Speech Recognition using MFCC & CNN

Rajashree Arunashmi, Sudhansu Sekhar Gouda and Keshav Agrawal
Computer Science and Engineering
Institute of Technical Education and Research, Odisha, India
E-mail: rajashreecuty18@gmail.com, accessthecrash@gmail.com, keshavagrawal21@gmail.com

Abstract
In this paper, we present a way of implementing an automatic speech recognition system to recognize speech utterances in the presence of background additive noise with minimal word error rate (WER) for Odia language. The system uses Mel Frequency Cepstral Coefficients (MFCC) and Convolutional Neural Network (CNN) for its functioning. The preparation of the audio is done using MFCC, training of audio files and speech recognition is done using CNN.

Keywords:
Speech recognition, Odia, MFCC, CNN, serviceology

1 INTRODUCTION

The rapid growth and advancement of speech technology makes it cheaper, efficient and convenient to use. Hence, the application of speech-based-technology is spreading in several directions. Walking hand in hand with the technology is the need of the hour. Very often the uneducated and elderly people are unable to use social media due to their lack in English knowledge. The fast advancement in technology makes it difficult for the older generation to cope up with. In a multi-linguistic country like India, where several languages are spoken, it becomes strenuous for the older and the illiterate to use English fluently. Lack of research in speech recognition in Odia language and this scenario motivated the research towards regional to standard speech translator, which can help the needy to match up with the modern technology.

The research regarding speech recognition has been going on since the last 8 decades. There has been at least 4 generations of different approaches and a new dawn of technology is on the rise. Dating back to the 18th century where the main focus was on vowel sounds and to interpret phonemes. This led to the invention of Dictation Machines by Thomas Edison in the late 19th century and was popular due to their ability to take notes on a daily basis. However, these machines focused only on speech recording and not on interpretation. One of the major breakthroughs happened in 1950's when Audrey, a machine created by Bell Labs, was able to interpret the digits (0-9) with 90% accuracy only when spoken by its inventor whereas recorded an accuracy of 70% when words were spoken by others. In 1971, Harpy, created by Carnegie Mellon University, was able to comprehend 1,011 words and some phrases. The main obstacle till date remained the noise and problems of inconsistency of words due to dialects, gender and speed.

Released in the mid-1980's, IBM Tangora [1], which required 20 minutes of trained data in the form of recorded speech was able to recognize up to 20,000 English words and full sentences. The breakthrough was the use of the Hidden Markov Model to increase flexibility through data clustering. In 1997, the world's first "continuous recognizer speaker", Dragon's Naturally speaking Software [2] was released. It was able to interpret 100 words per minute. The arrival of "Google Voice Search App for Iphone"[3] in the year 2008 was a major benchmark for the speech recognition technologies. Combining the power of latest technology and cloud computing, it draws knowledge about a speaker to be able to revert back. It was actually Siri, Apple's AI powered voice recognition system which engulfed the whole globe. Following that, we now have Cortana launched by Microsoft and Alexa launched by Amazon battling out to gain supremacy in the speech recognition field.

In India, speech recognition system has been adopted in many Indian languages. Samudravijaya has made ASR for railway reservation enquiry system in Hindi. For designing the system he has used 320 sentences and a vocabulary size of 161 words. Isolated Hindi recognition system was developed by Ishant Bhardwaj using HMM toolkit. For Telugu language Gautam developed an ASR system using sphinx for retrieving commodity prices using data that was obtained from 96 speakers and 500 words were collected for each of the speaker. In Assamese, M.P Sarma and K K Sarma has developed a numeral ASR system. Mandal has designed an ASR system in Bengali using SPHINX.

2 SPEECH RECOGNITION SYSTEM

2.1 DATA COLLECTION AND TRANSCRIPTION

We have collected the speakers data using mobile phones. Speakers chosen were of different demographics in respect to age, gender, religion, etc. as it was necessary for variation in the dataset. They uttered the same sample of words. A metadata sheet for each speaker was maintained which consisted of his/her name or ID, district name, age, gender, qualification, etc. We did the speech data transcription with splitting the sentences and phrases based audio into just one word based audio files and writing down the transcription.

