cautious about initialization. It also acts as a regularizer for that batch, in some cases eliminating the need for Dropout. Applied to a state-of-the-art image classification model, Batch Normalization achieves the same accuracy with 14 times fewer training steps and beats the original model by a significant margin. [8]

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Fig 6: Batch Normalizing Transform, applied to activation x over a mini-batch [3]

Benefits of batch normalization

- works train faster
- Allows higher learning rates
- Makes weights easier to initialize
- Makes more activation functions viable
- Simplifies the creation of deeper networks
- Provides some regularization.

B. Optimizers and activation functions

- Adam optimizer: Adam was used because speech samples contained noise. The method is straightforward efficient to implement, requires less memory, it is also invariant to the diagonal rescaling of the gradients, Hence well suited for problems which are large in terms of data and/or parameters[16]. The method is also appropriate for non-stationary objectives and for problems having sparse gradients and/or very high noise[16]. Adaptive Moment Estimation (Adam) is another method where for each parameter adaptive learning is computed. To store an exponentially decaying average squared gradients vt of past like Adadelta and RMS prop, Adam also uses the exponentially decaying average gradients mt of past, which is similar to momentum [17]. Whereas momentum can be seen as a ball running down a slope, Adam behaves like a heavy ball with friction, which thus prefers flat minima in the error surface. The decaying averages of past and past squared gradients mt and vt can be computed respectively as follows:

$$m_t = \beta_1 . m_{t-1} + (1 - \beta_1) . g_t$$
$$v_t = \beta_2 . v_{t-1} + (1 - \beta_2) . {g_t}^2$$

mt and vt are first moment(the mean) estimates and second moment (the uncentered variance) estimates of the gradients respectively. As mt and vt are vectors initialized as 0's, the authors of Adam observed that they are zero biased, especially at the time of initial time steps, and also at smaller decay rates (i.e. β1 and β2are close to 1). They counteract these biases by computing bias-corrected first and second-moment estimates:

$$m_t = \frac{m_t}{\left(1 - {\beta_1}^t\right)}$$

$$v_t = \frac{v_t}{\left(1 - {\beta_2}^t\right)}$$

They then use these to update the parameters just as we have seen in Ad delta and RMS prop, which yields the Adam update rule:

$$\theta_{t+1} = \theta_t - \eta \sqrt{v}_t + \epsilon_t^m$$

The authors propose default values of 0.9 for β_1, 0.999 for β_2, and 10−8 for ϵ. Hence, it is empirically shown that Adam works well in practice and also compares favourably to other adaptive learning-method algorithms [16].

- SoftMax

It is used to compute probability distribution from a vector of real numbers. The SoftMax function produces an output which is a range of values between 0and 1, with the sum of the probabilities been equal to 1. The main difference between the Sigmoid and SoftMax AF is that the Sigmoid is used in binary classification while the SoftMax is used for multivariate classification tasks. [4]

Suppose there are M classes and l labelled training data $(x_1, y_1), \cdots, (x_l, y_l)$, where $xi \in Rm$ is the ith training example and $yi \in \{1, \cdots, M\}$ is the class label of xi. For an example, let us denote the output (decision function value) of the kth binary classifier (class ωk versus the rest) as $rik; rik$ is expected to be large if xi is in class ωk and small otherwise[18]. After Mone-versus-all binary classifiers are constructed, we can obtain the posterior probabilities through a soft-max function

$$P_k{}^i = Prob(X_i) = \frac{e^{w_k {r_k}^i + w_{k_0}}}{z^i} \qquad (1)$$

where

$$z^i = \sum_{k=1}^{M} e^{w_k {r_k}^i + w_{k_0}}$$

is a normalization term that ensures that [18].

$$\sum_{k=1}^{M} P_k{}^i = 1$$

The parameters of the soft-max function, $(w_1, w_{1o}), \cdots, (w_M, w_{Mo})$, can be designed by minimizing a penalized negative log-likelihood (NLL) function , i.e.,

$$E = \frac{1}{2}\|w\|^2 - C\sum_{i=1}^{l} logP_{yi}{}^i (2)$$

subject to $w_k, w_{ko} > 0, k = 1, \cdots, M (3)$

where $\|w\|^2 = \sum_{i=1}^{l} ({w_k}^2 + {w_{ko}}^2)$

and C is a positive regularization parameter.[11]

- ReLu:

ReLu is used as an activation function in DNNs, with SoftMax function as their classification function.[6]

C. Neural Network Layers

Conv2d:

The most basic building block in a general neural network may be written as$x_n + 1 = \sigma(A_n x_n + B_n)$ where x_n is the data vector, or array, in the n^{th} layer, A represents a linear operator, bins a vector of biases in the n^{th} layer and the non-linearity is applied component-wise. Note that in each layer the array x_n may have a different dimension. Now, in the case of convolutional layers of CNN's, the matrix A has a particular structure for the convolutional layers, namely, it is a block-Toeplitz matrix, or, depending on the implementation of the filters, a concatenation of circular matrices, each representing one convolution kernel. There may be an arbitrarily high number of convolutional layers, followed by a certain number of so-called dense layers, for which is again an arbitrary linear operator. [9]

The first required parameter in conv2d is the number of filters that the convolutional layer will learn.

