Music Genre Classification

A

Project Report

Submitted for the partial fulfillment

of B.Tech Degree

in

COMPUTER SCIENCE & ENGINEERING

by

Abhishek Kumar Singh	**1805210001**
Akash Saha	**1805210005**
Shivam Yadav	**1805210049**

Under the supervision of

Dr. Upendra Kumar

Mr. Ajeet Shukla

Department of Computer Science and Engineering

Institute of Engineering and Technology

Dr. A.P.J. Abdul Kalam Technical University, Lucknow, Uttar Pradesh July 2022

Contents

Contents	**1**
Declaration	**2**
Certificate	**3**
Acknowledgment	**4**
Abstract	**5**
List of Figures	**6**
Chapter 1	**7**
1.1 Introduction	7
1.2 Motivation	8
1.3 Project Objective	9
1.4 Report Layout	9
Chapter 2	**10**
2.1 Literature Review	10
Chapter 3	**16**
3.1 Methodology	16
3.1.1 Dataset	16
3.1.2 CNN Architecture	18
3.1.3 Flow	18
3.1.4 Creating Spectrograms	18
3.1.5 Creating Wavelets	19
3.1.6 Preprocessing and Feature Extraction	19
3.1.7 Scaling the Features	20
3.1.8 Model Training	20
4.1 Experimental Results	24
Chapter 5	**29**
5.1 Conclusions	29
5.2 Future Works	31
References	**32**

Declaration

We hereby declare that this submission is our own work and that, to the best of our belief and knowledge, it contains no material previously published or written by another person or material which to a substantial error has been accepted for the award of any degree or diploma of university or another institute of higher learning, except where the acknowledgment has been made in the text. The project has not been submitted by us at any other institute for the requirement of any other degree.

Submitted by: - Date: **26 May 2022**

(1) Name: Abhishek Kumar Singh

Roll No.: 1805210001

Branch: Computer Science and Engineering

Signature: abhishek

(2) Name: Akash Saha

Roll No.: 1805210005

Branch: Computer Science and Engineering

Signature: *akash saha*

(2) Name: Shivam Yadav

Roll No.: 1805210049

Branch: Computer Science and Engineering

Signature: Shivam

Certificate

This is to certify that the project report entitled "Music Genre Classification" presented by Abhishek Kumar Singh, Akash Saha, and Shivam Yadav in the partial fulfillment for the award of Bachelor of Technology in Computer Science and Engineering, is a record of work carried out by them under my supervision and guidance at the Department of Computer Science and Engineering at Institute of Engineering and Technology, Lucknow.

It is also certified that this project has not been submitted to any other institute for the award of any other degrees to the best of my knowledge.

Dr. Upendra Kumar

Department of Computer Science and Engineering

Institute of Engineering and Technology, Lucknow

Mr. Ajeet Shukla

Department of Computer Science and Engineering

Institute of Engineering and Technology, Lucknow

Acknowledgment

We would like to express our gratitude to Dr. Divakar Singh Yadav, Head, CSE Department, IET Lucknow, and our humble supervisors *Dr. Upendra Kumar* and *Mr. Ajeet Shukla* for their continued support and suggestions so far in this project. They have been very constructive, supportive, and kind and made it easier to write up this thesis work. Their continuous support gave us the motivation to work on this write-up and helped me to gain knowledge in various fields. I would also like to thank them for suggesting changes. They allowed me to locate improvement areas that helped me write this Interim write-up. We would like to express gratitude to all our classmates and teammates who motivated us and helped us in our time of need to complete this project. We shall be failing in duty if we do not acknowledge with thanks the authors of the references and other literature referred to in this thesis work.

Abhishek Kumar Singh(1805210001)
Signature: abhishek

Akash Saha 1805210005
Signature: *akash saha*

Shivam Yadav 1805210049
Signature: Shivam

Abstract

Music type categorization is a vital venture that includes the classification of music types from visual and audio entertainment transmitted via radio waves dossier. In the field of music computer data storage and retrieval, pleasant sounds, and harmonized type classification are repeatedly handled. The proposed foundation handles three main steps: dossier preprocessing, feature extraction, and categorization. A convolutional interconnected system (CNN) is the arrangement used to tackle music type categorization. The projected system uses feature principles of spectrograms produced from slices of air as the input into a CNN to categorize the verses into their pleasant sounds, and harmonized genres. An advice scheme is again implemented later in the categorization process. The approval system aims to advise canticles on each user's advantages and interests. Extensive experiments carried out on the GTZAN dataset show the influence of the projected system concerning different arrangements.

List of Figures

Figure 1.1: Music genre classification…………………………………………………7

Figure 1.2:GTZAN Dataset………………………………………………………...10

Figure 3.1.2 Flow……….…………………………………………………………..14

Figure 3.1.3 Spectrogram generation...…………………………………………………15

Figure 3.1.5 Preprocessing and Feature Extraction……...………………………………16

Figure 3.1.7(a) Model Training…………………………………………………….17

Figure 3.1.7(b) Model Training…………………………………………………….18

Figure 3.1.7(c) Model Training Code snippet……………………………………...18

Figure 3.1.7(d) Model Training Code snippet……………………………………...18

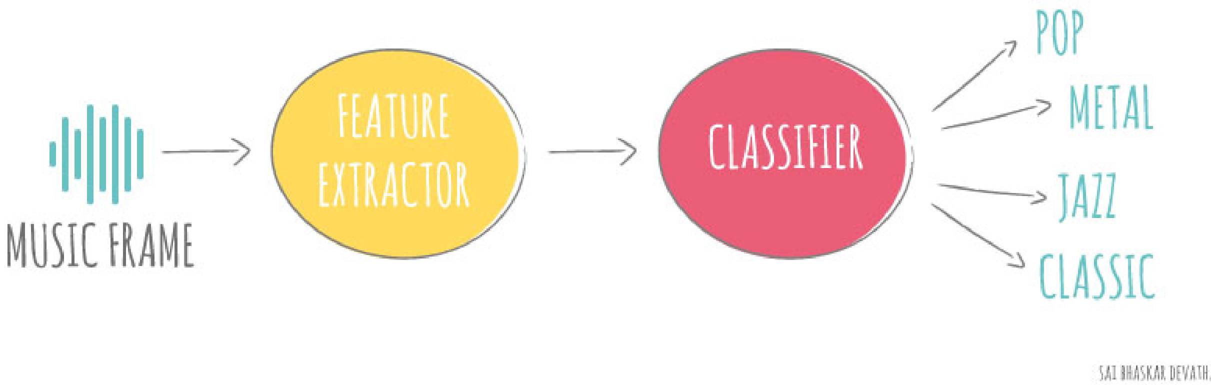
Figure 3.1.7(e) Model Training Code snippet……………………………………...18