The audio format used for speech data collections are:

- Bit rate: 128 kbps
- Bit depth: 16 bit
- Channel: Mono (1)
- Audio format: WAV

2.2 FEATURE EXTRACTION USING MFCC

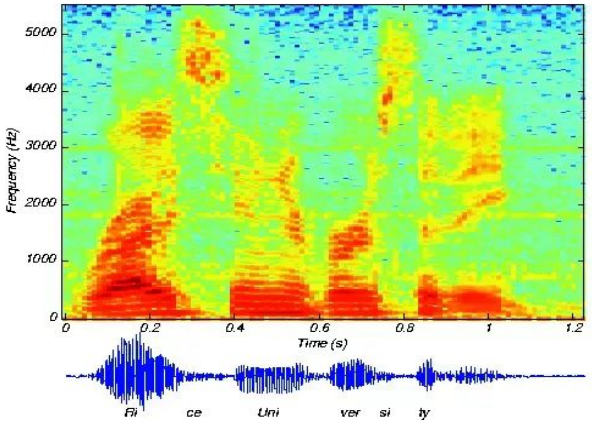
Audio processing systems work with rich, high-dimensional datasets encoded as vectors of the power spectral density coefficients for audio data. For speech recognition, the information required to successfully perform the task is encoded in the data (because humans can perform these tasks from the raw data). However, natural language processing systems traditionally treat words as discrete atomic symbols, and therefore 'cat' may be represented as 'ID-1' and 'dog' as 'ID-2'. These encodings are arbitrary, and provide no useful information to the system regarding the relationships that may exist between the individual symbols. This means that the model can leverage very little of what it has learned about 'cats' when it is processing data about 'dogs'. Representing words as unique, discrete ids furthermore leads to data sparsity, and usually means that we may need more data in order to successfully train statistical models. Using vector representations can overcome some of these obstacles.

Mel-frequency cepstral coefficients (MFCCs) [4] are coefficients that collectively make up an MFC. They are derived from a type of cepstral representation of the audio clip.

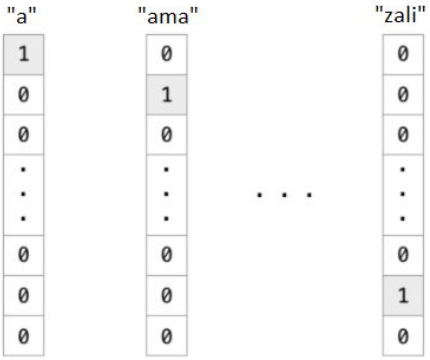
A signal goes through a pre-emphasis filter; then gets sliced into frames and a window function is applied to each frame; then, we do a Fourier transform on each frame and calculate the power spectrum; and subsequently compute the filter banks. To obtain MFCCs, a Discrete Cosine Transform (DCT) is applied to the filter banks retaining a number of the resulting coefficients while the rest are discarded. The final step is mean normalization.

LibROSA [5] is a python package for music and audio analysis. At a high level, librosa provides implementations of a variety of common functions used throughout the field of music information retrieval. It allows us to extract feature using MFCC from the audios.

Audio Spectrogram (Dense)

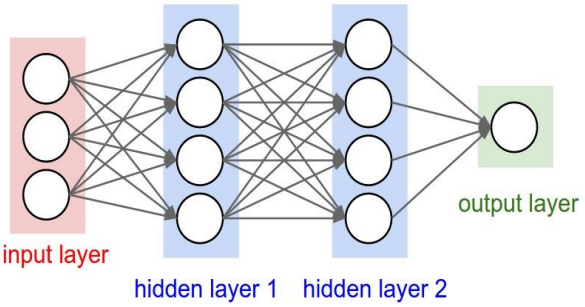


Word Vectors (Sparse)



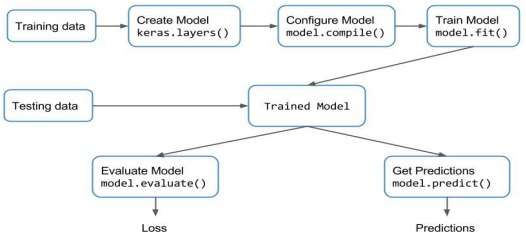
2.3 TRAINING DATA USING CNN

In deep learning, a **convolutional neural network** (CNN, or ConvNet)[7] is a class of deep[8], feedforward artificial neural network. CNNs use a variation of multilayer perceptrons designed to require minimal preprocessing. Neural Networks receive an input (a single vector), and transform it through a series of hidden layers. Each hidden layer is made up of a set of neurons, where each neuron is fully connected to all neurons in the previous layer, and where neurons in a single layer function completely independently and do not share any connections. The last fully-connected layer is called the "output layer" and in classification settings it represents the class scores.



