Music Genre Recognition using Machine Learning

Ayush K. Rai
Ecole CentraleSupelec
Paris, France
ayush.rai2512@student-cs.fr

Louis De Vitry
Ecole CentraleSupelec
Paris, France
louis.devitry@student-cs.fr

Alami C. Mohamed
Ecole CentraleSupelec
Paris, France
m.alamichehboune@student-cs.fr

ABSTRACT

In this work we present a technique for automatic music genre recognition using various machine learning techniques on Free Music Archive Dataset [5]. It is important to note that in this dataset track genres have parent/child hierarchical relationship and a certain song can have multiple labels. We therefore address this problem as multiclass classification problem instead of multilabel classification problem to reduce the complexity of the task by considering only the parent genre of every audio and ignoring the child of that parent. In addition our efforts are mostly concentrated on supervised machine learning and deep learning methods to address this problem. However we also dive into few unsupervised machine learning techniques. Along with the results of our ML models, we perform extensive and detailed analysis of feature engineering process, interpretation of confusion matrix for different classifiers and an in depth examination of convolutional neural networks.

1 INTRODUCTION

Music Information Retrieval (MIR) is a field that involves retrieval of useful information from music and has many real world applications. Music Genre Recognition is one such application, which over the years has received a lot of attention not only from MIR research community but also from giant tech firms. Nowadays companies like Spotify and SoundCloud recommend music to their customers using personalized music recommender systems and music genre recognition is an important component of building those recommendation engines. Another application of music genre recognition might be to use the predicted genres or sub-genres and combine them with the music meta-data and acoustic features (extracted using signal processing techniques) in order to group similar songs together. Finally music genre recognition can also help a music artist or music composer to understand which music genre is popular among different section of the audience.

Therefore in this work we address the problem of Music Genre Recognition as a multiclass classification task. The approach that we adopted to tackle this task involves a combination of supervised machine learning methods, unsupervised and deep learning methods.

The rest of the report is structured as follows: In Section 2 related topics and works are discussed. Section 3 describes the dataset in detail followed by an explanation on feature engineering in Section 4. In Section 5, we describe the supervised machine learning techniques, unsupervised machine learning methods and Deep learning based methods we used in our work, which is accompanied by the evaluation table. In Section 6 and Section 7 we explain the experiments we perform and the results we observe. Finally in Section 8, we provide conclusive remarks and also mention future directions of our work.

2 RELATED WORK

Music Genre Recognition is certainly not a novel problem in the field of Music Information Retrieval (MIR) and many studies and research papers have been published on tackling this problem using the whole spectrum of machine learning methods. [25] addressed this problem with machine learning approaches such as Gaussian Mixture model and k-nearest neighbour classifiers. They introduced 3 sets of features for this task categorized as timbral structure, rhythmic content and pitch content. Spectral Contrast and Spectral Rolloff were some of the features used by them. [15] proposed a feature extraction method based on wavelet coefficients histogram and published an study on this multi-class learning problem. [16] discuss the importance of psycho-acoustic features for recognizing music especially the importance Short Time Fourier Transform on Bark Scale. In the pre-deep learning era, Mel Frequency Cepstral Coefficients (MFCC) based features were a prominent way of addressing this task. [10] trained a Hidden Markov Model (HMM) model using MFCC features. [20] tackle the genre classification problem by trying to improve the measurement of audio similarities .[13] worked on the problem of audio music mood classification by empirically selecting multiple descriptors to extract spectral, temporal, loudness features and further trained a SVM (Support Vector Machine Model) to perform classification. [19] combined visual and acoustic features to train a SVM and adaboost classifier to predict the genres of the songs. There are other methods that can be used in order to classify music that were not used in this project such as the Octave-Based Spectral Contrast (OSC) or Octave-Based Modulation Spectral Contrast (OMSC) as presented in [14]. [12] presented a method that relies on a language modeling approach and takes in account the temporal information of the music signals for genre classification.

With the rise of deep learning algorithms many researchers [3] have used Convolutional Neural Networks (CNNs) and Long Short-Term memory (LSTMs) based models over traditional machine learning algorithms to attack the problem of music genre classification. [26] proposed that spectrograms can be considered as images and can be further used to train a convolutional neural network. Representing audio in the time domain for input to neural networks is not very straight-forward because of the high sampling rate of audio signals. [4] outlines the deep networks architecture that have been successful in MIR tasks and in turn facilitate the selection of the building blocks for many problems in music information retrieval tasks.[18] extracted numerical features like low level average loudness, low level spectral flux median and trained ensemble classifiers like gradient boost classifier on those features whereas trained a Convolutional Neural Network on the corresponding spectrograms of the audio tracks.

3 DATASET AND TASK DESCRIPTION

Other fields of artificial intelligence like computer vision have many established and benchmark datasets like ImageNet and MSCOCO but in MIR there has been always been a dearth of such large scale datasets. In ISMIR 2017 Free Music Archive dataset [5] was published to aid this issue. The data includes:

- Audio track (encoded as mp3) of each of the 106,574 tracks. It is on average 10 millions samples per track.
- Audio features like **Zero Crossing Rate, Constant Q Chromagram, Chroma STFT Chroma CQT, Chroma Energy Normalized, Tonnetz, Spectral Centroid, Spectral Bandwidth, Spectral Contrast, Spectral Roll Off, RMSE, Mel Frequency Cepstral Coefficient** (consisting of 518 attributes with statistics such as mean, standard deviation, skew, kurtosis, median, minimum, maximum) for each of the 106,574 tracks.
- Metadata provided with the dataset includes song title, album, artist, genres; play counts, favorites, comments; description, biography, tags. However in this work we have not taken into account the metadata.
- The dataset is split into four sizes: small (8 balanced genres), medium (16 unbalanced genres), large (161 unbalanced genres), full (161 unbalanced genres). In this work we have focussed on fma small dataset only including parent genres like **Hip Hop, Pop, Folk, Experimental, Rock, International, Electronic and Instrumental.**

3.1 Task Description

- We address the problem of music genre recognition as multiclass classification problem.
- We apply supervised and unsupervised machine learning methods and also perform some analysis on deep learning based methods to tackle the problem.
- It is important to understand that in this dataset certain genres have parent child relationship. Therefore in order to lessen the complexity of the task, we convert this problem from multi-label classification problem to multi-class classification problem by only considering the parent genre of every audio track and ignoring the child genres of that parent.

4 FEATURE ENGINEERING

4.1 Audio feature engineering

The raw data at our disposal consists of *.mp3 files and is therefore not directly workable for Machine Learning models. We present here how we extract informations from these files.

4.1.1 Extracting a floating point time series. We proceed by loading each *mp3 file as a floating point signal resampled to the given rate (sr=22050). This signal will be used in the following features.

4.1.2 Typical use of the features. Most features discussed below are applied on successive frames of the song (time slices) and are then processed with different techniques depending on the context:

(1) *Machine Learning:* The high dimensions of the raw data (source of more than 22k time points) must be addressed for all input variables. We proceed with a dimension reduction step to this humongous set of features by using the first order features, which are usual statistics (mean, median, std, skew, kurtosis, min, max) that explains well the time dimension. Furthermore, the frequency-based features are binned and the aforementioned statistics are applied over the bins.
The MFCC is special as it outputs coefficients containing both the time and frequency informations.

