MUSIC GENRE CLASSIFICATION

Using Machine Learning Techniques

TABLE OF CONTENTS

1. Problem Statement
2. Data-Set
3. Data Pre-processing
4. Models Explored
5. Results and Discussion
6. Conclusion and Future Work

Honour Code:

We hereby affirm that we have upheld the highest principles of honesty and integrity while we worked on this project.

GROUP MEMBERS

MANAS JAIN (170040068)
NIKHIL CHOUDHARY (170110048)
HARSHITA MASAND (170110064)

PROBLEM STATEMENT

Music Genre Classification using Machine Learning and Deep Learning Techniques

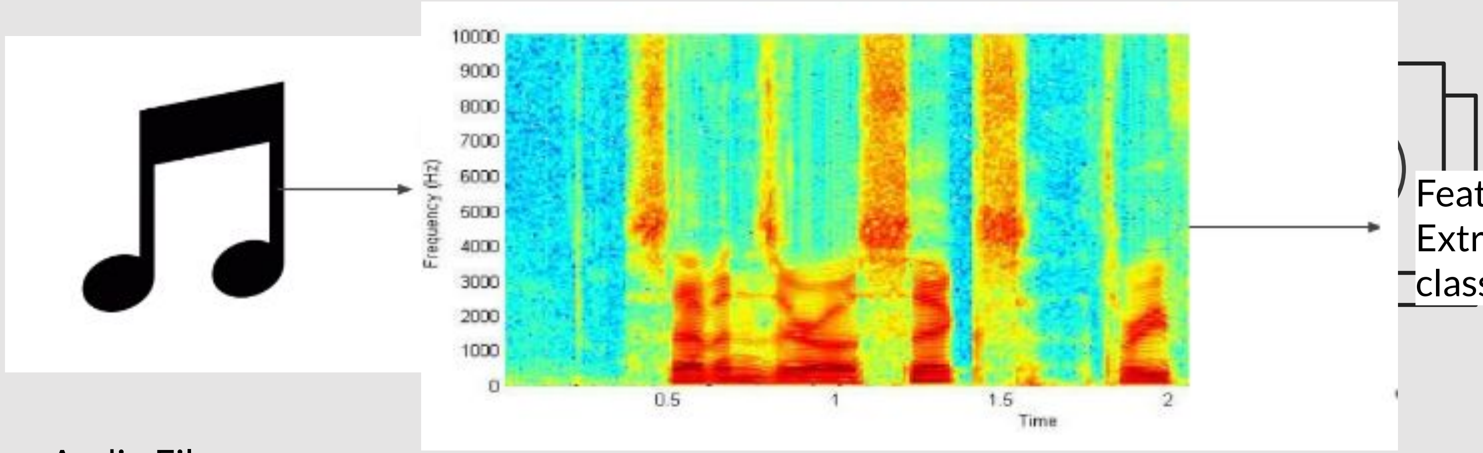
Genre classification is an important task with many real world applications. As the quantity of music being released on a daily basis continues to sky-rocket the need for accurate meta-data required for database management and search/storage purposes climbs in proportion.

INTRODUCTION

Categorizing music files according to their genre is a challenging task in the area of music information retrieval (MIR). In this project, we implemented four different classification algorithms admitting the input - transformed mel-spectrograms of our raw amplitude data. We implemented the following algorithms :

- K-Nearest Neighbours
- Support Vector Machine
- Feed Forward Neural Network
- Convolutional Neural Network

GENERAL ALGORITHM



Feature Extraction and classification

Audio File

Mel Spectrogram

BACKGROUND

2002

The mixture of Gaussians model and k-nearest neighbors was used. It achieved 61% accuracy [1]

2003

Multiple layers of SVM achieved upto 90% accuracy. [2]

2012 - PRESENT

CNN prove to be incredibly accurate music genre classifiers with an accuracy of around 91% [3]

DATASET

We used labeled public GTZAN dataset (1k 30s audio clips). From each clip, we sampled 2s window at 8 random location, thus augmenting our data to 8000 clips of two seconds each. Data was sampled at 22050Hz, this leaves us with 44100 features.