Keras [6] is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. Keras is a deep learning library that allows for easy and fast prototyping (through user friendliness, modularity, and extensibility),

supports both convolutional networks and recurrent networks, as well as combinations of the two and runs seamlessly on CPU and GPU. Keras was initially developed as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System).



We used Keras to create and configure the CNN model and then trained it with the dataset of Odia words.

The CNN model was built and those arrays were fed to CNN which was trained for around 50 epochs (steps).

```
Train on 5 samples, validate on 4 samples
Epoch 1/50
5/5 [==============================] - 3s 628ms/step - loss: 3.2416 - acc: 0.6000 - val_loss: 8.1247 - val_acc: 0.25
Epoch 2/50
5/5 [==============================] - 0s 10ms/step - loss: 1.3835 - acc: 0.8000 - val_loss: 2.5657 - val_acc: 0.500
Epoch 3/50
5/5 [==============================] - 0s 6ms/step - loss: 4.0408 - acc: 0.4000 - val_loss: 2.5355 - val_acc: 0.2500
Epoch 4/50
5/5 [==============================] - 0s 6ms/step - loss: 2.3871 - acc: 0.6000 - val_loss: 1.6951 - val_acc: 0.5000
Epoch 5/50
5/5 [==============================] - 0s 6ms/step - loss: 2.4418 - acc: 0.6000 - val_loss: 3.3250 - val_acc: 0.5000
Epoch 6/50
5/5 [==============================] - 0s 6ms/step - loss: 6.3721 - acc: 0.2000 - val_loss: 1.1728 - val_acc: 0.5000
Epoch 7/50
5/5 [==============================] - 0s 5ms/step - loss: 0.4336 - acc: 0.8000 - val_loss: 3.1662 - val_acc: 0.2500
Epoch 8/50
5/5 [==============================] - 0s 5ms/step - loss: 2.5874 - acc: 0.6000 - val_loss: 0.8888 - val_acc: 0.5000
Epoch 9/50
5/5 [==============================] - 0s 6ms/step - loss: 1.3240 - acc: 0.6000 - val_loss: 0.8218 - val_acc: 0.5000
Epoch 10/50
5/5 [==============================] - 0s 6ms/step - loss: 0.1720 - acc: 1.0000 - val_loss: 1.0339 - val_acc: 0.5000
```

After the model has been trained, trained, the audio files were fed into the model's predict function to be predicted.

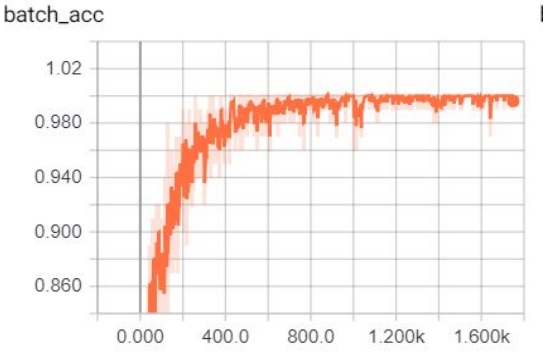
3 RESULTS AND DISCUSSION

This section represents the performance of our system. The system gets to 99.25% test accuracy after 12 epochs (there is still a lot of margin for parameter tuning). It takes 16 seconds per epoch on a GRID K520 GPU.