Figure 4.1.1(a) Experimental Results………………………………………………21

Figure 4.1.1(b) Experimental Results………………………………………………22

Chapter 1

1.1 Introduction



Music type classification is a lively venture that contains the categorization of pleasant sounds, harmonized types from ocular and visual and audio entertainment transmitted via radio waves entertainment sent by way of wireless waves file. In the field of pleasant sounds, harmonized calculating data conversion and recovery sounds that are friendly, and harmonized type categorization are regularly controlled. The projected foundation handles three main steps: file preprocessing, feature ancestry, and classification. A convolutional pertain plan (CNN) is the composition used to tackle pleasant sounds and harmonized type classification. The thrown whole uses the feature law of spectrograms presented from slices of air as the recommendation into a CNN to categorize the verses into their friendly sounds, and harmonized types. A recommendation blueprint is repeated and achieved later in the classification process. The authorization scheme aims to warn canticles on each user's benefits and interests. Extensive experiments completed the activity on the GTZAN dataset show the influence of the discharged system having to do with various plans. Furthermore, the Convolutional Neural Network (CNN), one of the most important deep learning systems, has been used to categorise and recommend music genres.

Genres in all activities are subjective. Definitions can change over time, and the more explicit you try to realize, the more people will fiercely communicate to you you're wrong. But all the categories of music types have a few unique qualities that make them easier to recognize and categorize. For the most part, types are defined using the emotions that they induce and universal compositional elements. Horror flicks scare you by accompanying frightening monsters with a disturbing plot.

The categorising of music types is a crucial step in creating a comprehensive recommendation system. The goal of this project is to figure out how to work with sound files in Python, compute different attributes and audible acoustic wave facial features from them, run Machine Learning Algorithms on them, and show the results. The main goal is to build a machine intelligence model that categorises pleasant sounds and harmonised samples into different categories. Its goal is to predict the genre by recommending an audible acoustic wave signal.

The goal of automating pleasant sounds and harmonised categorisation is to rapidly and easily generate a collection of tracks. If a person needs to manually categorise the songs, he or she must confess to all of the verses and thus choose the genre. This is not only time-consuming, but also inconvenient. Automating piece classification may undoubtedly aid in the discovery of useful dossiers such as styles, standard type, and musicians. Choosing a music category is the first step in this process. The goal is to: To divide visual and audio entertainment delivered by radio waves files (in ".wav") into ten lyrical categories: ‘blues, classical, disco, hip-hop, jazz, metal, pop, reggae, rock, and country’.

1.2 Motivation

In Music IR research, genre classification is a common question. To group feature headings produced from short-opportunity record parts into kinds, the majority of music type categorization approaches use pattern recognition algorithms. Support Vector Machines (SVMs), Nearest-Neighbor (NN) classifiers, Gaussian Mixture Models, Linear Discriminant Analysis (LDA), and other classifiers are commonly utilised. Several low sound datasets have been utilised in studies to match the claimed classification accuracies; The GTZAN dataset (Tzanetakis and Cook, 2002) is, for example, the final dataset used for sound type categorization. This encourages us to use harmonic and text data from two databases to complete this goal.

1.3 Project Objective

The project objectives are

1. To categorize audio files into ten different musical genres: ‘classical, country, hip-hop, jazz, metal, disco, pop, blues, reggae, and rock’.
2. Try to introduce a web interface to upload music files to find their genres using both front-end and back-end technologies.

1.4 Report Layout

❖ Literature Review describes all the previous works done in this field.
❖ The methodology describes the architecture of the project.
❖ Implementation Details describe the tools that are used and the process that needs to be followed in each mode.
❖ The final outputs of this project are shown in the results section.
❖ The project's restrictions and future scope are presented in the conclusion.

Chapter 2

2.1 Literature Review

‘Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software is available from tensorflow.org’

TensorFlow is both a description interface and an implementation for putting machine learning algorithms into action. With little or no modification, TensorFlow calculations can be conducted on a wide range of heterogeneous systems, from mobile devices such as phones and tablets to large-scale distributed systems with hundreds of computers and thousands of processing devices such as GPU cards.

‘Hareesh Bahuleyan. Music genre classification using machine learning techniques. CoRR, abs/1804.01149, 2018’

Hareesh compares two aspects of musical information retrieval in this work. The first is convolutional neural networks, which determine the genre based solely on the spectrography of the input audio file. The second technique made use of hand-crafted characteristics from the temporal and frequency domains. Hareesh evaluates the performance of four machine learning classifiers he trained. He found the elements that were most important in genre classification. Google published the dataset with the following counts and labels:

	Genre	Count
1	Pop Music	8100
2	Rock Music	7990
3	Hip Hop Music	6958
4	Techno	6885
5	Rhythm Blues	4247
6	Vocal	3363
7	Reggae Music	2997
	Total	**40540**

‘François Chollet et al. Keras. https://keras.io, 2015’

Keras is a human-centric API rather than a machine-centric one. By offering consistent and straightforward APIs, decreasing the amount of user activities necessary for typical use cases, and delivering clear and actionable error signals, Keras follows best practises for lowering cognitive burden. There's also a lot of documentation and development tutorials.

‘Mingwen Dong. The convolutional neural network achieves human-level accuracy in music genre classification. CoRR, abs/1802.09697, 2018’

In this paper, the authors provide a novel approach to genre classification that blends human perception research in music genre classification with neurophysiology of the auditory system. In this technique, a convolutional neural network (CNN) was taught to give auditory system categorization by being trained on a small section of the musical signal. Following that, the music is divided into many audio segments, and CNN makes predictions for each segment before combining the results to produce a forecast for the entire song. The approach attained 70% accuracy (human-level accuracy), compared to 61 percent accuracy achieved with existing methods. The trained CNN approximated the auditory system's spectrotemporal receptive field (STRF).