Feature	Original features	Statistics	Total
RMSE	1	7	7
Spectral bandwidth	1	7	7
Spectral centroid	1	7	7
Spectral rolloff	1	7	7
ZCR	1	7	7
Tonnetz	6	7	42
Chroma stft	12 bins	7	84
Chroma cqt	12 bins	7	84
Chroma cens	12 bins	7	84
MFCC	139	NA	139

(2) *Deep Learning:* Refer to the Deep Learning section.

4.1.3 Root Mean Squared Energy. The energy of a signal corresponds to the total magnitude of the signal, namely:

$$\sum_{n=1} x(n)^2 \text{ or } \sqrt{\frac{1}{N}\sum_{n=1} x(n)^2}$$

For audio signals, that roughly corresponds to how loud the signal is. It is common practice to take its Root-Mean-Square version (right formula). We plot below the Root-Mean-Square Energy for an extract of Queen - We are the champions. This extract will be used throughout the feature engineering part.

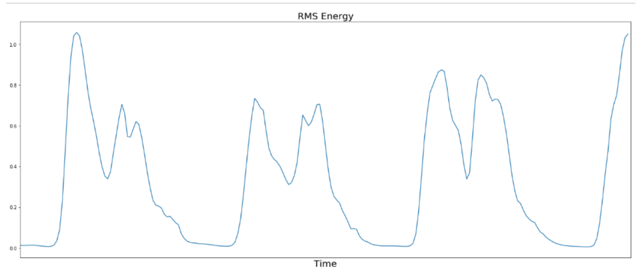


Figure 1: Root Mean Squared Energy of our audio sample

4.1.4 *Spectral centroid.*

The spectral centroid is a measure used in digital signal processing to characterise a spectrum. It indicates where the "center of mass" of the spectrum is located. Perceptually, it has a robust connection with the impression of "brightness" of a sound.
It is calculated as the weighted mean of the frequencies present in the signal, determined using a Fourier transform, with their magnitudes as the weights:

$$\mathrm{C} = \frac{\sum_{n=1}^{N} f(n)x(n)}{\sum_{n=1}^{N} x(n)}$$

where $x(n)$ represents the weighted frequency value, or magnitude, of bin number n, and $f(n)$ represents the center frequency of that bin.

4.1.5 *Spectral bandwidth.*

Spectral width is the wavelength interval over which the magnitude of all spectral components is equal to or greater than half of the magnitude of the component having the maximum value.

Mathematically, it is the weighted distance between each frequency and the centroid. The weights are the spectrogram magnitude. The p-th order spectral bandwidth for each frame:

$$\text{Spectral bandwidth}_p = (\sum_{n=1}^{N} x(k) * (f(k) - C)^p)^{1/p}$$

4.1.6 *Spectral rolloff.*

This is a measure measure of the amount of the right-skewedness of the power spectrum. Mathematically, it is the frequency R_t below which 85% of the magnitude distribution is concentrated:

$$\sum_{n=1}^{R_t} x_t(n) = 0.85 \cdot \sum_{n=1}^{N} x_t(n)$$

where $x_t(n)$ is the cumulative energy of the spectrum of frame t evaluated at frequency n.

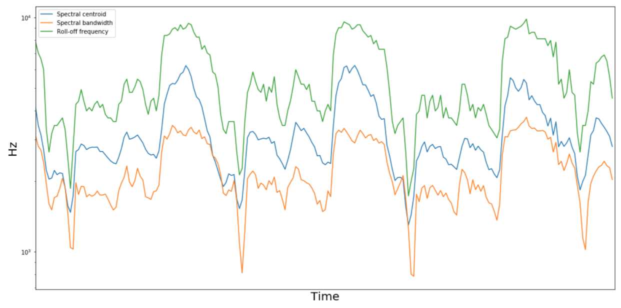


Figure 2: Spectral Centroid, Rolloff and Bandwidth) are key metrics to characterize a spectrum evolution. They are however highly correlated

4.1.7 *Spectral contrast[8].*

In most modern musics, strong spectral peaks correspond to harmonic components; while non-harmonic components, or noises, often appear at spectral valleys. Instead of averaging the spectral envelope like the MFCC does, the spectral contrast compute both spectral peak and valleys, thereby reflecting the distribution of harmonic and nonharmonic components.

To compute the spectral contrast, we proceed by:

1. Short-Time FFT is applied to the signal
2. Apply Octave-scale filters on the spectrum
3. The strength of spectral peaks, valleys, and their differences are estimated in each sub-hand.
4. Convert to log domain
5. Apply Karhunen-Loeve transform (K-L)

In the theory of stochastic processes, the KarhunenâĂŞLoÃĺve theorem is a representation of a stochastic process as an infinite linear combination of orthogonal functions, analogous to a Fourier series representation of a function on a bounded interval.

From a standpoint of eliminating relativity, K-L transform and DCT transform are equivalent. The K-L transform has however the benefit of yielding orthogonal base vectors. It should be noticed that the orthogonal base vectors for K-L transform are got from the training data set.

We plot below the spectral contrast of our sample:

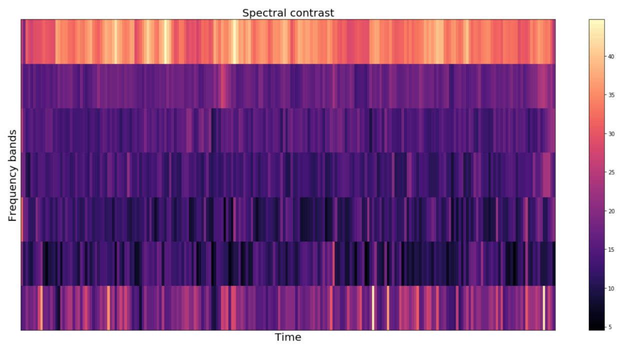


Figure 3: Spectral contrast

Each row of spectral contrast values corresponds to a given octave-based frequency.

4.1.8 Fast Fourier and Q-constant transforms. The frequencies that have been chosen to make up the scale of Western music are geometrically spaced. Thus the discrete Fourier transform (DFT), although extremely efficient, yields components which do not map efficiently to musical frequencies. The Q-constant transform solves thus using series of logarithmically spaced filters f_k, with the k-th filter having a spectral width δf_k equal to a multiple of the previous filter's width, namely:

$$\delta f_k = 2^{1/n} \cdot \delta f_{k-1} = \left(2^{1/n}\right)^k \cdot \delta f_{\min}$$

The Q-constant transform strength is its inherent dimensionality reduction: fewer frequency bins are required to cover a given range effectively, especially when frequencies span several octaves (human hearing covers approximately ten octaves from 20 Hz to around 20 kHz).