In the network architecture the early layers (i.e., closer to the actual input image) learn convolutional filters fewer while layers in the network deeper (i.e., closer to the output predictions) will learn *more* filters.[12]

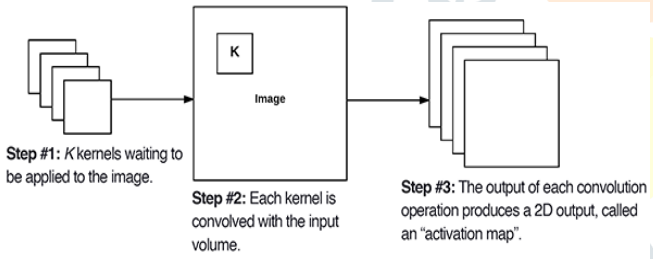


Fig 7: Conv2d block diagram[9]

VII. Training Methodology

The process began with extracting MFCC and STD features from
the Google Commands Dataset and received an accuracy of max 80% using a conv2d using Keras, The learning rate was to 0.001 and the dataset was divided into 80% to 20% of training and testing samples respectively. We used ReLU with SoftMax and applied Adam, Adamax, Nadam though higher accuracy was achieved with Adam optimizer.12 labels were used for training the model. Deep neural susceptible to highly correlated data and hence we will be using Mel spectrograms instead of mfcc, since spectrograms do not necessarily include the DCT (Discrete Fourier Transform) step.

Before applying Mel spectrogram, the audio samples were padded and chopped according to set parameters, the window size was set to 20 and step size of 10 was used for Mels, images are stored for each sample. With Mfcc 80% accuracy was achieved. Accuracy was improved with batch normalization with 5 layers of conv1d and ReLu activation, 80:20 ratio of training to testing was used from the same and achieved an accuracy of 92% validation accuracy for it.

CONCLUSION

Speech has been the easiest and reliable form of communication for living beings. Speech Recognition has helped to make machines user friendly and to work with ease. Speech as a biometric authentication has emerged due to decades of research. Research in speech recognition has led to creation of varied models and striving to achieving greater accuracies. Machines have been made to respond to human speech commands with acceptable accuracies. This paper attempts to provide a insight on usage on neural networks for detecting certain speech commands. Research is underway to obtain more accuracy and research on digital signal processing for understanding signal patterns more effectively.

Acknowledgment

We wish to express our deep sense of gratitude to thank our project guide Prof.Stephy Benny for providing timely assistant to our query and guidance. We take this opportunity to thank our HOD. Prof.K. Lakshmisudha and Principal, Dr. Vikram Patil for their valuable guidance and immense support in providing all the necessary facilities. We would like to thank the entire faculty of the IT Department for their valuable ideas and timely assistance in this project. Last but not least, we would also like to thank teaching and non-teaching staff members of our college for their support, in facilitating timely completion of this project.

References

[1] Haytham Fayek, "Speech Processing for Machine Learning: Filter banks Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between", Apr 2016, [online] Available: https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html.

[2] Simeon kostadinov,Dec 16,2017,"Understanding GRU Networks", retrieved from url: https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be

[3] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation Functions: Comparison oftrends in Practice and Research for Deep Learning," arXiv:1811.03378, pp. 1–20, 2018.

[4] Adrian Rosebrock, December 31 2018, "Keras Conv2D and Convolutional Layers"from url:https://www.pyimagesearch.com/2018/12/31/keras-conv2d-and-convolutional-layers/

[5] A. F. Agarap, "Deep Learning using Rectified Linear Units (ReLU)," arXiv:1803.08375, no. 1, pp. 2–8, 2018.

[6] H. Sak, A. Senior, and F. Beaufays, "Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition," , arXiv:1402.1128, no. Cd, 2014.

[7] Jaron Colins,Jun 27,2017, "Glossary of Deep Learning: Batch Normalisation"[online], available :https://medium.com/deeper-learning/glossary-of-deep-learning-batch-normalisation-8266dcd2fa82

[8] S. Ioffe and S. Christian, "Batch Normalization:

Accelerating Deep Network Training by Reducing Internal Covariate Shift," arXiv:1502.03167, vol. abs/1502.0, 2018.

[9] M. Dörfler, T. Grill, R. Bammer, and A. Flexer, "Basic filters for convolutional neural networks applied to music: Training or design?,arXiv 1709.02291, Neural Comput. Appl., 2018.

[10] S. B. Davis and P. Mermelstein, "Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences," IEEE Trans. Acoust., vol. 28, no. 4, pp. 357–366, 1980.

[11] K. Duan, A. N. Poo, S. K. Shevade, S. S. Keerthi, and W. Chu, "Multi-category Classification by Soft-Max Combination of Binary Classifiers,", MCS'03 Proceedings of the 4th international conference on Multiple classifier systems, pp 125-134, 2007.

[12] Adrian Rosebrock, 31/12/2018,"Keras Conv2D and Convolutional Layers"[online], available : https://www.pyimagesearch.com/2018/12/31/keras-conv2d-and-convolutional-layers/

[13] S. Swamy and R. K.V, "An Efficient Speech Recognition System," Comput. Sci. Eng. An Int. J., vol. 3, no. 4, pp. 21–27, 2013.

[14] R. E. Gruhn, W. Minker, and S. Nakamura, "Statistical Pronunciation Modeling for Non-Native Speech Processing," Stat. Pronunciation Model. Non-Native Speech Process. Springer, pp. 5–17, 2011.

[15] M. D¨orfler, T. Grill, R. Bammer, A. Flexer ,Basic Filters for Convolutional Neural Networks Appliedto Music: Training or Design,19 Sep 2018,arXiv:1709.02291v3

[16] R. Shindjalova, K. Prodanova,V. Svechtarov, Modeling Data for Tilted Implants in Grafted with Bio-Oss Maxillary Sinuses Using Logistic Regression, AIP Conference Proceedings, vol. 1631, pp. 58-62, 2014.

[17] Ruder, S. An overview of gradient descent optimization algorithms.CoRR, abs/1609.04747, 2016.

[18] K. Duan, S. Keerthi, W. Chu, S. Shevade, and A. Poo, "Multi-category classification by soft-max combination of binary classifiers," Multiple Classifier Systems, pp. 160-160, 2003.