‘Brian McFee, Matt McVicar, Stefan Balke, Carl Thomé, Vincent Lostanlen, Colin Raffel, Dana

Lee, Oriol Nieto, Eric Battenberg, Dan Ellis, Ryuichi Yamamoto, Josh Moore, WZY, Rachel Bittner, Keunwoo Choi, Pius Friesch, Fabian-Robert Stöter, Matt Vollrath, Siddhartha Kumar, nehz, Simon Waloschek, Seth, Rimvydas Naktinis, Douglas Repetto, Curtis "Fjord" Hawthorne, CJ Carr, João Felipe Santos, JackieWu, Erik, and Adrian Holovaty. librosa/librosa: 0.6.2, August 2018'

Brian McFee; Alexandros Metsai; Matt McVicar; Stefan Balke; Carl Thomé; Colin Raffel; Frank Zalkow; Ayoub Malek; Dana; Kyungyun Lee; Oriol Nieto; Dan Ellis; Jack Mason; Eric Battenberg; Scott Seyfarth; Ryuichi Yamamoto; viktorandreevichmorozov; Keunwoo Choi; Josh Moore; Rachel Bittner; Shunsuke Hidaka; Ziyao Wei; nullmightybofo; Darío Hereñú; Fabian-Robert Stöter; Pius Friesch; Adam Weiss; Matt Vollrath; Taewoon Kim; Thassilo

'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'

The authors describe a powerful framework for music genre categorization that integrates the rich, psycho-physiologically grounded properties of slow temporal modulations in music recordings with the capabilities of sparse representation-based classifiers. Linear subspace dimensionality reduction methodologies have been shown to be essential in the circumstance at hand. The proposed approach achieves music genre classification accuracy of 91 percent and 93.56 percent on the GTZAN and ISMIR2004 Genre datasets, respectively.

'F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M'

Scikit-learn is a Python package that includes a wide range of cutting-edge machine learning algorithms for supervised and unsupervised medium-scale applications. By utilising a general-purpose high-level language, this package tries to make machine learning accessible to non-specialists. The emphasis is on usability, performance, documentation, and API consistency. It has few dependencies and is offered under a simplified BSD licence, making it suitable for usage in both academic and commercial environments. The source code, binaries, and documentation are available for download at http://scikit-learn.sourceforge.net.

‘Music Genre Classification and Recommendation by Using Machine Learning Techniques Innovations in Intelligent Systems and Applications Conference (ASYU), IEEE. the Posted: 2018’

One of the subjects of interest in digital music processing is music genre prediction. Acoustic elements of music were extracted using digital signal processing techniques in this work, and subsequently music genre categorization and music suggestions were created using machine learning methods. Furthermore, deep learning approaches such as convolutional neural networks were utilised for genre categorization and music recommendation, and the performance of the generated results was compared. The GTZAN database was employed in the study, and the SVM method yielded the best results.

‘Basili, R., Serafini, A., Stellato, A.: Classification of musical genre: a machine learning approach. In: ISMIR (2004)’

In this context, the concept of community serves as a self-organizing complex system that supports and prompts the creation and assessment of a genre. According to this viewpoint, the community's job is to construct an ontology of inner phenomena (properties and norms that define genre) and exterior differences (habits that embody distinguish behaviour and trends).

‘Keunwoo Choi, George Fazekas, Mark Sandler, Kyunghyun Cho. Convolutional Recurrent Neural Networks for Music Classification. ArXiv 1609.04243’

For music tagging, the authors of this paper present a convolutional recurrent neural network (CRNN). CNNs use convolutional neural networks (CNNs) to extract local features and recurrent neural networks (RNNs) to summarise the extracted information across time. We compare CRNN to three CNN architectures used for music tagging while regulating the amount of parameters in terms of performance and training time per sample.

’Lin Feng, Shenlan Liu, Music Genre Classification with Paralleling Recurrent Convolutional Neural Network, arXiv:1712.08370G. Tzanetakis and P. Cook. Musical genre classification of audio signals. IEEE Transactions on Speech and AudioProcessing, 10(5):293-302, 2002’

Deep learning has proven to be effective and efficient in music genre classification. However, the current accomplishments have various flaws that hinder the completion of this categorization assignment. In this paper, we propose a hybrid architecture made up of paralleling CNN and Bi-RNN blocks. They are primarily concerned with spatial feature extraction and temporal frame order extraction, respectively. The two outputs are then combined to form a single strong representation of musical signals, which is then input into the softmax algorithm for categorization.

'Jan W¨ulfing and Martin Riedmiller. Unsupervised learning of local features for music classification. In ISMIR, pages139-144, 2012'

Automatic classification of music songs into mood, artist, or genre categories is a well-studied issue in music information retrieval. These categorization problems are divided into two steps: feature selection/extraction and classification.

'Changsheng Xu, MC Maddage, Xi Shao, Fang Cao, and QiTian. Musical genre classification using support vector machines. In Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP'03). 2003 IEEE International Conference on, volume 5, pages V-429. IEEE, 2003'

This study proposes an effective and successful method for automated musical genre categorization. A set of attributes is obtained and used to characterise music content. To categorise musical genres, a multi-layer classifier based on support vector machines is utilised. By learning from training data, support vector machines are used to find the best class borders across different musical genres.

'Spectral Features and Multiple Prototype VectorsRepresentation by Chang-Hsing Lee, Chih-Hsun Chou, Cheng-Chang Lien, and Jen-Cheng Fang Department of Computer Science and Information Engineering Chung HuaUniversity, Hsinchu, Taiwan'

The authors present a technique for automatically identifying music genres based on long-term modulation spectrum analysis of spectral (OSC and MPEG-7 NASE) and cepstral (MFCC) features in this paper. The collection of MFCC/OSC/NASE modulation spectra will be converted into a modulation spectrogram. The modulation spectrum is then partitioned into several logarithmically separated modulation subbands.

'N. Scaringella, G. Zoia, and D. Mlynek. Automatic genre classification of music content: a survey. IEEE SignalProcess. Mag., 23(2):133 – 141, 2006'

In this essay, we demonstrate how muddled musical genre designations are, despite their historical and cultural significance. We examined popular feature extraction methodologies used in music information retrieval for various music elements (see Table I); based on these features, we offered the three primary paradigms for audio genre identification, along with their benefits and drawbacks. The most recent findings from the MIREX 2005 music genre categorization contest are presented and debated. Finally, we discussed folksonomies and perceptual categories as new developing study disciplines and tools for investigating the closeness of musical genres.

‘Derek A. Huang, Arianna A. Serafini, Eli J. Pugh. Music Genre Classification’

We created a variety of categorization algorithms that work with two different sorts of data. The researchers tested an RBF kernel support vector machine, k-nearest neighbours, a simple feed-forward network, and an advanced convolutional neural network. We used both raw amplitude data and modified mel-spectrograms of the raw amplitude data to evaluate our algorithms. After that, a list of ten typical music genres is used to generate the anticipated genre. The performance of all of our models was improved by converting raw audio into mel-spectrograms, with our convolutional neural network outperforming human accuracy.