To represent the difference between the two, we compute from their resulting frequencies the amplitude spectrogram and convert it to a dB-scaled spectrogram and plots their heatmaps:

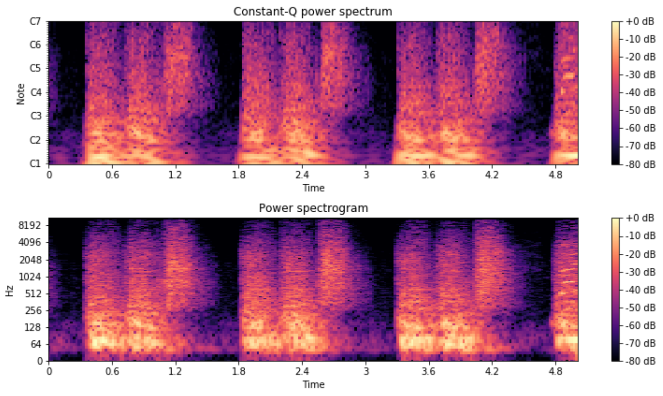


Figure 4: Spectrogram computed with SFFT and CQT

As we can see in the figure above, the constant-Q transform captures more information in the higher frequencies compared to the FFT.

According to [Judith C. Brown], note identification, instrument recognition, and signal separation are more easily tackled by pattern recognition algorithms if Q-constant transform. However, her experience was mainly restrained to instruments used in classical music. We will therefore use both transforms in our analysis. This distinction will prove useful in the remainder of the feature engineering section.

4.1.9 Chroma. Chroma features indicate how much energy of each pitch class ($C, C^*, D, D^*, E, ..., B$) is present in the signal. Specifically, they project the spectrum onto 12 bins representing the 12 distinct semitones (or chroma) of the musical octave. It is typically computed for several frames, thus yielding an matrix (easily convertible into an image using a heatmap).

The standard variants is to use either the FFT[6] of constant-Q transform[2] spectrum to compute the normalized energy for each chroma bin at each frame (see below). We normalize so that all bins amount to 1.

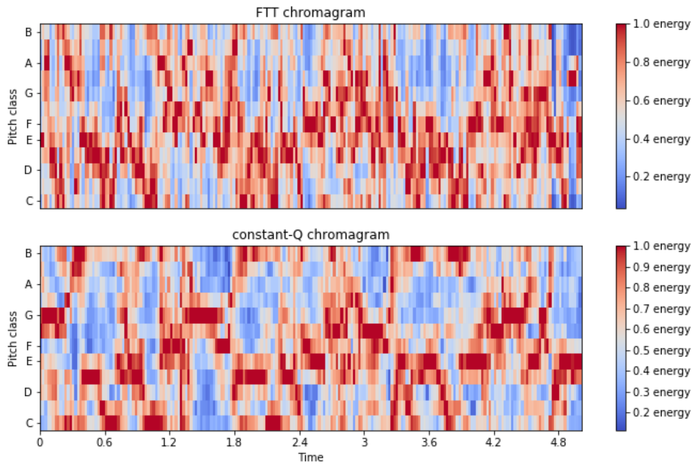


Figure 5: Chromagram based on FFT and constant-Q transform

The chroma works by measuring the distribution of the notes energy over time. It is therefore a key mid-level representation to capture melodic and harmonic characteristics, as well as note material. Furthermore, it is robust to changes in timbre and instrumentation.

Another variant called chroma CENS[17] consists of taking statistics over large windows to smooths local deviations in tempo, articulation, and musical ornaments such as trills and arpeggiated chords (see below). Those statistics are taken from the constant-Q chromagram (see Fig. 4).

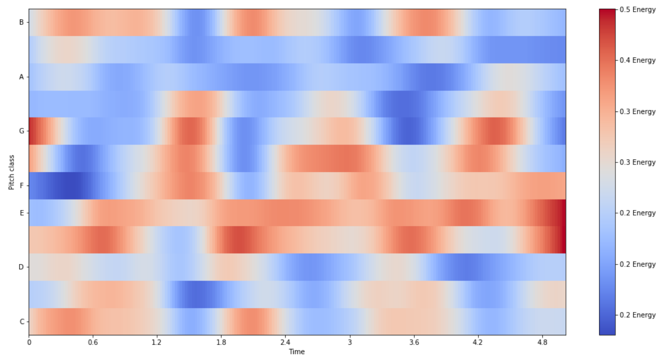


Figure 6: Chromagram Energy Normalized Statistics

The main advantage of CENS are its flexible granularity (window length) and its predictive powers for audio matching and similarity.

4.1.10 Mel-Frequency Cepstral Coefficients. Like other cepstrum techniques, MFC's aim is to transform a signal from the temporal domain to another one (frequency, time...). Specifically, Mel-spectrogram is a 2D representation that is optimized for human auditory perception. It compresses the STFT in frequency axis and therefore can be more efficient in its size while preserving the most perceptually important information. Mel-spectrogram only provides the magnitude (or energy) of the time-frequency bins, which means it is not invertible to audio signals.

The exhaustive method to compute the MFCC is the following:

(1) Apply a pre-emphasis filter to amplify the high frequencies
(2) Slice into (overlapping) frames
(3) Apply hamming window function each frame (counteract the assumption made by the FFT that the data is infinite and to reduce spectral leakage)
(4) Compute the Short-Time Fourier transform on each frame.
(5) Compute power spectrum.
(6) Compute the filter banks. Mel-scale aims to mimic the non-linear human ear perception of sound, by being more discriminative at lower frequencies and less discriminative at higher frequencies. The resulting features are however highly correlated.

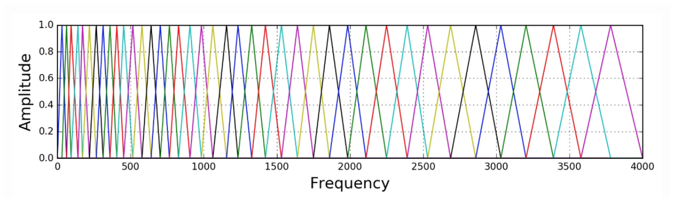


Figure 7: Mel-scale

(7) 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.

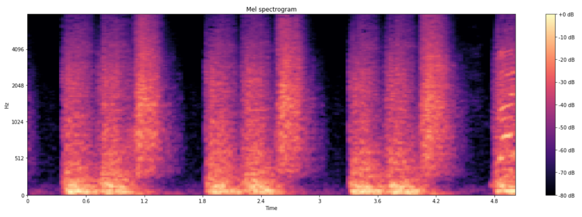


Figure 8: Mel-spectrogram

4.1.11 Tonnetz. The Tonnetz (shown in figure 2) is a well known planar representation of pitch relations first attributed to Euler and re-appropriated by Neo-Riemanninan Music Theorists. We choose to include the tonnetz in our analysis as various visual representations of the Tonnetz can be used to show traditional harmonic relationships in European classical music.

In the planar representation, close harmonic relations are modelled by small distances on the plane. Lines of fifths travel from left to right, lines of major thirds travel from bottom left to top right and lines of minor thirds travel from top left to bottom right.

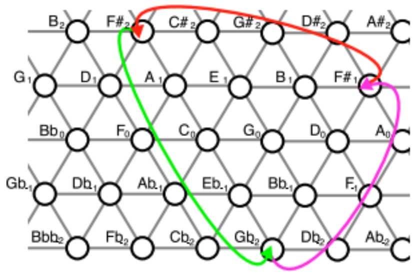


Figure 9: The Harmonic Network or Tonnetz. Arrows show the three circularities inherent in the network if octave equivalence are assumed.

