MUSIC GENRE CLASSIFICATION

USING RECURRENT CONVOLUTIONAL NEURAL NETWORKS

By

Meet Shah: 13D070003

Sarthak Daga:130070015

Nihar Mehta: 13D100011

Saumil Shah: 13D170001

Course Project- CS 725:
Foundations of Machine Learning
under
Prof Ganesh Ramakrishnan,
Department of Computer Science and Engineering,
Indian Institute of Technology, Bombay Powai,
Mumbai-400076 November, 2016.

Declaration:

This report comprises of solving a classification problem involving various Music Genres from the audio inputs, done as a part of ME 725, Foundations of Machine Learning course at IIT Bombay. This has been performed using Convolutional Recurrent Neural Networks- using Tensorflow framework. Due references have been provided at the end

CONTENTS:

.

1. PROBLEM DESCRIPTION

1.1 Introduction

Music is categorized into subjective categories called genres. Attribution of genre-tags to songs has been done manually. We have made an attempt to automate this task. Machine learning methods have been applied previously too for this problem. But we tried to apply the Recurrent Convolutional neural network, which is very slow to train, but gives very accuracte results. Genres are universal features of music that all forms have, regardless of scoring, form, rhythm or timbre. Genre classification, if automated shall reduce great amount of human work and time spent in doing so. It can further be used in recommender systems for recommending playlists of songs of a particular genre that a particular person may like.

1.2 Related Work

Robert O. Gjerdingen and David Perrott discovered that participants correctly matched the genre of a song 70 percent of the time after hearing the song for 3 seconds. Using Gaussian mixture models and diagonal covariance matrices, George Tzanetakis and Perry Cook achieved 61 percent classification accuracy with ten genres.

2. EXPERIMENT

2.1 Data Collection

The dataset has been downloaded from Marsyas (Music Analysis Retreival and Synthesis for Audio Simulation)

http://marsyasweb.appspot.com/download/data_sets/

The dataset consists of 1000 audio tracks each 30 seconds long. It contains 10 genres, each represented by 100 tracks. The tracks are all 22050Hz Mono 16-bit audio files in .wav format.

This dataset was used for the well known paper in genre classification " Musical genre classification of audio signals " by G. Tzanetakis and P. Cook in IEEE Transactions on Audio and Speech Processing 2002.

2.2 Reading Data

We first convert the data into spectrogram, which can be used as an input to the Recurrent Convolutional network (RCNN). Next, we extract the labels of the data. We split the data into 80% training and 20% validation datasets. One hot encoding is performed on the labels.

2.3 Preprocessing data

The data is preprocessed into a Melspectrogram.
An object of type MelSpectrogram represents an acoustic time-frequency representation of a sound: the power spectral density$P(f,t)$.It is sampled into a number of points around equally spaced times *ti* and frequencies *fj* (on a Mel frequency scale).

Mel-Frequency analysis of speech is based on human perception experiments
It is observed that human ear acts as filter. It concentrates on only certain frequency components. These filters are non-uniformly spaced on the frequency axis
More filters in the low frequency regions. Less no. of filters in high frequency regions

Mel stands for melody.
The following equation is used to convert the frequencies to mel-scale:

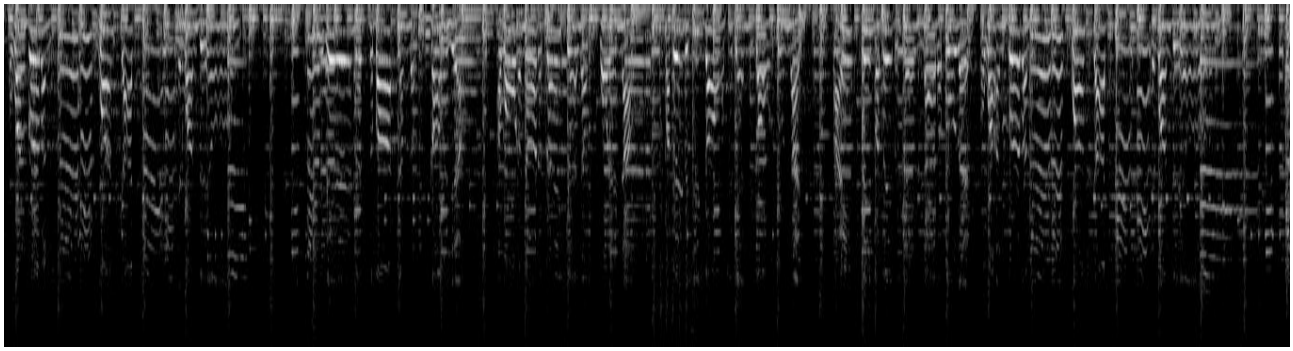
$$m = 2595 \log_{10}\left(1 + \frac{f}{700}\right)$$

Advantages:
Time-Frequency representation of the audio signal
Spectrogram is a tool to study audio sounds (phonemes)
Screeenshots of spectrogram act as 2D image inputs to the convolutional neural networks.