Chapter 3

3.1 Methodology

3.1.1 Dataset

For this undertaking, the dataset that we will be using for testing and training is the GTZAN Genre Classification dataset which comprises 1,000 soundtracks, each sound 30 seconds long in duration. It holds 10 categories, each represented by 100 tracks.

The 10 genres are as follows:

- Blues
- Disco
- Classical
- Country
- Rock
- Hip-hop
- Pop
- Jazz
- Metal
- Reggae

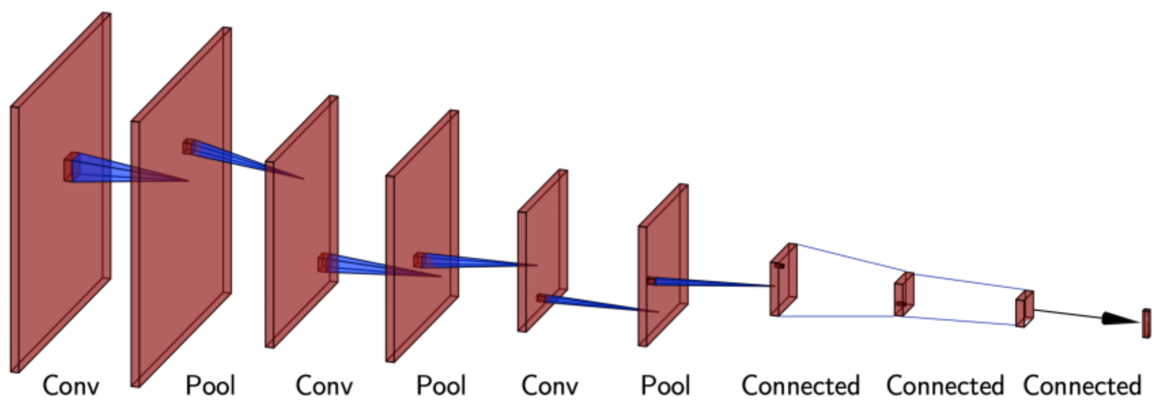
The dataset is divided into the following folders:

Original Genres — ‘A compilation of ten genres, each with 100 audio files lasting 30 seconds (the famous GTZAN dataset, the MNIST of sounds)’

Original images — A visual representation of each audio file. Because neural networks often take in some type of picture representation, they may be used to categorise data.

2 CSV files: include audio file characteristics. Each song (30 seconds long) in a file contains a mean and variance calculated using many functions that may be derived from an audio file. The other file has the same format as the first, except the songs are first separated into 3-second audio files.

3.1.2 CNN Architecture



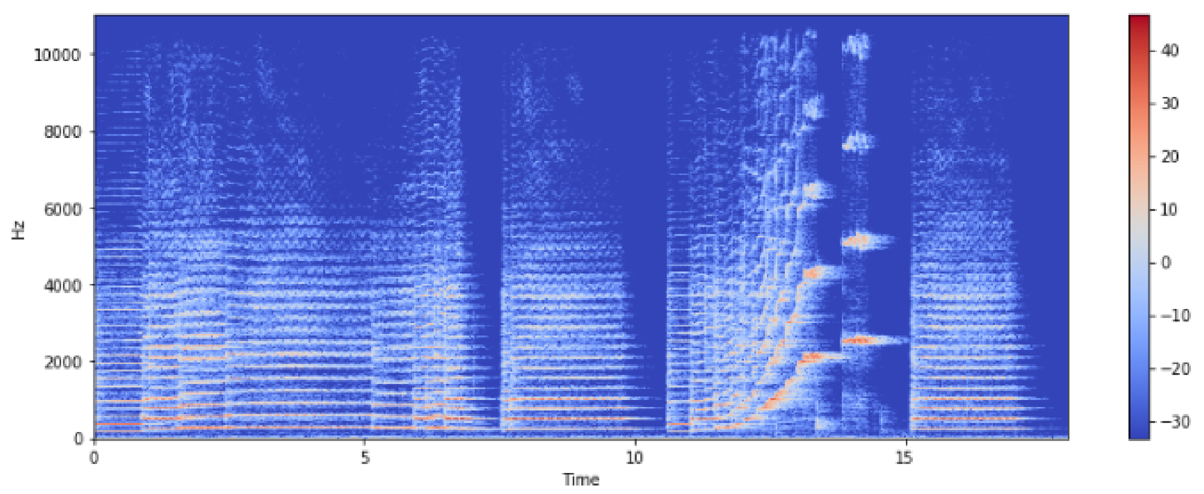
3.1.3 Flow

The figure below represents the general description of our methodology for the genus classification task. We will discuss each phase in detail. We train three types of deep learning models to explore the data and derive insights from it.

First, we need to convert the audio signals into a format compatible with the deep learning model. We use two types of formats, namely the following:

3.1.4 Creating Spectrograms

A spectrogram is a visual depiction of the signal frequencies in the spectrum as they change over time. We'll utilise the librosa package to convert any audio file into a spectrogram.



3.1.5 Creating Wavelets

The Wavelet Transform is a transformation for analysing the spectral and temporal features of non-stationary sources such as audio. We will use the librosa package to generate wavelets from each sound file.

3.1.6 Preprocessing and Feature Extraction

We produce training and test data after producing spectrograms and wavelets using standard picture pre-processing techniques. Before we can train the data, it must first be pre-processed. We'll try to concentrate on the last column, "label," and code it with the LabelEncoder() method from sklearn.preprocessing.

If we want to run a model on our data, we can't include text in it. So, before we can run a model, we must first prepare this data. The LabelEncoder class is used to convert category text data into numeric data that the model can understand.