If we assume that octave equivalence (see fig.2), the plane wraps up and forms a tube (Chew's Spiral Array) with the line of fifths becoming a helix on its surface. This helix is arranged so that major third intervals are directly above each other on the surface of the tube. If we further assume enharmonic equivalence, we can reduce the infinite number of pitch names to just 12 pitch classes. Topologically, it means we can join the end of the tubes and mapping it to an hypertorus, with the circle of fifths wrapping around its surface three times.

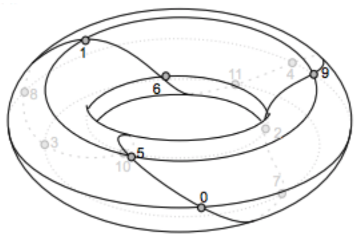


Figure 10: A projection showing how the Tonnetz wraps around the surface of a Hypertorus with the pitch classes following the spiral of fifths when enharmonic and octave equivalence are assumed.

Now that we projected our data into a suitable space to study harmonic relations, we can now compute the tonnal centroids, i.e. the centroid of the chroma vectors in this new space.
The tonal centroid ζ_n of the chroma for a frame n is computed in the the following fashion. We first compute Constant-Q chromagram.

Furthermore, we project each chroma pitch l onto dimension d: $\phi(d,l)$ and weights the results with chroma values c_l. Finally, we normalize the results by the norm of the chroma vector $|| \ c \ ||$. Mathematically, it translates to:

$$\zeta_n(d) = \frac{1}{|| \ c \ ||} \sum_{l=0}^{11} \phi(d,l) \cdot c_l$$

We plot below the tonnal centroids against the chroma below:

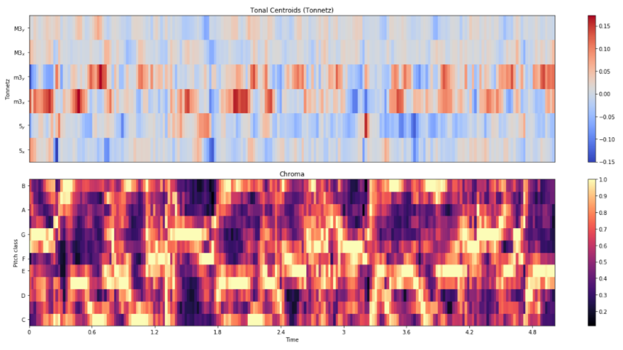


Figure 11: Tonnal centroids and chroma vectors plotted over time

The advantages of this approach is two-fold:

- Event-driven feature analysis has been shown to give more accurate musical feature extraction than more traditional approaches based on frames of equal length
- Preprocessing stage for further harmonic recognition and classification algorithms, mainly in chord change detection.

4.1.12 Zero-crossing rate. is the rate of sign-changes along a signal, i.e., the rate at which the signal changes from positive to negative or back. Namely for a signal s

$$\text{ZCR} = \frac{1}{T-1} \sum_{t=1}^{T-1} 1_{\mathbb{R}}(s_t s_{t-1})$$

This feature has been used heavily in both speech recognition and music information retrieval, being a key feature to classify percussive sounds. It is typically computed over several frames, yielding a vector.

4.1.13 Metadata feature engineering.

5 METHODOLOGY

In this paper, we will explore music genre recognition as a supervised and unsupervised categorization. We also look at the Deep Learning techniques that can be used for this task. The code for the entire pipeline can be found here. The code for this work can be found github. [1]

We first split the total dataset (8000 songs) using a stratified sampling into a training dataset of 6480 songs (roughly 80%), validation dataset of 720 songs (roughly 10%) and test dataset of 800 songs (roughly 10%).

[1]http://github.com/

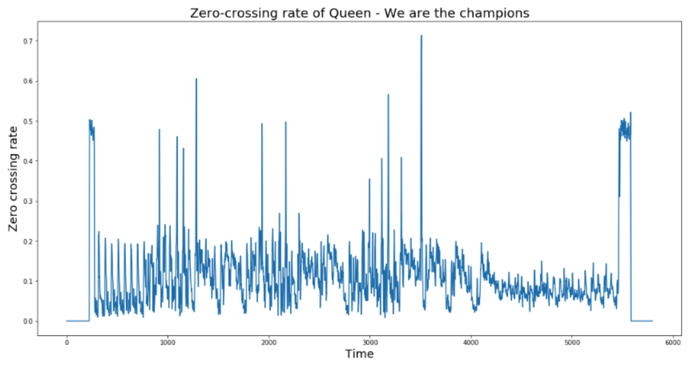


Figure 12: Zero-Crossing Rate of Queen

5.1 Supervised Machine Learning pipeline

Our machine learning pipeline contains following steps:

5.1.1 ***Pre-processing:*** For feature normalization we perform standard minmax normalization strategy in order ensure that all the features lie within the same range thereby making classifier less biased towards any particular feature.

5.1.2 ***Feature engineering:*** This part has been covered in detail in Section 3.

5.1.3 ***Supervised Methods:*** We applied a plethora of supervised machine learning techniques like Logistic Regression, Support Vector Classifier, Random Forest, Extra Trees, Gradient Boosting Classifier, Adaboost Classifier from [21]

5.1.4 ***Evaluation:*** In order to measure the performance of the machine learning classifiers, we consider two evaluation metrics i.e accuracy and the F1-Score (harmonic mean of precision and recall) for each classifier. The results of each classifier on the Validation dataset has been highlighted in Table1. We notice that the Extreme Gradient Boosting classifier from XgBoost outperforms the rest of the classifiers.

Machine_Learning_Methods	**Accuracy**	**F1 Score**
Logistic Regression Classifier	0.575	0.5673
Decision Tree Classifier	0.3652	0.3651
Random Forest Classifier	0.3680	0.3254
Multi-Class SVC	0.4555	0.4431
Multi-Class NuSVC	0.5638	0.5592
Multi-Class Linear SVC	0.5611	0.5459
Gradient Boosting Classifier (Scikit)	0.5638	0.5606
Extra Trees	0.5583	0.5498
Ext. Gradient Boosting Classifier (XgBoost)	**0.5833**	**0.5808**
Adaboost Classifier	0.4361	0.4229

Table 1: Results of Machine Learning Models on Val Dataset

Logistic Regression based classifier despite of its simplicity does quite well on our task and achieves a F1 score of 0.5673. The confusion matrix of the logistic regression classifier indicates that the algorithm has a competitive performance on Hip-Hop, Folk, Rock, International and Instrumental genres while performance drops on Pop genres. We also make an interesting observation that a lot of Pop songs are misclassified as Folk songs by this classifier.