Training Data	8000
Testing Data	100
Cross-Validation	100

From Kaggle, we imported 2 CSV files - One file has for each song (30 seconds long) a mean and variance computed over multiple features that can be extracted from an audio file. The other file has the same structure, but the songs were split before into 3 seconds audio files (this way increasing 10 times the amount of data we fuel into our classification models).

No. of samples*No. Of features	9990*58
Testing data	3296 (33%)
Training data	6694 (67%)

PRE - PROCESSING

- Converting the raw audio into mel-spectrograms
- Mel-spectrograms are a commonly used method of featurizing audio because they closely represent how humans perceive audio i.e. in log frequency
- For this one must apply short-time Fourier transforms across sliding windows of audio, most commonly around 20ms wide

$$\mathbf{STFT}\{x[n]\}(m,\omega) = \sum x[n]w[n-m]e^{-j\omega n}.$$

. where signal x[n], window w[n], frequency axis ω, and shift m computed using sliding DFT algorithms

PRE - PROCESSING

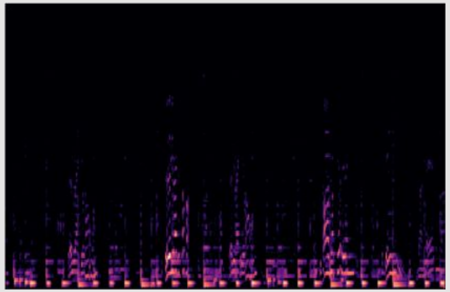
- STFT are then mapped to the mel scale by transforming the frequencies f by

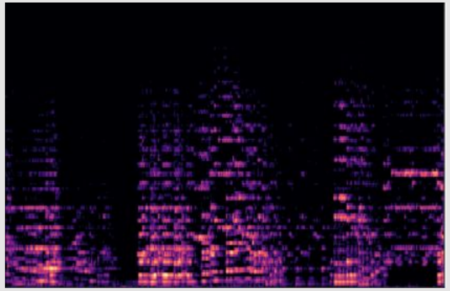
$$m = 2595 \log_{10}(1 + \frac{f}{700}).$$

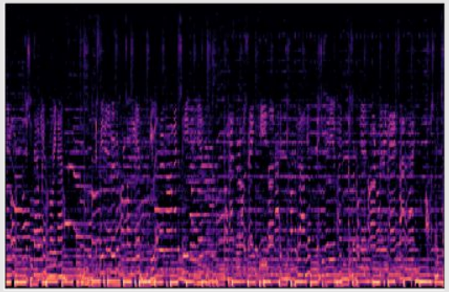
- Then we take the discrete cosine transform of the result (common in signal processing) in order to get our final output mel-spectrogram
- We used Librosa library and chose to use 64 mel-bins and a window length of 512 samples with an overlap of 50% between windows

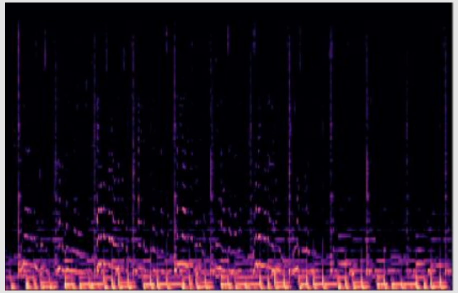
  Shape of our data (8000,64,173)

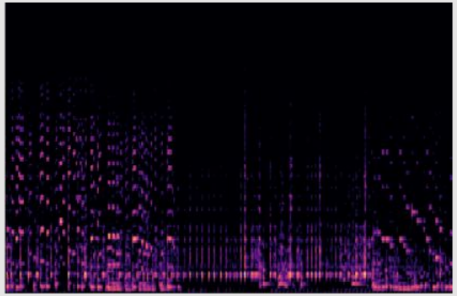
SAMPLE MEL SPECTROGRAMS OF OUR PRE-PROCESSED DATA

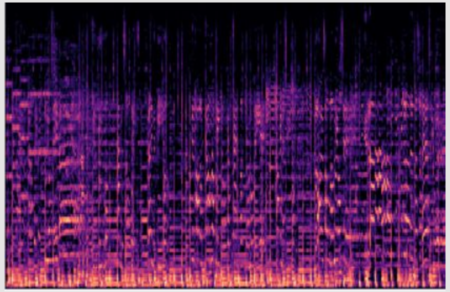

Blues


Classical


Country


Rock


Jazz


Pop

MODELS IMPLEMENTED

SVM

We created a model with RBF Kernel. This gave a baseline accuracy to compare with the Deep-Learning Models.