3.1.7 Scaling the Features

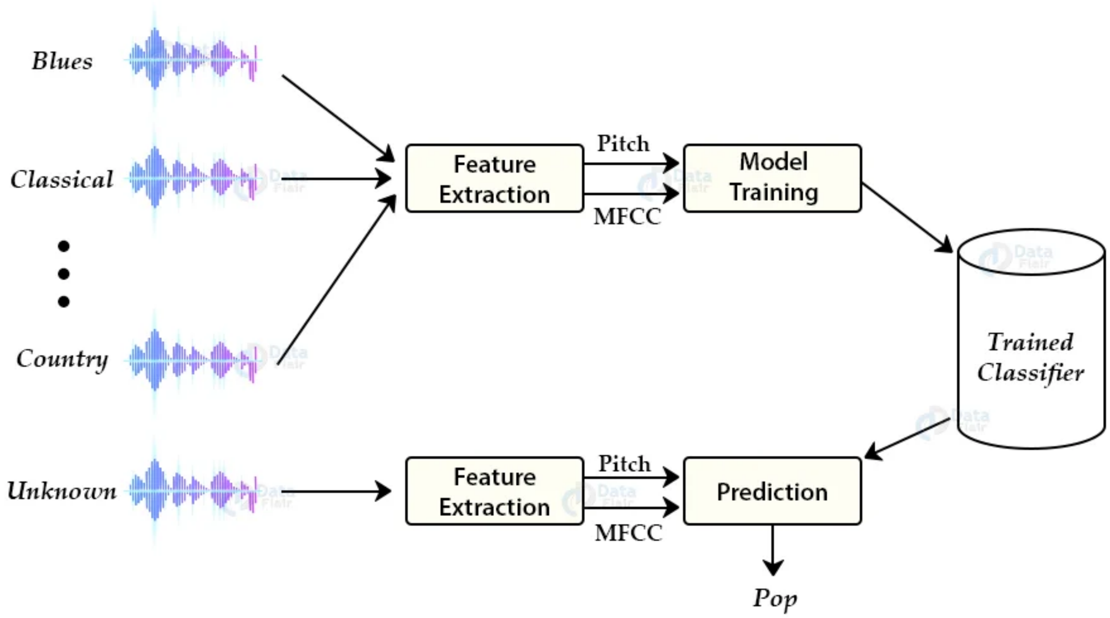
A standard scaler is used to normalise characteristics by decreasing the mean and scaling to unit variance.The standard score of sample x is calculated as z = (x - u) / s.

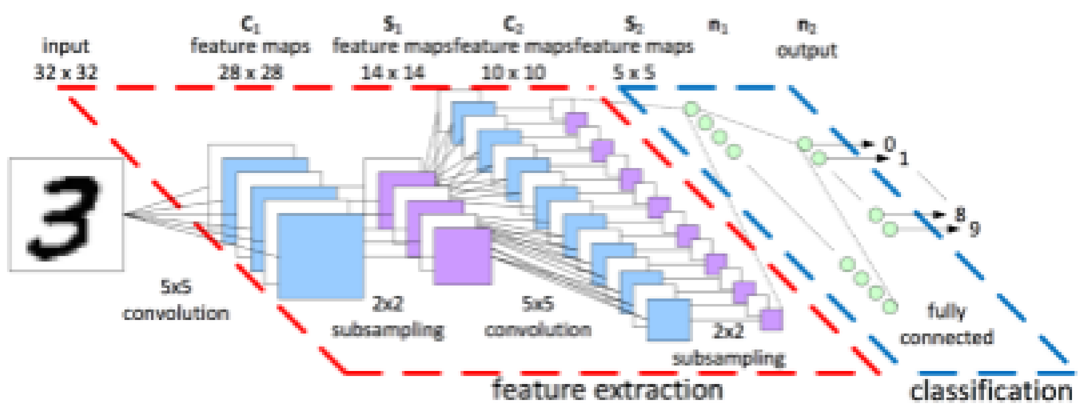
Many machine learning estimators need dataset standardisation: if individual features do not reflect standard normally distributed data, they may perform poorly.

3.1.8 Model Training

In this experiment, we will train the CNN model using both spectrogram and wavelet data. In this multi-modal training, we're applying the late-fusion strategy.

The Pytorch framework will be used to build our Convolutional Neural Network (CNN), which will have three layers of convolutions and a final fully connected layer with softmax activation and ten outputs (for 10 genres).


Blues
Classical
Country
Unknown
Feature Extraction
Pitch
MFCC
Model Training
Trained Classifier
Feature Extraction
Pitch
MFCC
Prediction
Pop