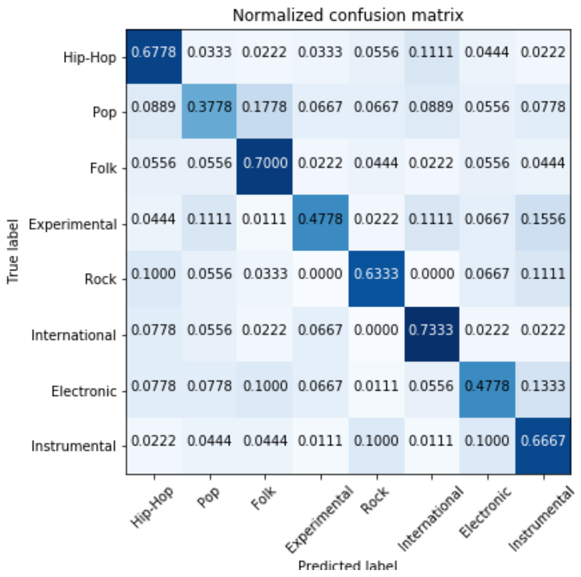


Figure 13: Confusion Matrix for the Logistic Regression Classifier

Examination of the confusion matrix of Support Vector Classifier (Linear Kernel) in Fig 12 brings us to the conclusion that this classifier follows the same pattern as followed by the Logistic Regression but with one exception on Experimental genre, where Linear Kernel Based Support Vector Classifier performs better. It is important to specify that the strategy used by this classifier to perform multiclass classification is One v Rest.

When we consider RBF Kernel based Nu Support Vector Classifier (variant of SVC), it achieves a F1 Score of 0.5638. The performance on Hip-Hop, Experimental and Electronic genres suffers a relative drop from the previous two classifiers but there is a significant improvement in the classification of the Pop genre. However again we notice that many songs whose true label is Pop are misclassified as Folk. It is important to specify that this classifier uses one v one classification strategy to perform multiclass classification.

Finally on scrutinizing the confusion matrix for extreme gradient boosting classifier (xgboost) (refer Fig 11), we can make certain interesting observations regarding the classifier. The classifier performs relatively good on genres like International, Instrumental, Folk and HipHop while the classifier achieves very poor performance on songs with Pop genres. In this case also we observe that

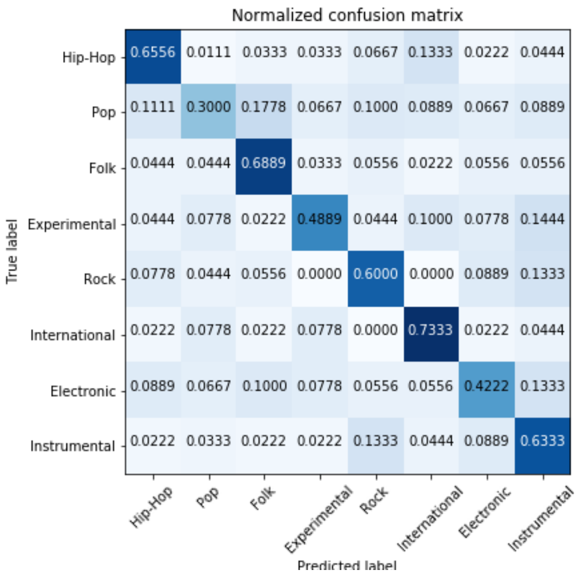


Figure 14: Confusion Matrix for the Multiclass Linear SVC

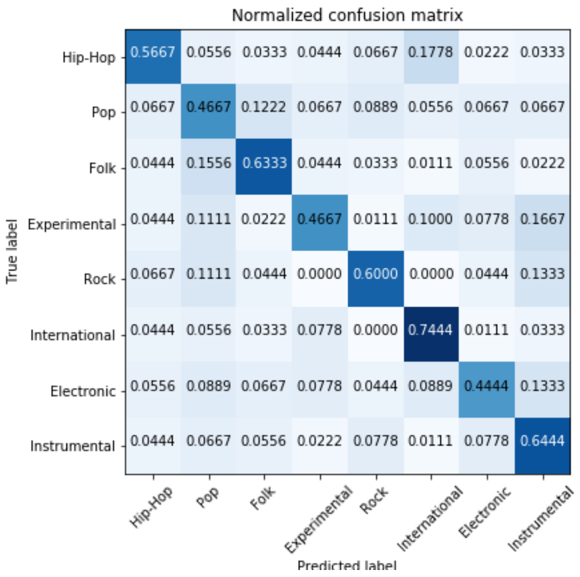


Figure 15: Confusion Matrix for the Multiclass RBF Kernel NuSVC

a large number of songs with Pop genre are misclassified as Folk, Experimental and Rock. Such observations motivate us to develop majority voting type of framework to improve this result. However we will address this framework in our future work on this problem.

In order to estimate best hyperparameters of our models, we used Stratified K-Fold Cross Validation (with K=5) on the training dataset and further use random search over the set of hyperparameters of Extreme Gradient Boosting Classifier from XgBoost Library. The advantage of using Stratified K-Fold Cross Validation

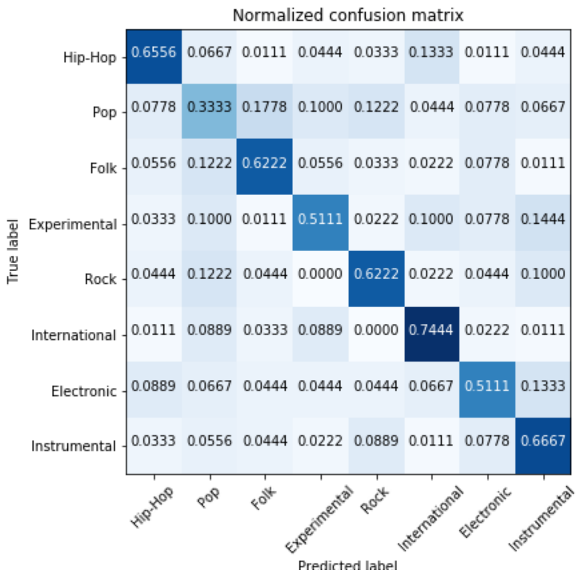


Figure 16: Confusion Matrix for the Gradient Boosting Classifier

Technique is that it preserves percentage of tracks per genre. We have used Random search strategy over Grid Search for tuning the parameters because the former turns out to work better in practice and is also much faster. The best parameters yielded by this strategy are gamma=2, max_depth=4, min_child_weight=1. **The accuracy and F1 Score achieved by the Tuned Classifier Model on Test Dataset are 0.525 and 0.5154 respectively.**

5.1.5 ***Limitations with the approach:*** Considering that the FMA dataset is quite a recent and majority of the work that has been done on it involves heavy use of deep learning and representation learning based approaches which is why we cannot directly compare our results to the ones achieved by the state of the art methods on this dataset. In keeping alignment with the content of the course, we have concentrated on more of a traditional machine learning based pipeline which involves amalgamation of domain specific feature engineering and application of general machine learning models on the problem. In our future work, we will aim to rectify this issue.

5.2 Unsupervised Techniques

Often applying unsupervised machine learning techniques gives a better insight about latent information in the dataset, which eventually motivated us to apply popular unsupervised learning based models to our problem. We are interested in discerning if unsupervised models can aid supervised to improve the performance.

5.2.1 ***Dimensionality Reduction****:* We perform dimensionality reduction on FMA Small Dataset using the technique of Principal Component Analysis (PCA), which is illustrated in Figure 15 and using tSNE, which is highlighted in Figure 16. While PCA finds the direction to project data, which maximizes the variance of the data whereas tSNE converts similarities between data points to joint probabilities and tries to minimize the Kullback-Leibler divergence between the joint probabilities of the low-dimensional embedding and the high-dimensional data.