K-NN

Post PCA, we used k=10 and distance weighting.

FEED FORWARD NEURAL NETWORK

Our network has 3-fully connected layers, each using ReLU activation. We use softmax o/p with cross-entropy loss and Adam optimisation.

CNN

Adam optimisation with ReLU Activation.

PRINCIPAL COMPONENT ANALYSIS

- PCA to reduce dimension for two of our models (k-NN & SVM)
- In order to do this, we first transformed the mel-spectrograms by normalizing their mean and variance

$$\frac{1}{m}\sum_{i=1}^{m}(x^{(i)^T}u)^2 = \frac{1}{m}\sum_{i=1}^{m}x^{(i)}x^{(i)^T}.$$

m

- We had best results reducing to 15 dimensions, analogous to taking the top 15 eigenvectors of Σ and projecting our data onto the subspace spanned by them. We implemented this using scikit-learn in python.

K - NN

- Since there are 10 genres we got best accuracy from K=10
- Those which return the largest value on $||x - x^{(i)}|| 2$, where $|| \cdot ||^2$ denotes the Euclidean distance between points, be the 10 closest neighbours to x.
- We choose weights w_i for each i such that:

$$w_i \propto ||x - x^{(i)}||_2, \quad \sum_{i=1}^{1} 0 w_i = 1.$$

$$\arg\max_y \sum w_i \mathbb{1}(y = y^{(i)}),$$

- Finally we return label which is most prevalent among x’s 10 nearest neighbors when weighted by distance. This was implemented using scikit-learn in python.

K-NN - RESULTS

Accuracy on Training Data	1
Accuracy on CV	0.53
Accuracy on Testing Data	0.54

SUPPORT VECTOR MACHINE

To make our data linearly separable, we used the following kernel function:

$$\min_{\gamma,w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}\xi_i \quad \text{s.t. } y^{(i)}(\sum_j \alpha_j K(x^{(j)}, x^{(i)}) + b) \geq 1 - \xi_i \text{ and } \xi_i \geq 0$$

where $x^{(i)}$ are examples, α_j , b are weights and biases, C is a penalty parameter, and $1 - \xi_i$ is the functional margin for example i. $K : R^n \times R^n \rightarrow R$ is a kernel function. We used the RBF Kernel Function:

$$K(x^{(j)}, x^{(i)}) = \exp\left(-\frac{||x^{(j)} - x^{(i)}||_2^2}{2\sigma^2}\right)$$

This kernel, also sometimes called the Gaussian kernel, corresponds to an infinite dimensional feature space is related to Euclidean distance. This function as a whole is often minimized using sequential minimization optimization.

SVM - RESULTS

Accuracy on Training Data	0.59025
Accuracy on CV	0.5
Accuracy on Testing Data	0.5

This model was implemented using Scikit-Learn in Python.

FEED -FORWARD NEURAL NETWORK

We used a fully connected neural network as well, with ReLU activation and 6 layers, with cross-entropy loss. As the input to our model was 1D, when using mel-spectrograms, we flattened the data. Our model is fully connected, which means each node is connected to every other node in the next layer. At each layer, we applied a ReLU activation function to the output of each node, following the formula:

$$\mathrm{ReLU}(x) = \begin{cases} x & x \geq 0 \\ 0 & x < 0. \end{cases}$$

we construct a probability distribution of the 10 genres by

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

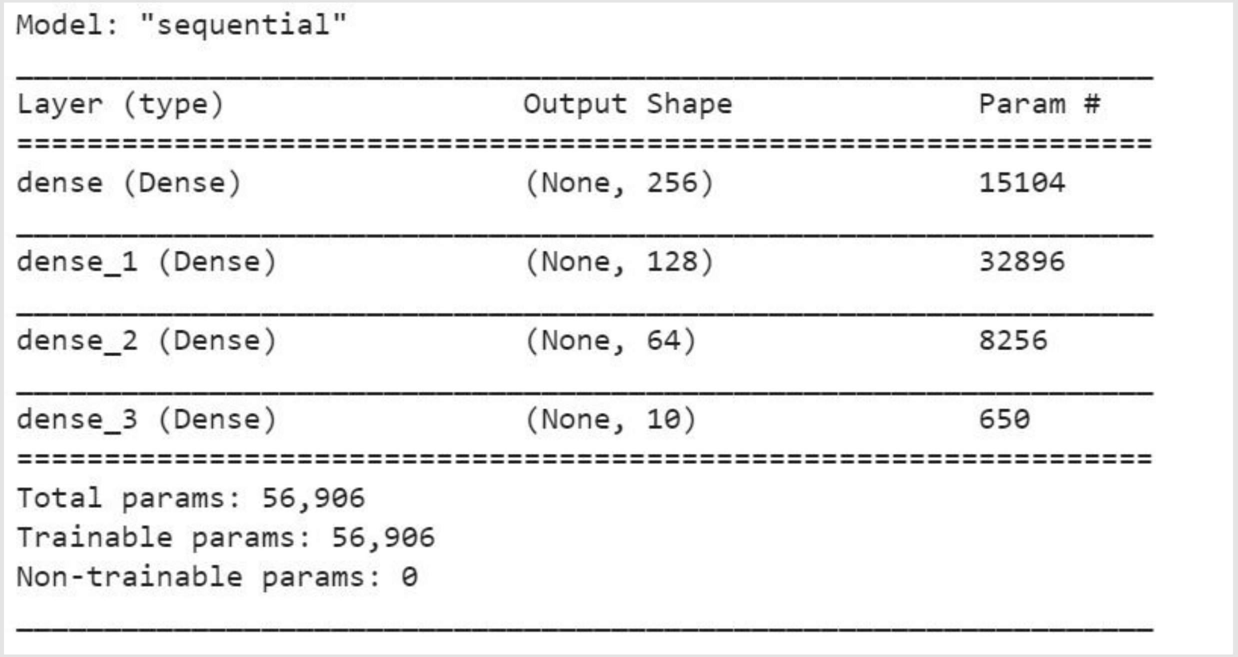
To optimize our model, we minimized cross entropy loss:

$$CE(\theta) = -\sum_{x \in X} y(x) \log \hat{y}(x)$$

This was implemented with TensorFlow

FEED FORWARD NN - ARCHITECTURE

The model has been implemented using Keras and Tensor Flow.



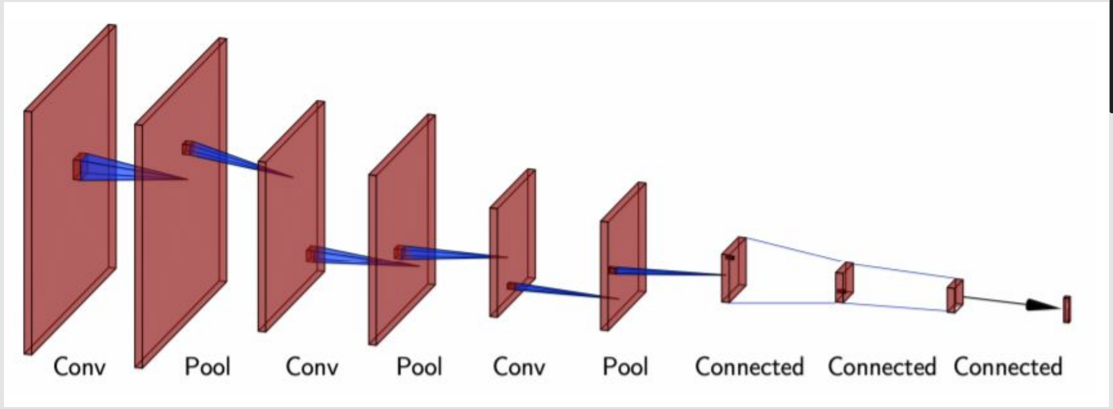
```
Model: "sequential"

_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 256)               15104
_________________________________________________________________
dense_1 (Dense)              (None, 128)               32896
_________________________________________________________________
dense_2 (Dense)              (None, 64)                8256
_________________________________________________________________
dense_3 (Dense)              (None, 10)                650
=================================================================
Total params: 56,906
Trainable params: 56,906
Non-trainable params: 0
_________________________________________________________________
```