input
32 x 32
C_1
feature maps
28 x 28
S_1
feature maps
14 x 14
C_2
feature maps
10 x 10
S_2
feature maps
5 x 5
n_1
n_2
output
3
5x5
convolution
2x2
subsampling
5x5
convolution
2x2
subsampling
0
1
8
9
fully
connected
feature extraction
classification

```python
def __init__(self):
    super(genreNet, self).__init__()

    self.conv1  = Conv2d(in_channels=1,     out_channels=64,    kernel_size=3,  stride=1,   padding=1)
    torch.nn.init.xavier_uniform(self.conv1.weight)
    self.bn1    = BatchNorm2d(64)
    self.pool1  = MaxPool2d(kernel_size=2)

    self.conv2  = Conv2d(in_channels=64, out_channels=128,      kernel_size=3,  stride=1,   padding=1)
    torch.nn.init.xavier_uniform(self.conv2.weight)
    self.bn2    = BatchNorm2d(128)
    self.pool2  = MaxPool2d(kernel_size=2)

    self.conv3  = Conv2d(in_channels=128, out_channels=256,      kernel_size=3,  stride=1,   padding=1)
    torch.nn.init.xavier_uniform(self.conv3.weight)
    self.bn3    = BatchNorm2d(256)
    self.pool3  = MaxPool2d(kernel_size=4)

    self.conv4  = Conv2d(in_channels=256, out_channels=512,      kernel_size=3,  stride=1,   padding=1)
    torch.nn.init.xavier_uniform(self.conv4.weight)
    self.bn4    = BatchNorm2d(512)
    self.pool4  = MaxPool2d(kernel_size=4)

    self.fc1    = Linear(in_features=2048,  out_features=1024)
    self.drop1  = Dropout(0.5)

    self.fc2    = Linear(in_features=1024,  out_features=256)
    self.drop2  = Dropout(0.5)

    self.fc3    = Linear(in_features=256,   out_features=10)
```

```
for epoch in range(EPOCH_NUM):
    inp_train, out_train    = Variable(torch.from_numpy(x_train)).float() , Variable(torch.from_numpy(y_train)).long()
    inp_valid, out_valid    = Variable(torch.from_numpy(x_valid)).float(), Variable(torch.from_numpy(y_valid)).long()
    # ------------------------------------------------------------------------------------------------- #
    ## TRAIN PHASE # TRAIN PHASE # TRAIN PHASE # TRAIN PHASE # TRAIN PHASE # TRAIN PHASE # TRAIN PHASE  #
    # ------------------------------------------------------------------------------------------------- #
    train_loss = 0
    optimizer.zero_grad()  # <-- OPTIMIZER
    for i in range(0, TRAIN_SIZE, BATCH_SIZE):
        x_train_batch, y_train_batch = inp_train[i:i + BATCH_SIZE], out_train[i:i + BATCH_SIZE]

        pred_train_batch    = net(x_train_batch)
        loss_train_batch    = criterion(pred_train_batch, y_train_batch)
        train_loss          += loss_train_batch.data.cpu().numpy()

        loss_train_batch.backward()
    optimizer.step()  # <-- OPTIMIZER

    epoch_train_loss    = (train_loss * BATCH_SIZE) / TRAIN_SIZE
    train_sum           = 0
    for i in range(0, TRAIN_SIZE, BATCH_SIZE):
        pred_train      = net(inp_train[i:i + BATCH_SIZE])
        indices_train   = pred_train.max(1)[1]
        train_sum       += (indices_train == out_train[i:i + BATCH_SIZE]).sum().data.cpu().numpy()
    train_accuracy  = train_sum / float(TRAIN_SIZE)

    # ------------------------------------------------------------------------------------------------- #
```

```
    # ------------------------------------------------------------------------------------------------- #
    ## VALIDATION PHASE ## VALIDATION PHASE ## VALIDATION PHASE ## VALIDATION PHASE ## VALIDATION PHASE #
    # ------------------------------------------------------------------------------------------------- #
    valid_loss = 0
    for i in range(0, VALID_SIZE, BATCH_SIZE):
        x_valid_batch, y_valid_batch = inp_valid[i:i + BATCH_SIZE], out_valid[i:i + BATCH_SIZE]

        pred_valid_batch    = net(x_valid_batch)
        loss_valid_batch    = criterion(pred_valid_batch, y_valid_batch)
        valid_loss          += loss_valid_batch.data.cpu().numpy()

    epoch_valid_loss    = (valid_loss * BATCH_SIZE) / VALID_SIZE
    valid_sum           = 0
    for i in range(0, VALID_SIZE, BATCH_SIZE):
        pred_valid      = net(inp_valid[i:i + BATCH_SIZE])
        indices_valid   = pred_valid.max(1)[1]
        valid_sum       += (indices_valid == out_valid[i:i + BATCH_SIZE]).sum().data.cpu().numpy()
    valid_accuracy  = valid_sum / float(VALID_SIZE)

    print("Epoch: %d\t\tTrain loss : %.2f\t\tValid loss : %.2f\t\tTrain acc : %.2f\t\tValid acc : %.2f" % \
            (epoch + 1, epoch_train_loss, epoch_valid_loss, train_accuracy, valid_accuracy))
    # ------------------------------------------------------------------------------------------------- #
```

Chapter 4

4.1 Experimental Results

Because there are ten different genres in the GTZAN data set, accuracy was chosen as the major performance indicator. The average percentage of music similarity is used as a criterion for the quality of music suggestion, despite the fact that it is a subjective statistic from the listener's perspective. A confusion matrix is also used to calculate accuracy, recall, and F-measure scores.

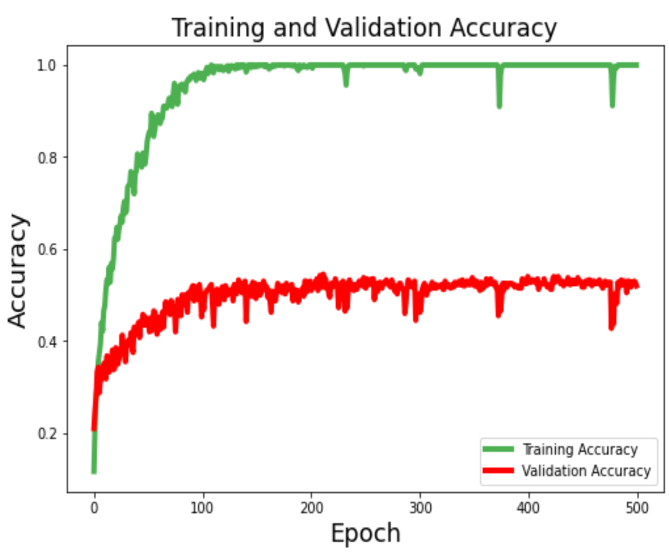
Although performance varied with each classification algorithm, similar trends were identified in the results of each. For example, Folk and Hip-Hop were classified the most accurate across most experiments. Folk may have been one of the most successfully classified genres because, as discussed earlier, its average scores for a number of the features were significantly different from those of other genres (refer to the MFCC1 diagram in the 'Feature and Genre Analysis' section of this report for an example). This implies the presence of sonic differences that were detected successfully in the training of the algorithms. Hip Hop's place as the most accurately classified genre is harder to explain; it may be that the presence of rapped vocal passages (i.e. vocal passages that are not sung) helped differentiate it if such passages led to distinctive feature trends. Further investigation of the dataset would be required to understand why Hip-Hop was classified the best. The accuracy kept on increasing with each epoch till a certain point and became almost constant after that, the max accuracy of **our model is 76% at epoch 210**.

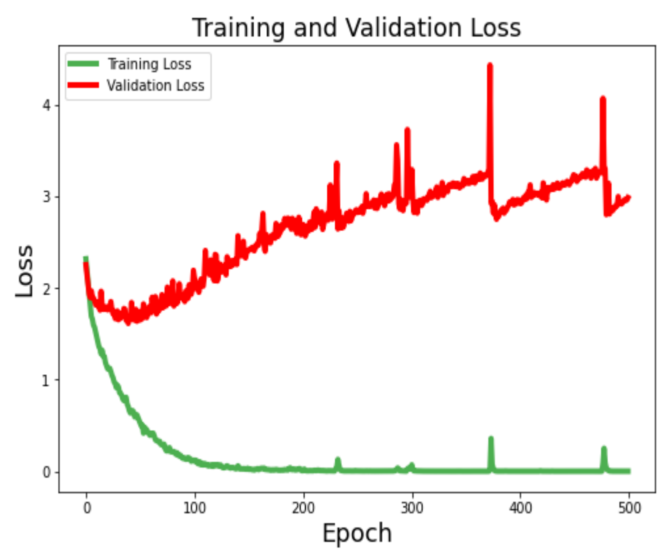
<u>Performance metrics of our CNN model</u>

Total Dataset	1000 mp3 files (10 genres * 100 files per genre)
Percentage of data used for training	70% (700)
Percentage of data used for validation	20% (200)
Percentage of data used for testing	10% (100)
Accuracy achieved	76% (at 210 epochs)

Why CNN is better than other methods of classification?