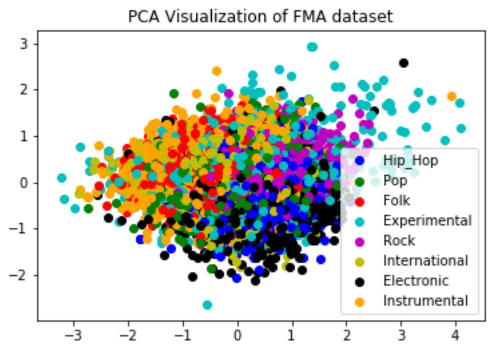


Figure 17: PCA Visualization of FMA Dataset

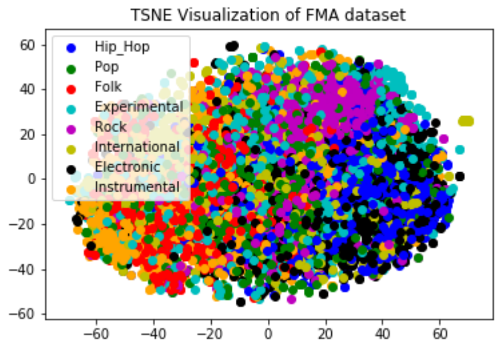


Figure 18: tSNE Visualization of FMA Dataset

5.2.2 ***K Means Clustering****:* Although PCA representation gives us a good intuition that this problem is not a good candidate for application of unsupervised machine learning methods directly but we are interested understanding if there is any hidden structure that KMeans clustering method can capture from the dataset and therefore we fix the value of the number of clusters to 8 while applying this method. Fig 17 represents 2D representation of various clusters created by the KMeans algorithm.

As a concluding remark on the unsupervised techniques we can comment that unsupervised machine learning techniques won't be very effective in increasing the efficiency and efficacy of our work because of the structure of the data points in the dataset.

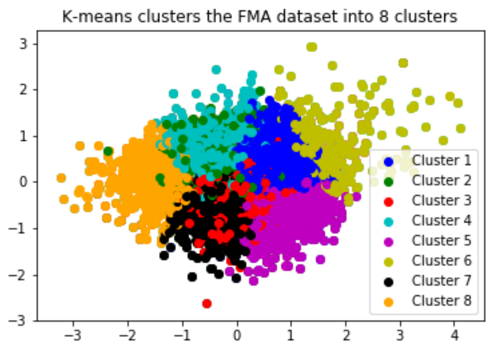


Figure 19: KMeans Clustering on FMA Dataset with 8 Clusters

5.3 Deep Learning:

In recent years, great results have been achieved in generating and processing images with neural networks. This can partly be attributed to the great performance of deep CNNs to capture and transform high-level information in images. A notable example of this is the process of image style transfer using CNNs proposed by L. Gatys et. al. which can render semantic content of an image in a different style [11].

Deep learning research is pretty much based on shared modules and methodologies such as dense layers, convolutional layers, recurrent layers, activation functions, loss functions, and backpropagation-based training. This makes the knowledge on deep learning generalizable for problems in different domains, e.g., convolutional neural networks were originally used for computer vision, but are now used in natural language processing and Music Information Retrieval.

5.3.1 *Why Deep Learning:*

It appears that we dispose of a relatively narrow set of features and therefore there may be many possibly relevant features that we missed or did not know about. One of the main advantages of this technique is that the CNN, by transforming the image through filters and maxpooling operations, will be able to discriminate between parameters that may not have a tangible sense (after all the transformations) but are highly relevant. However, the input remains decisive. The audio signal has not been the most popular choice; researchers have preferred 2D representations such as STFT and mel-spectrograms because learning a network starting from the audio signal requires even a larger dataset. We decided to use mel-spectrograms as inputs for our neural networks:

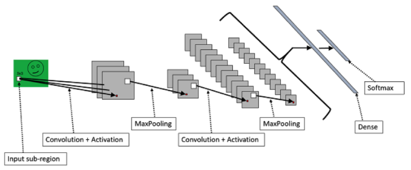
- **Short-Time Fourier Transform:** STFT provides a time-frequency representation with linearly-spaced centre frequencies. The computation of STFT is quicker than other time-frequency representations thanks to fast Fourier transform (FFT) which reduces the cost O(n*n) to O(nlog(n)) with respect to the number of FFT points. The linear centre frequencies are not always desired in music analysis. They do not match to the frequency resolution of human auditory system, nor musically motivated like the frequencies of Constant Q Transform (CQT). This is why STFT is not the most popular choice in deep learning âĂŞ it is not efficient in size as melspectrogram and not as raw as audio signals. One of the merits of STFT is that it is invertible to the audio signal, for which STFT was used in sonification of learned features [1].
- **Mel-spectrogram:** Mel-spectrogram is a 2D representation that is optimized for human auditory perception. It compresses the STFT in frequency axis and therefore can be more efficient in its size while preserving the most perceptually important information. Mel-spectrogram only provides the magnitude (or energy) of the time-frequency bins, which means it is not invertible to audio signals.

5.3.2 *Why not ANN:*.

By stacking dense layers on the top of a spectrogram, one can expect that the network will learn how to reshape the frequency responses into vectors in another space where the problem can be solved more easily (the representations becomes linearly separable). For example, if the task is pitch recognition, we can expect the first dense layer to be trained in such a way that each output node represents a different pitch. By its definition, a dense layer does not facilitate a shift or scale invariance. For example, if a STFT frame length of 257 is the input of a dense layer, the layer maps vectors from 257-dimensional space to another V -dimensional space. This means that even a tiny shift in frequency, which we might hope the network be invariant to for certain tasks, is considered to be a totally different representation.

5.3.3 *Framework:*

Here is an illustration of the CNN we used :



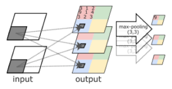


Figure 20: An illustration of a convolutional layer in details, where the numbers of channels of in- put/output are 2 and 3, respectively. The dotted arrows represent a convolution operation in the region, i.e., a dot product between convolutional kernel and local regions of input.

5.3.4 ***Kernel Size:*** The kernel size determines the maximum size of a component that the kernel can precisely capture in the layer. How small can a kernel be in solving MIR tasks? The layer would fail to learn a meaningful representation if the kernel is smaller than the target pattern. For example, for a chord recognition task, a kernel should be big enough to capture the difference between major and minor chords. For this reason, relatively large-sized kernels such as 17 ÃŮ 5 are used on 36-bins/octave CQT [1].

The second question would then be how big can a kernel be? One should note that a kernel does not allow an invariance within it. Therefore, if a large target pattern may slightly vary inside, it would better be captured with stacked convolutional layers with subsamplings so that small distortions can be allowed.

However, as large kernels imply more parameters, due to our limited computation capacity we used relatively small kernels (3*3).

5.3.5 ***Depth of the Network:*** In designing a network, one may find it arbitrary and empirical to decide the depth of the network. The network should be deep enough to approximate the relationship between the input and the output. If the relationship can be (roughly) formulated, one can start with a depth with which the network can implement the formula. Fortunately, it is becoming easier to train a very deep network. For example, networks for music tagging, boundary detection, and chord recognition in 2014 used 2- layer convnet [23] but recent research often uses 5 or more convolutional layers [9].

5.3.6 ***Generating the Spectrograms:*** The code used to generate the spectrograms used as inputs for our Deep Learning models can be found in the notebooks.