FEED FORWARD NN - RESULTS

Accuracy on Training Data	0.9994
Accuracy on Testing Data	0.8893

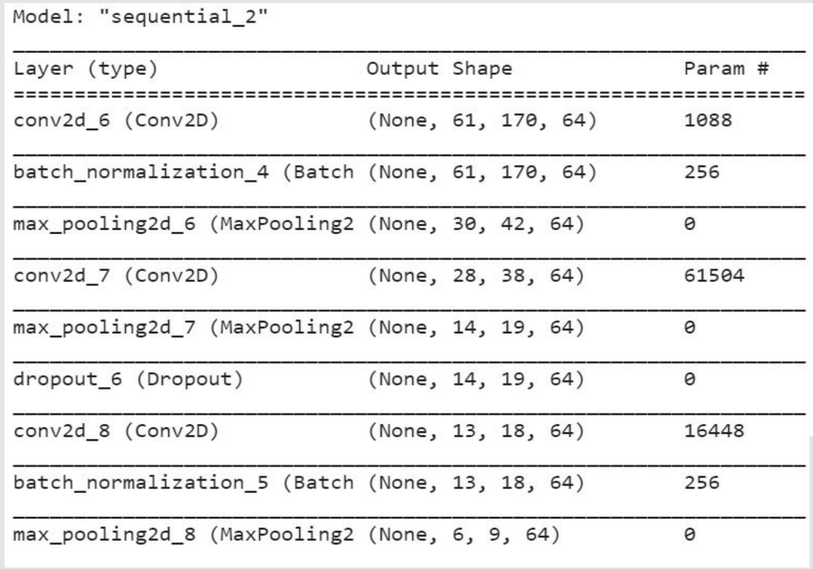
CONVOLUTIONAL NEURAL NETWORK

This was our most advanced model, using 3 convolution layers, each with its own max pool and regularization, feeding into 3 fully connected layers with ReLU activation, softmax output, and cross entropy loss.

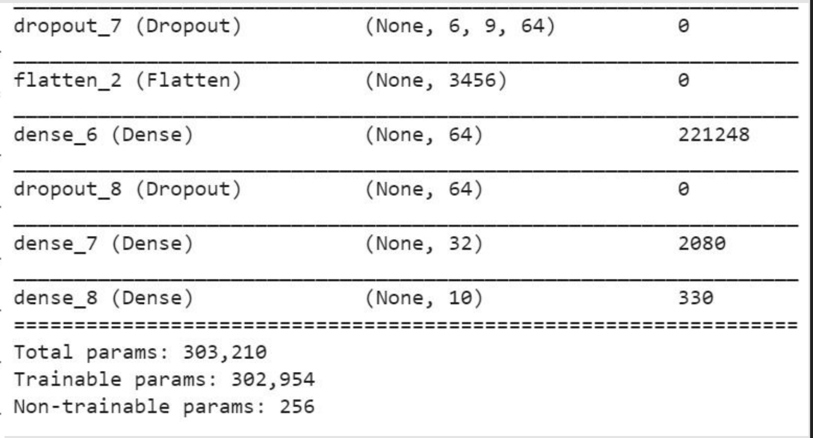


CONVOLUTIONAL NEURAL NETWORK- IMPLEMENTATION DETAILS

As the accuracy in our Cross-Validation crossed 80%, the model saved the confusion matrix. This happened for us at No. of Epochs=150, Batch size:=200