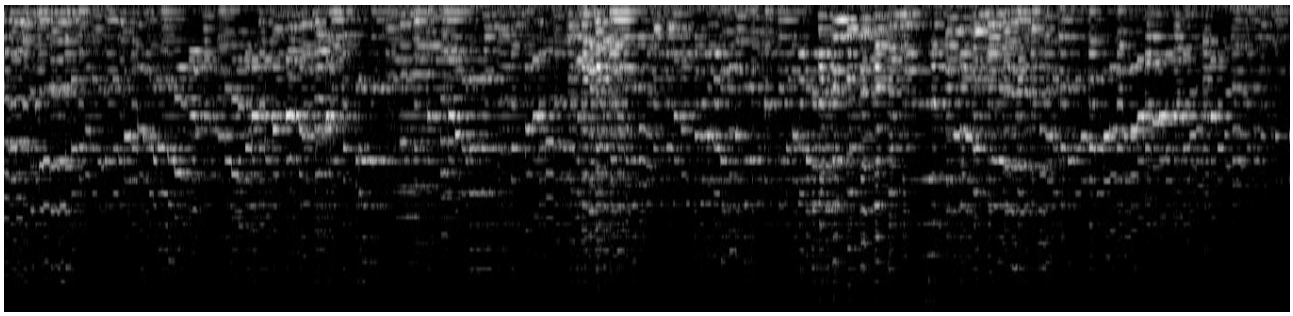
2.4 MELSPECTROGRAMS

Here are the melspectrograms of various types of the songs depending on their class:

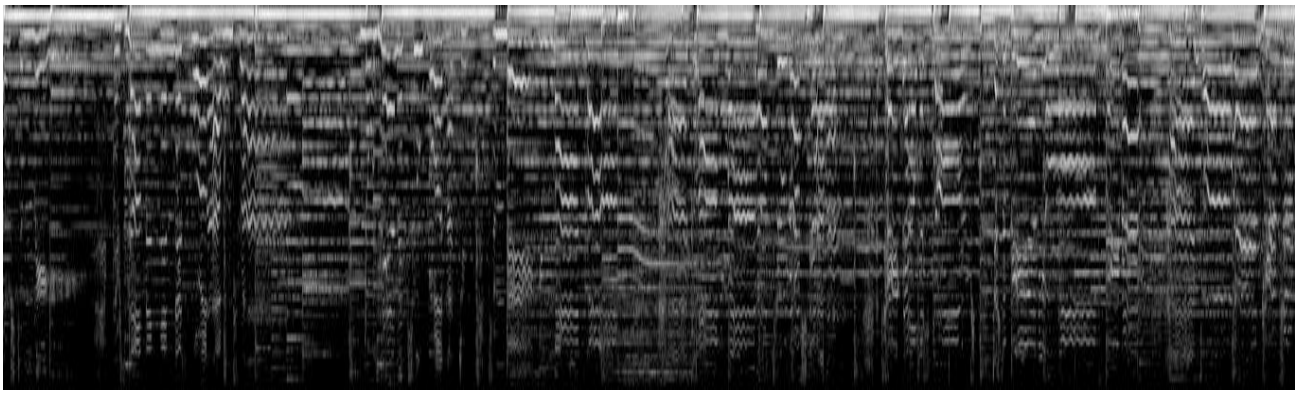
1. Blues:



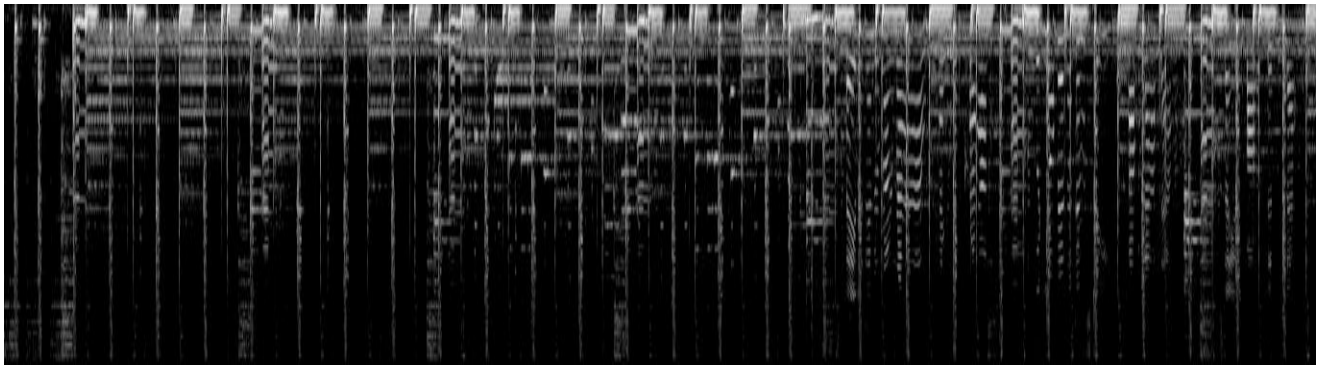
2. Classical:



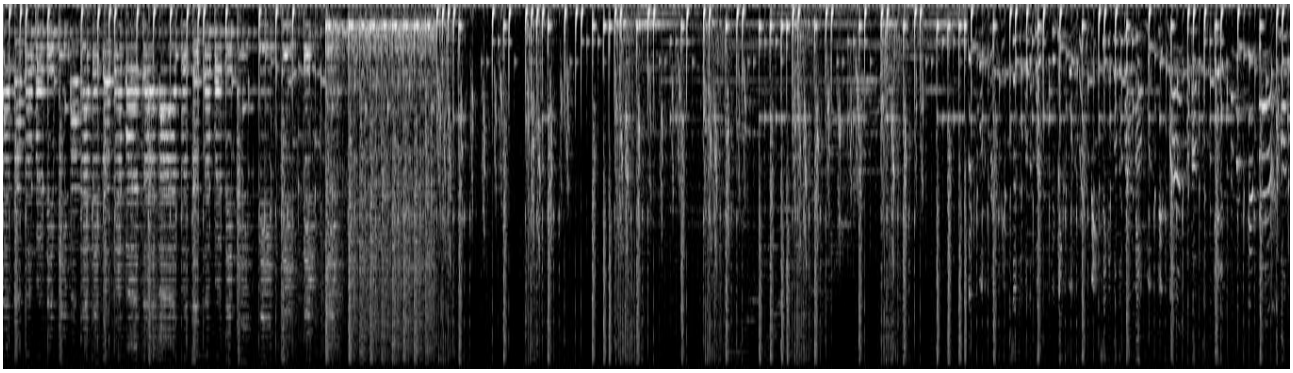
3.Country:



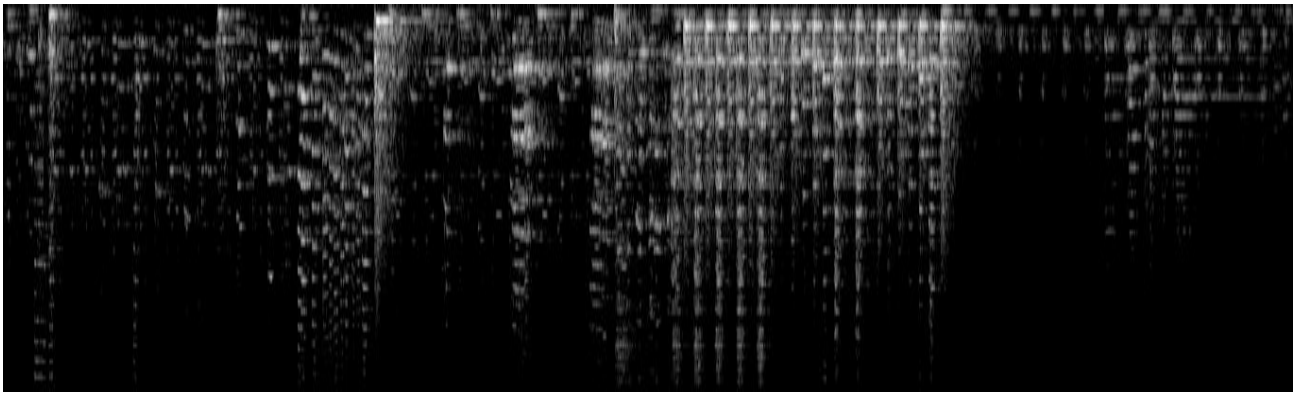
4. Disco:

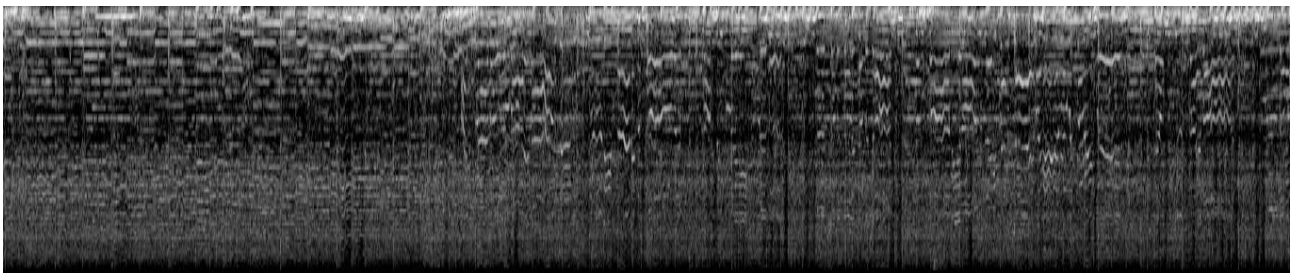


5. Hiphop:



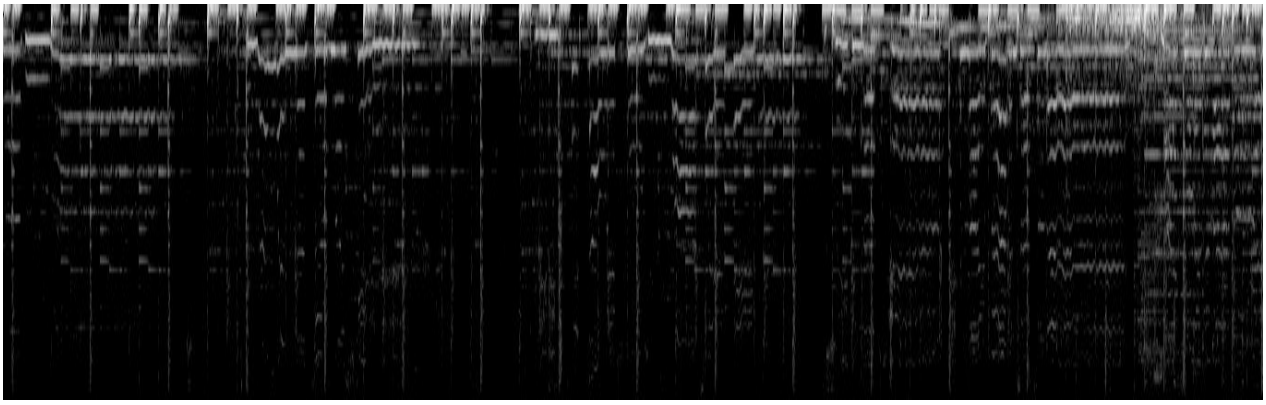
6.Jazz:



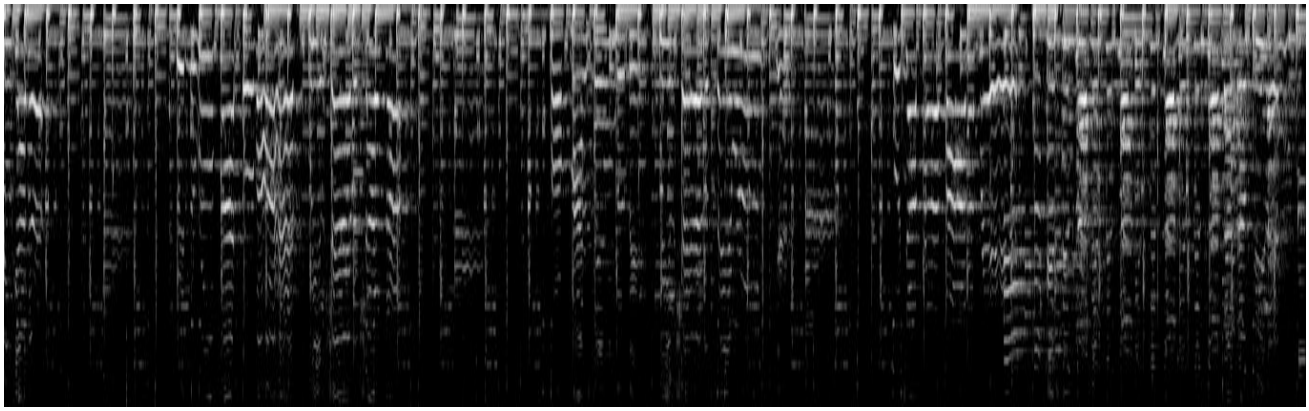


7. Metal:

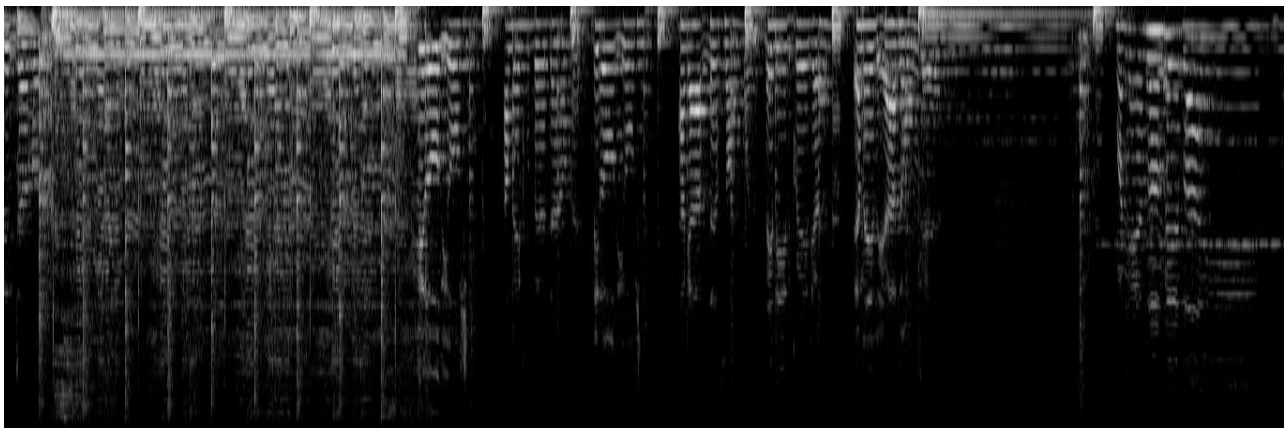
8. Pop:



9. Reggae:



10. Rock:



3. NEURAL NETWORK ARCHITECTURE

3.1 Convolutional Neural Networks:

Convolutional neural networks (LeCun, 1989) are a natural extension of neural networks for treating images. Their architecture, somewhat inspired by the biological visual system, possesses two key properties that make them extremely useful for image applications: spatially shared weights and spatial pooling.
Convolutional networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers.

3.2 Recurrent Neural Networks:

Recurrent neural networks or RNNs (Rumelhart et al, 1986a) are a family of neural networks for processing sequential data. Recurrent networks can scale to much longer sequences than would be practical for networks without sequence-based specialization. Most recurrent networks can also process sequences of variable length.

3.3 Recurrent Convolutional Neural Networks

The combination of recurrent convolutional neural networks enhances the accuracy of the model significantly. Recurrent neural networks are helpful for time series data. Convolutional neural networks work well on 2D static data. Here we have exploited the problem by converting the continous audio signal as a time varying melspectrogram and the network takes a screeshot of the melspectrogram as an input. The screenshot of the melspectrogram at any time can be considered as a 2D image. Filters are applied to the screenshot and final 2D tensor obtained is inputted in the model.

3.4 CONVOLUTIONAL PART OF NEURAL NETWORK

The model has 4 convolutional networks. Each convolutional network consists of a convolutional layer plus a max pooling layer plus a dropout layer.

Max pooling:

A typical layer of a convolutional network consists of three stages.
In the first stage, the layer performs several convolutions in parallel to produce a set of linear activations. In the second stage, each linear activation is run through a nonlinear activation function, such as the rectified linear activation function. This stage is sometimes called the detector stage. In the third stage, we use a pooling function to modify the output of the layer further.
A pooling function replaces the output of the net at a certain location with a summary statistic of the nearby outputs.