5.3.7 ***Organizing the Dataset***. In order to use a supervised CNN on images, one needs to have a very precise dataâĂŹs architecture. Indeed, it is mandatory to split the data into training, validation and test sets, but one also has to arrange all the images regarding their class. Here we used 8 different classes so in every folder (ie training, validation and test), there have to be 8 folders, each one containing the spectrograms of the corresponding class.

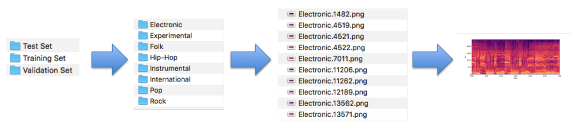


Figure 21: Illustration of dataset architecture for CNN

Moreover, the splits have to be randomly generated while the distribution between training, validation and test should be well balanced for each class.

A detailed code for dataset organization can be found in the CNN notebook.

6 EXPERIMENTS AND RESULTS

6.1 Naive CNN

As said above, the first CNN we tried consisted of 2 convolutional layers, each followed by a maxpooling layer and finally one fully connected layer. We ran it on 300 epochs using Adam optimizer. Our training set consisted of 5597 tracks while the validation set contained 801. Despite the long training time it required, we deliberately chose a high number of epochs in order to track overfitting if it occurs. Given the small dataset we used, it is highly likely that overfitting will occur. After 300 epochs we obtained an accuracy of 1 for training set while we only obtained 0.4 for validation set. Therefore we can suspect some overfitting.

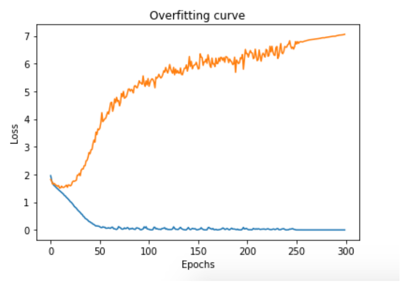


Figure 22: Plot of the loss depending on the number of epochs for naive CNN. Training set in blue and validation set in yellow

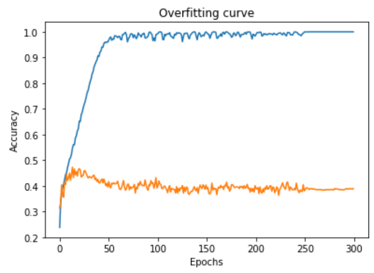


Figure 23: Plot of the accuracy depending on the number of epochs for naive CNN. Training set in blue and validation set in yellow

As we were expecting it appeared that our model heavily overfitted over the training set. The minimum loss and the highest accuracy was reached around 27 epochs. However, even at this moment we only obtained an accuracy of 0.46 over the validation set, which is still below the results of our best achieving Machine Learning models.

6.2 Batch Normalization

One of the easiest regularization techniques to avoid overfitting is to use the Dropout method. Dropout is a technique where randomly selected neurons are ignored during training. They are âĂIJdropped-outâĂİ randomly. This means that their contribution to the activation of downstream neurons is temporally removed on the forward pass and any weight updates are not applied to the neuron on the backward pass. Therefore, if neurons are randomly dropped out of the network during training, that other neurons will have to step in and handle the representation required to make predictions for the missing neurons. This is believed to result in multiple independent internal representations being learned by the network. The effect is that the network becomes less sensitive to the specific weights of neurons. This in turn results in a network that is capable of better generalization and is less likely to overfit the training data.

Unfortunately, dropout is generally less effective at regularizing convolutional layers. Since convolutional layers have few parameters, they need less regularization to begin with. Furthermore, because of the spatial relationships encoded in feature maps, activations can become highly correlated. This renders dropout ineffective [7].

A workaround is Batch Normalization. For an artificial neural network, scientists use to normalize the input layer by adjusting and scaling the activations. For example, when we have features from 0 to 1 and some from 1 to 1000, we should normalize them to speed up learning. If the input layer is benefiting from it, why not do the same thing also for the values in the hidden layers, that are changing all the time, and get 10 times or more improvement in the training speed.Batch normalization reduces the amount by what the hidden unit values shift around (covariance shift) [24].

Therefore we performed a Batch normalization after each convolutional layer and performed only 50 epochs this time.

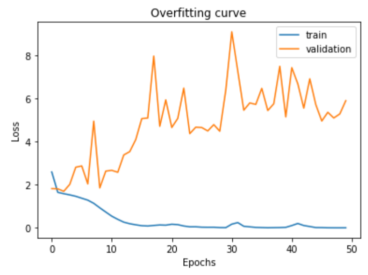


Figure 24: Plot of the accuracy depending on the number of epochs for normalized CNN.

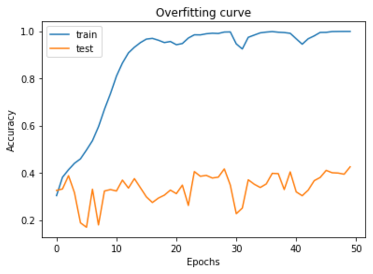


Figure 25: Plot of the accuracy depending on the number of epochs for normalized CNN.

This time we can observe that the model may overfit but later than the 50th epoch as the validation accuracy has an increasing trend is still not decreasing. However, the best accuracy has been reached at the 50th epoch with 0.43 for the validation set. It is lower than the best result achieved with naive CNN but it is likely to be due to the small number of epochs. With 300 epochs, the model may overfit again but could achieve way better results. Obviously, due to the huge training time it would require, we were not able to try running the model on 300 epochs.

6.3 What else can be done?

In order to avoid overfitting, one can:

- Use more data
- Use data augmentation
- Use architectures that generalize well
- Add regularization

In order to obtain better result, the easiest way is to get a bigger dataset. Unfortunately, we do not dispose of a bigger one at the moment. Therefore, one of the solutions remains data augmentation. However, the spectrograms we used are well standardized and adding noise or flipping the images will certainly cause a huge loss of information. Hence, data augmentation is not possible in this case. Moreover, our careful split of the dataset between Train, test and validation could not cause an unbalance resulting into overfitting. The two last points are already met by our normalized model.

Finally, in order to obtain better results we can also use cross validation along with parameter tuning using keras scikit wrapper. The model has been implemented on the notebook, but unfortunately, due to a lack of resources we were not able to run it. However, the full code can be found in the notebook.

6.4 Why don't CNNs on spectrograms perform as expected?

To explain those results we first should not forget that CNN are well suited for Image Analysis but we are not using them here for this task. Indeed, we are using a "machine vision" technique on a "machine hearing" problem. There are some remarkable differences between hearing and seeing hinting that the treatment for these two tasks should not be the same.