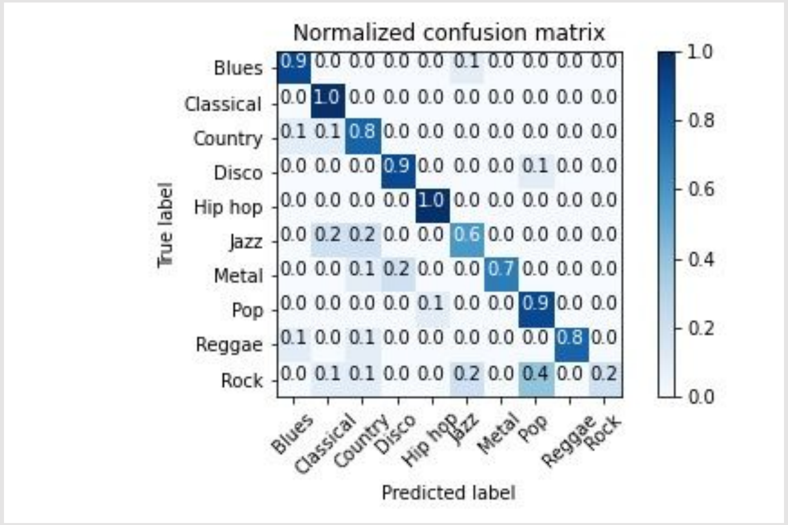
```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d_6 (Conv2D)            (None, 61, 170, 64)       1088      
_________________________________________________________________
batch_normalization_4 (Batch (None, 61, 170, 64)       256       
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 30, 42, 64)        0         
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 28, 38, 64)        61504     
_________________________________________________________________
max_pooling2d_7 (MaxPooling2 (None, 14, 19, 64)        0         
_________________________________________________________________
dropout_6 (Dropout)          (None, 14, 19, 64)        0         
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 13, 18, 64)        16448     
_________________________________________________________________
batch_normalization_5 (Batch (None, 13, 18, 64)        256       
_________________________________________________________________
max_pooling2d_8 (MaxPooling2 (None, 6, 9, 64)          0         
```



```
_________________________________________________________________
dropout_7 (Dropout)          (None, 6, 9, 64)          0         
_________________________________________________________________
flatten_2 (Flatten)          (None, 3456)              0         
_________________________________________________________________
dense_6 (Dense)              (None, 64)                221248    
_________________________________________________________________
dropout_8 (Dropout)          (None, 64)                0         
_________________________________________________________________
dense_7 (Dense)              (None, 32)                2080      
_________________________________________________________________
dense_8 (Dense)              (None, 10)                330       
=================================================================
Total params: 303,210
Trainable params: 302,954
Non-trainable params: 256
```

CONVOLUTIONAL NEURAL NETWORK- RESULTS

Training Accuracy	0.932250
Cross-Validation Accuracy	0.843750
Testing Accuracy	0.828125



This model was implemented using tensorflow and keras in Python. Google Colab free GPU was used for training the model.

RESULTS AND DISCUSSION

- Since many rock music excerpts lack the easily visible beats that other genres such as hip-hop and disco possess, we found by looking at our confusion matrix that our CNN managed to correctly classify 50% of rock audio as rock, labeling the others as mainly country or blues. Additionally, it incorrectly classified some country, as well as a small fraction of blues and reggae, as rock music.
- We also noted that correct classification of jazz was less accurate than most other categories. Our algorithm falsely classified some jazz music as classical, although never did the reverse of this.
- The accuracy achieved by FFNN is 88%, which is also good given the huge amount of data we have. However, it requires feature extraction and used a large number of features that made this computationally heavy. Hence, CNN is preferred.

CONCLUSION & FUTURE WORK

- CNN performed the best, as we expected. It took the longest time to train as well, but the increase in accuracy justifies the extra computation cost.
- In the future, given that this is time series data, some sort of RNN model may work well (GRU, LSTM)
- Also we may have opportunities for transfer learning, for example in classifying music by artist or by decade.

CONTRIBUTIONS

MANAS JAIN

CNN+ SVM

HARSHITA MASAND

k-NN+ PPT

NIKHIL CHOUDHARY

Feed Forward Neural Network+ PPT

THANK YOU !

Please keep this slide for attribution

CREDITS: This presentation template was created by **Slidesgo,** including icons by **Flaticon**, infographics & images by **Freepik**

REFERENCES

1. G. Tzanetakis and P. Cook. Musical genre classification of audio signals. IEEE Transactions on Speech and Audio Processing,
   10(5):293–302, July 2002.
2. G. Tzanetakis and P. Cook. Musical genre classification of audio signals. IEEE Transactions on Speech and Audio Processing,
   10(5):293–302, July 2002.
3. Y. Panagakis, C. Kotropoulos, and G. R. Arce. Music genre classification via sparse representations of auditory temporal
   modulations. In 2009 17th European Signal Processing Conference, pages 1–5, Aug 2009.