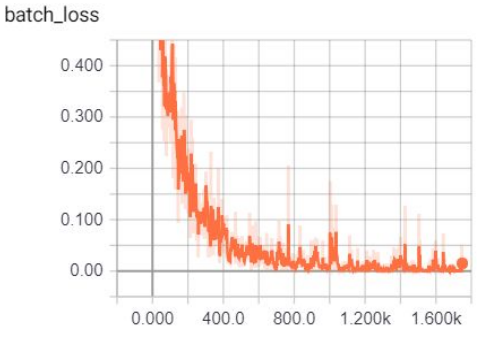
3.1 AUDIO CLIPS

No. of words	250
No. of clips per word	500
Training clips per word	350
Validation clips per word	140
Test clips per word	10

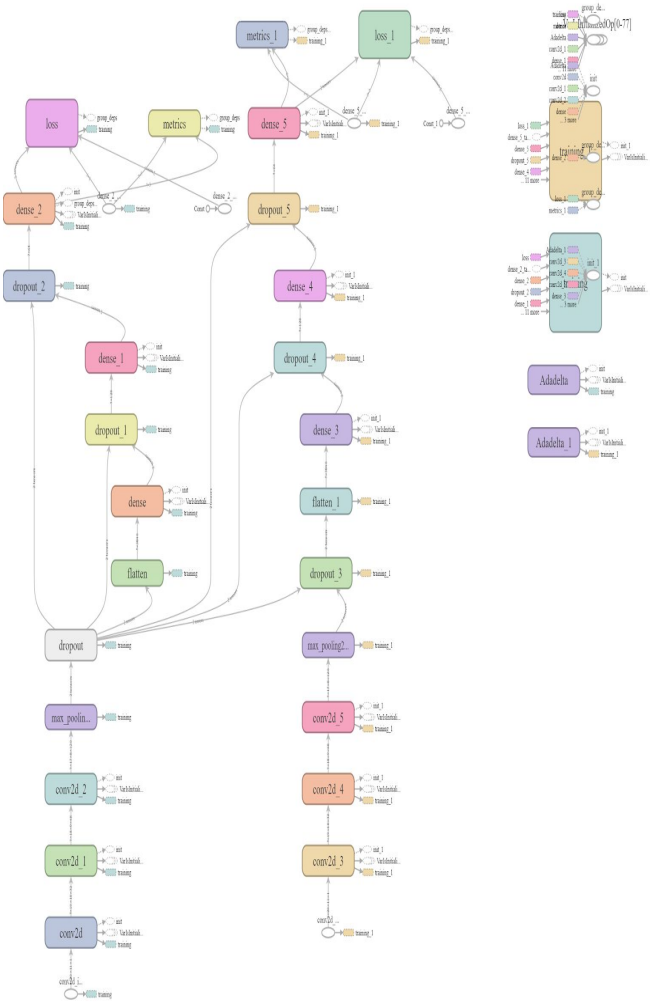
It is noticeable that the accuracy of batch increased with increase in steps of learning rate.



The loss rate decreased per batches with each iterations.



CNN Model:



4 CONCLUSION AND FUTURE WORK

In this paper, we present our Automatic Speech Recognition using Librosa and Keras toolkit for Odia language. To improve the system performance Long Short Term Memory (LSTM) can be used with CNN and Batch Normalization Layer [9] can be added. Batch normalization is a technique for improving the performance and stability of artificial neural networks.[4] It is a technique to provide any layer in a neural network with inputs that are zero mean/unit variance.It is used to normalize the input layer by adjusting and scaling the activations. The number of speech utterances for building the system was less, more data and variation is to be introduced and in presence of noses, better noise reduction technique can be implemented.

5 REFERENCE

[1] S. K. DasM. A. Picheny. "Issues in Practical Large Vocabulary Isolated Word Recognition: The IBM Tangora System"

[2] Baker, James K. (1975). "The DRAGON System - An Overview". IEEE Transactions on Acoustics, Speech, and Signal Processing.

[3] "Voice Search in Underrepresented Languages". Google Research Blog. November 9, 2010

[4] Sahidullah, Md.; Saha, Goutam (May 2012). "Design, analysis and experimental evaluation of block based

transformation in MFCC computation for speaker recognition"

[5] Librosa. https://github.com/librosa/librosa

[6] Keras. https://keras.io/

[7] LeCun, Yann; Bengio, Yoshua (1995). "Convolutional networks for images, speech, and time series". In Arbib, Michael A. The handbook of brain theory and neural networks (Second ed.). The MIT press. pp. 276–278.

[8] LeCun, Yann; Bengio, Yoshua; Hinton, Geoffrey (2015). "Deep learning". Nature.

[9] Ioffe, Sergey; Szegedy, Christian. "Batch Normalization: Accelerating Deep Network Training b y Reducing Internal Covariate Shift"