- **Sounds are transparent:** visual objects and sound events do not accumulate in the same manner. When encountering a pixel of a certain color in an image, it can most often be assumed to belong to a single object. Discrete sound events do not separate into layers on a spectrogram: Instead, they all sum together into a distinct whole. That means that a particular observed frequency in a spectrogram cannot be assumed to belong to a single sound as the magnitude of that frequency could have been produced by any number of accumulated sounds or even by the complex interactions between sound waves such as phase cancellation [22].
- **The axes of spectrograms do not carry the same meaning:** A CNN trains weights on 2D filters. This relies on the assumption that an object shown in image remains the same even if it is moved accross space. However, for a spectrogram, the x axis represents time while the y axis represent frequencies.Therefore, moving a point in a spectrogram vertically

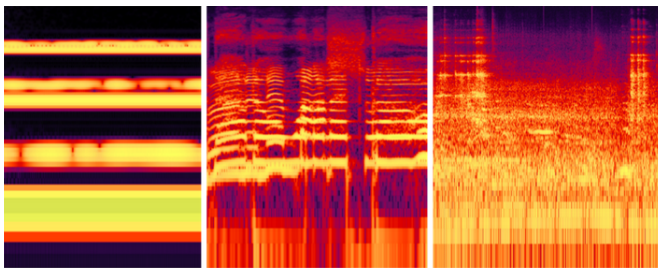


Figure 26: Three examples of difficult scenarios of spectrogram analysis. (Left): Two similar tones cause uneven phase cancellations across frequencies. (Middle): Two simultaneous voices with similar pitch are difficult to tell apart. (Right): Noisy and complex auditory scenes make it particularly difficult to distinguish sound events [22]

might affect its meaning, transforming a deep voice into a falsetto. Therefore, the spatial invariance that 2D CNNs provide might not perform as well for this form of data.

- **The spectral properties of sounds are non-local:** In images, similar neighboring pixels can often be assumed to belong to the same visual object. But for a sound, the timbre of a sound depends on the fundamental frequency and the harmonics around it. That is why in a guitar for example, there are countless different possibilities to play the same note. The frequencies of the fundamental note followed by its harmonics are not locally grouped but they move together according to a common relationship. This further complicates the task of finding local features in spectrograms using 2D convolutions as they are often unevenly spaced apart even though they move according to the same factors.

Knowing these substantial differences between image and sound treatment, we understand better why our CNNs are underperforming. This opens a new study field for sound analysis through deep learning.

7 CONCLUSION AND FUTURE WORK

Mention the highlights of the work.

Future work involves majority voting, using deep learning on specific parts of spectrogram generated from audio. We also incorporate techniques like Bayesian Optimisations for Hyperparameter optimization.

REFERENCES

[1] [n. d.]. Image Style Transfer Using Convolutional Neural Networks. In *Acoustics, Speech and Signal Processing.*

[2] 2018. Python implementation of constant-Q chromagram. *Labrosa* (2018). https://librosa.github.io/librosa/generated/librosa.feature.chroma_cqt.html

[3] Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu. 2014. Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on audio, speech, and language processing* 22, 10 (2014), 1533–1545.

[4] Keunwoo Choi, György Fazekas, Kyunghyun Cho, and Mark Sandler. 2017. A tutorial on deep learning for music information retrieval. *arXiv preprint arXiv:1709.04396* (2017).

[5] Michaël Defferrard, Kirell Benzi, Pierre Vandergheynst, and Xavier Bresson. 2017. FMA: A Dataset for Music Analysis. In *18th International Society for Music Information Retrieval Conference.* https://arxiv.org/abs/1612.01840

[6] Daniel P.W. Ellis. 2007. Chroma feature analysis and synthesis. *Labrosa* (2007). https://labrosa.ee.columbia.edu/matlab/chroma-ansyn/

[7] Yarin Gal Zoubin Ghahramani. [n. d.]. Bayesian Convolutional Neural Networks With Bernoulli Approximate Variational inference. ICLR 2016.

[8] Dan-Ning Jiang, Lie Lu, Hong-Jiang Zhang, Jian-Hua Tao, and Lian-Hong Cai. 2002. Music type classification by spectral contrast feature. In *Proceedings. IEEE International Conference on Multimedia and Expo*, Vol. 1. 113–116 vol.1. https://doi.org/10.1109/ICME.2002.1035731

[9] G. Fazekas K. Choi and M. Sandler. [n. d.]. Automatic tagging using deep convolutional neural networks. The 17th International Society of Music Information Retrieval Conference, New York, USA. International Society of Music Information Retrieval, 2016.

[10] Igor Karpov and Devika Subramanian. 2002. Hidden Markov classification for musical genres. *Course Project* (2002).

[11] A. S. Ecker L. A. Gatys and M. Bethge. [n. d.]. From music audio to chord tablature: Teaching deep convolutional networks toplay guitar. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[12] Thibault Langlois and Gonçalo Marques. 2009. Automatic music genre classification using a hierarchical clustering and a language model approach. In *Advances in Multimedia, 2009. MMEDIA'09. First International Conference on.* IEEE, 188–193.

[13] C. Laurier and Perfecto Herrera. 2007. Audio music mood classification using support vector machine. In *International Society for Music Information Research Conference (ISMIR).* files/publications/b6c067-ISMIR-MIREX-2007-Laurier-Herrera.pdf

[14] Chang-Hsing Lee, Jau-Ling Shih, Kun-Ming Yu, and Hwai-San Lin. 2009. Automatic music genre classification based on modulation spectral analysis of spectral and cepstral features. *IEEE Transactions on Multimedia* 11, 4 (2009), 670–682.

[15] Tao Li, Mitsunori Ogihara, and Qi Li. 2003. A comparative study on content-based music genre classification. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval.* ACM, 282–289.

[16] Thomas Lidy and Andreas Rauber. 2005. Evaluation of feature extractors and psycho-acoustic transformations for music genre classification. In *ISMIR.* 34–41.

[17] Meinard Müller, Frank Kurth, and Michael Clausen. 2005. Audio Matching via Chroma-Based Statistical Features. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* 288–295.

[18] Benjamin Murauer and Günther Specht. 2018. Detecting Music Genre Using Extreme Gradient Boosting. In *Companion of the The Web Conference 2018 on The Web Conference 2018.* International World Wide Web Conferences Steering Committee, 1923–1927.

[19] Loris Nanni, Yandre MG Costa, Alessandra Lumini, Moo Young Kim, and Seung Ryul Baek. 2016. Combining visual and acoustic features for music genre classification. *Expert Systems with Applications* 45 (2016), 108–117.

[20] Elias Pampalk, Arthur Flexer, Gerhard Widmer, et al. 2005. Improvements of Audio-Based Music Similarity and Genre Classificaton.. In *ISMIR*, Vol. 5. London, UK, 634–637.

[21] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[22] Daniel Rothmann. [n. d.]. WhatâĂŹs wrong with CNNs and spectrograms for audio processing?. In *Available online: https://towardsdatascience.com/whats-wrong-with-spectrograms-and-cnns-for-audio-processing-311377d7ccd?fbclid=IwAR24YuULVExdzc*$_X - O1YwGygqzut0yOwoJbbJz1Kq5a9VCd5AaYfIiCkm44$.

[23] J. Schluter and S. Bock. [n. d.]. Improved musical onset detection with convolutional neural networks. Acoustics, Speech and Signal Processing, IEEE International Conference on. IEEE, 2014.

[24] Sergey Ioffe Christian Szegedy. [n. d.]. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.

[25] G. Tzanetakis and P. Cook. 2002. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing* 10, 5 (July 2002), 293–302. https://doi.org/10.1109/TSA.2002.800560

[26] Lonce Wyse. 2017. Audio spectrogram representations for processing with convolutional neural networks. *arXiv preprint arXiv:1706.09559* (2017).