	With data processing			Without data processing		
	Train	CV	Test	Train	CV	Test
Support Vector Machine	.97	.60	.60	.75	.32	.28
K-Nearest Neighbors	1.00	.52	.54	1.00	.21	.21
Feed-forward Neural Network	.96	.55	.54	.64	.26	.25
Convolution Neural Network	.95	.84	**.82**	.85	.59	.53





Metrics related to epochs and individual inputs

```
❯ python3 get_genre.py ../test.mp3
     disco:      61.70   %
      rock:      38.30   %
```

```
Epoch: 44    Train loss : 1.63    Valid loss : 1.65    Train acc : 0.42    Valid acc : 0.41
Epoch: 45    Train loss : 1.62    Valid loss : 1.75    Train acc : 0.37    Valid acc : 0.36
Epoch: 46    Train loss : 1.69    Valid loss : 1.82    Train acc : 0.35    Valid acc : 0.35
Epoch: 47    Train loss : 1.81    Valid loss : 1.85    Train acc : 0.35    Valid acc : 0.34
Epoch: 48    Train loss : 1.83    Valid loss : 1.69    Train acc : 0.41    Valid acc : 0.39
Epoch: 49    Train loss : 1.65    Valid loss : 1.64    Train acc : 0.43    Valid acc : 0.42
Epoch: 50    Train loss : 1.61    Valid loss : 1.63    Train acc : 0.43    Valid acc : 0.42
Epoch: 51    Train loss : 1.58    Valid loss : 1.69    Train acc : 0.41    Valid acc : 0.41
Epoch: 52    Train loss : 1.63    Valid loss : 1.67    Train acc : 0.42    Valid acc : 0.40
Epoch: 53    Train loss : 1.63    Valid loss : 1.67    Train acc : 0.41    Valid acc : 0.39
Epoch: 54    Train loss : 1.61    Valid loss : 1.63    Train acc : 0.44    Valid acc : 0.42
Epoch: 55    Train loss : 1.58    Valid loss : 1.62    Train acc : 0.44    Valid acc : 0.42
Epoch: 56    Train loss : 1.56    Valid loss : 1.60    Train acc : 0.45    Valid acc : 0.44
Epoch: 57    Train loss : 1.55    Valid loss : 1.58    Train acc : 0.46    Valid acc : 0.44
Epoch: 58    Train loss : 1.52    Valid loss : 1.60    Train acc : 0.45    Valid acc : 0.44
Epoch: 59    Train loss : 1.54    Valid loss : 1.57    Train acc : 0.46    Valid acc : 0.43
Epoch: 60    Train loss : 1.52    Valid loss : 1.65    Train acc : 0.44    Valid acc : 0.41
Epoch: 61    Train loss : 1.60    Valid loss : 1.66    Train acc : 0.43    Valid acc : 0.41
Epoch: 62    Train loss : 1.59    Valid loss : 1.78    Train acc : 0.36    Valid acc : 0.35
Epoch: 63    Train loss : 1.75    Valid loss : 1.59    Train acc : 0.46    Valid acc : 0.45
Epoch: 64    Train loss : 1.54    Valid loss : 1.53    Train acc : 0.49    Valid acc : 0.46
Epoch: 65    Train loss : 1.48    Valid loss : 1.52    Train acc : 0.49    Valid acc : 0.48
Epoch: 66    Train loss : 1.45    Valid loss : 1.50    Train acc : 0.50    Valid acc : 0.48
Epoch: 67    Train loss : 1.43    Valid loss : 1.51    Train acc : 0.48    Valid acc : 0.46
Epoch: 68    Train loss : 1.47    Valid loss : 1.60    Train acc : 0.45    Valid acc : 0.43
Epoch: 69    Train loss : 1.53    Valid loss : 1.58    Train acc : 0.44    Valid acc : 0.45
Epoch: 70    Train loss : 1.54    Valid loss : 1.53    Train acc : 0.48    Valid acc : 0.47
Epoch: 71    Train loss : 1.47    Valid loss : 1.47    Train acc : 0.51    Valid acc : 0.50
Epoch: 72    Train loss : 1.40    Valid loss : 1.46    Train acc : 0.52    Valid acc : 0.49
Epoch: 73    Train loss : 1.39    Valid loss : 1.49    Train acc : 0.51    Valid acc : 0.49
Epoch: 74    Train loss : 1.42    Valid loss : 1.63    Train acc : 0.46    Valid acc : 0.44
Epoch: 75    Train loss : 1.55    Valid loss : 1.79    Train acc : 0.38    Valid acc : 0.39
Epoch: 76    Train loss : 1.77    Valid loss : 1.54    Train acc : 0.47    Valid acc : 0.45
Epoch: 77    Train loss : 1.49    Valid loss : 1.51    Train acc : 0.50    Valid acc : 0.48
Epoch: 78    Train loss : 1.45    Valid loss : 1.49    Train acc : 0.49    Valid acc : 0.47
Epoch: 79    Train loss : 1.42    Valid loss : 1.50    Train acc : 0.51    Valid acc : 0.49
Epoch: 80    Train loss : 1.41    Valid loss : 1.49    Train acc : 0.50    Valid acc : 0.48
Epoch: 81    Train loss : 1.41    Valid loss : 1.45    Train acc : 0.51    Valid acc : 0.51
Epoch: 82    Train loss : 1.37    Valid loss : 1.44    Train acc : 0.52    Valid acc : 0.50
Epoch: 83    Train loss : 1.36    Valid loss : 1.47    Train acc : 0.52    Valid acc : 0.51
```

```
poch: 1        Train loss : 2.35     Valid loss : 2.66     Train acc : 0.10     Valid acc : 0.11
poch: 2        Train loss : 2.67     Valid loss : 2.50     Train acc : 0.17     Valid acc : 0.18
poch: 3        Train loss : 2.52     Valid loss : 2.21     Train acc : 0.18     Valid acc : 0.19
poch: 4        Train loss : 2.22     Valid loss : 2.14     Train acc : 0.20     Valid acc : 0.20
poch: 5        Train loss : 2.15     Valid loss : 2.10     Train acc : 0.20     Valid acc : 0.22
poch: 6        Train loss : 2.11     Valid loss : 2.10     Train acc : 0.22     Valid acc : 0.23
poch: 7        Train loss : 2.11     Valid loss : 2.13     Train acc : 0.20     Valid acc : 0.20
poch: 8        Train loss : 2.14     Valid loss : 2.10     Train acc : 0.22     Valid acc : 0.24