Dropout:

Dropout is a regularization technique for reducing overfitting n neural networks by preventing complex co-adaptations on training data. It is a very efficient way of performing model averaging with neural networks. The term "dropout" refers to dropping out units (both hidden and visible) in a network

3.5 RECURRENT PART OF NEURAL NETWORK:

The ouput of the last dropout layer is inputted into the recurrent neural network. It consists of 2 GRU cells., followed by a dropout

Finally a feedforward sigmoid layer is used to give the output as a list of 10 probabilities.
The highest probability corresponds to the class of the genre to which the music signal belonged to.

3.6 TRAINING

We have used tensorflow framework for writing the python code.

The input to the model is the screenshot of the spectrogram. It behaves as a 2D image and a convolutional neural net is very handy to deal with it. The outputs from the convolutional layer go into a recurrent neural network which makes sure that the past values are accumulated to get an intuition of what genre the song belongs to. In this problem, we need to have a continuous set of screenshots (time series data) to predict the classes and that explains the importance of a Recurrent Neural Network.

Loss Function

We have used the cross entropy as our loss function

Optimization algorithm:

We have used the RMSProp Algorithm as our optimization Algorithm:

Algorithm 6 RMSprop with Nesterov momentum

Require: Global learning rate η, decay rate ρ, momentum coefficient α

Require: Initial parameter θ, initial velocity v

Initialize accumulation variable $r = 0$

while Stopping criterion not met **do**

Sample a minibatch of m examples from the training set $\{x^{(1)}, \ldots, x^{(m)}\}$.

Compute interim update: $\theta \leftarrow \theta + \alpha v$

Set $g = 0$

for $i = 1$ to m **do**

Compute gradient:

$$g \leftarrow g + \nabla_\theta L(f(x^{(i)}; \theta)), y^{(i)}; \theta).$$

end for

Accumulate gradient: $r \leftarrow \rho r + (1 - \rho) g^2$

Compute velocity update: $v \leftarrow \alpha v - \frac{\eta}{\sqrt{r}} g$ ($\frac{1}{\sqrt{r}}$ applied element-wise)

Apply update: $\theta \leftarrow \theta + v$

end while

Batch Normalization

We have also used Batch Normalization on our data. It is simply normalization each activation to zero mean and unit variance. It has greatly helped to improve the accuracy of the model.
Some of the benefits of batch normalization are:

Fast learning: Learning rate can be increased compare to non-batch-normalized version.

Improved accuracy: Flexibility on mean and variance value for every dimension in every hidden layer provides better learning, hence accuracy of the network.

Normalization/whitening: Zero means, unit variances and or not decorrelated.

Solves the problem of internal covariate shifts: Transformation makes data to big or to small; change of the input distribution away from normalization due to successive transformation.

Not stuck in the saturation mode: Even if ReLU is not used.

Whitening within gradient descent: Requires inverse square root of covariance matrix as well as derivatives for backpropagation

Minibatch analysis

We train the data in minibatches of batch size =4 so that the whole memory doesnt accumulate once and we get an optimum between batch gradient descent and stochastic gradient descent.

4. RESULTS

1. When only the simple convolutional neural network is used, an accuracy of 64% is obtained after convergence.

2. When the convolutional neural network, coupled with recurrent neural network, this RCNN gives an accuracy of almost 87% after convergence.

5. CONCLUSIONS

We get a really good value of accuracy in case of an RCNN. It seems to have recognized the pattern in the spectrogram which helps it to identify the genre class.

However, it is very very slow to train. It takes about a day to train the RCNN. This is because recurrent neural nets are generally slow to train. Adding to that, the dimensionality of the spectrograms makes CNNs slow too. However, with the advancement of GPUs, we believe that these problems shall soon cease to exist and this method can be implemented for classifying various songs, videos etc which shall have various applications. Recommender systems can make use of these to identify what is the taste of the person with regards to the music he hears and accordingly, songs can be recommended to him.

The accuracy of an RCNN is higher than the accuracy of only a plain CNN without any recurrence or dependence on past inputs. This verifies the claim that we must use an RNN so as to capture the dependence on past inputs. The sound signal cannot be classified according simply the snapshots of the melspectrogram at various times

6. REFERENCES:

http://www.speech.cs.cmu.edu/15-492/slides/03_mfcc.pdf

http://marsyasweb.appspot.com/download/data_sets/

https://www.cs.swarthmore.edu/~meeden/cs81/s12/papers/AdrienDannySamPaper.pdf

http://www.deeplearningbook.org/