poch: 9        Train loss : 2.11     Valid loss : 2.05     Train acc : 0.23     Valid acc : 0.24
poch: 10       Train loss : 2.05     Valid loss : 2.01     Train acc : 0.24     Valid acc : 0.25
poch: 11       Train loss : 2.02     Valid loss : 2.00     Train acc : 0.25     Valid acc : 0.25
poch: 12       Train loss : 2.00     Valid loss : 1.97     Train acc : 0.26     Valid acc : 0.27
poch: 13       Train loss : 1.98     Valid loss : 1.99     Train acc : 0.25     Valid acc : 0.25
poch: 14       Train loss : 1.99     Valid loss : 2.03     Train acc : 0.24     Valid acc : 0.24
poch: 15       Train loss : 2.04     Valid loss : 1.96     Train acc : 0.26     Valid acc : 0.26
poch: 16       Train loss : 1.97     Valid loss : 1.93     Train acc : 0.27     Valid acc : 0.28
poch: 17       Train loss : 1.94     Valid loss : 1.91     Train acc : 0.28     Valid acc : 0.28
poch: 18       Train loss : 1.92     Valid loss : 1.94     Train acc : 0.29     Valid acc : 0.29
poch: 19       Train loss : 1.93     Valid loss : 1.97     Train acc : 0.27     Valid acc : 0.27
poch: 20       Train loss : 1.96     Valid loss : 1.90     Train acc : 0.29     Valid acc : 0.29
poch: 21       Train loss : 1.91     Valid loss : 1.90     Train acc : 0.29     Valid acc : 0.29
poch: 22       Train loss : 1.90     Valid loss : 1.93     Train acc : 0.29     Valid acc : 0.29
poch: 23       Train loss : 1.92     Valid loss : 2.01     Train acc : 0.27     Valid acc : 0.27
poch: 24       Train loss : 2.01     Valid loss : 1.90     Train acc : 0.30     Valid acc : 0.31
poch: 25       Train loss : 1.89     Valid loss : 1.86     Train acc : 0.32     Valid acc : 0.31
poch: 26       Train loss : 1.85     Valid loss : 1.83     Train acc : 0.33     Valid acc : 0.32
poch: 27       Train loss : 1.82     Valid loss : 1.83     Train acc : 0.32     Valid acc : 0.32
poch: 28       Train loss : 1.82     Valid loss : 1.86     Train acc : 0.32     Valid acc : 0.33
poch: 29       Train loss : 1.84     Valid loss : 1.82     Train acc : 0.34     Valid acc : 0.31
poch: 30       Train loss : 1.81     Valid loss : 1.81     Train acc : 0.34     Valid acc : 0.34
poch: 31       Train loss : 1.80     Valid loss : 1.82     Train acc : 0.33     Valid acc : 0.33
poch: 32       Train loss : 1.81     Valid loss : 1.80     Train acc : 0.36     Valid acc : 0.35
poch: 33       Train loss : 1.77     Valid loss : 1.79     Train acc : 0.34     Valid acc : 0.34
poch: 34       Train loss : 1.77     Valid loss : 1.78     Train acc : 0.36     Valid acc : 0.35
poch: 35       Train loss : 1.76     Valid loss : 1.80     Train acc : 0.35     Valid acc : 0.35
poch: 36       Train loss : 1.77     Valid loss : 1.84     Train acc : 0.34     Valid acc : 0.33
poch: 37       Train loss : 1.84     Valid loss : 1.73     Train acc : 0.38     Valid acc : 0.38
poch: 38       Train loss : 1.72     Valid loss : 1.69     Train acc : 0.40     Valid acc : 0.39
poch: 39       Train loss : 1.67     Valid loss : 1.70     Train acc : 0.39     Valid acc : 0.37
poch: 40       Train loss : 1.67     Valid loss : 1.82     Train acc : 0.35     Valid acc : 0.36
poch: 41       Train loss : 1.81     Valid loss : 1.76     Train acc : 0.37     Valid acc : 0.36
poch: 42       Train loss : 1.72     Valid loss : 1.72     Train acc : 0.40     Valid acc : 0.39
poch: 43       Train loss : 1.70     Valid loss : 1.66     Train acc : 0.41     Valid acc : 0.40
```

```
Epoch: 125     Train loss : 1.14     Valid loss : 1.31     Train acc : 0.60     Valid acc : 0.55
Epoch: 126     Train loss : 1.17     Valid loss : 1.31     Train acc : 0.60     Valid acc : 0.55
Epoch: 127     Train loss : 1.16     Valid loss : 1.30     Train acc : 0.61     Valid acc : 0.55
Epoch: 128     Train loss : 1.14     Valid loss : 1.32     Train acc : 0.60     Valid acc : 0.55
Epoch: 129     Train loss : 1.17     Valid loss : 1.29     Train acc : 0.62     Valid acc : 0.58
Epoch: 130     Train loss : 1.11     Valid loss : 1.29     Train acc : 0.61     Valid acc : 0.57
Epoch: 131     Train loss : 1.16     Valid loss : 1.24     Train acc : 0.64     Valid acc : 0.58
Epoch: 132     Train loss : 1.09     Valid loss : 1.23     Train acc : 0.63     Valid acc : 0.59
Epoch: 133     Train loss : 1.08     Valid loss : 1.34     Train acc : 0.61     Valid acc : 0.55
Epoch: 134     Train loss : 1.14     Valid loss : 1.34     Train acc : 0.58     Valid acc : 0.55
Epoch: 135     Train loss : 1.18     Valid loss : 1.39     Train acc : 0.57     Valid acc : 0.52
Epoch: 136     Train loss : 1.24     Valid loss : 1.25     Train acc : 0.63     Valid acc : 0.57
Epoch: 137     Train loss : 1.10     Valid loss : 1.22     Train acc : 0.65     Valid acc : 0.60
Epoch: 138     Train loss : 1.07     Valid loss : 1.26     Train acc : 0.62     Valid acc : 0.58
Epoch: 139     Train loss : 1.09     Valid loss : 1.27     Train acc : 0.63     Valid acc : 0.57
Epoch: 140     Train loss : 1.10     Valid loss : 1.31     Train acc : 0.60     Valid acc : 0.55
Epoch: 141     Train loss : 1.16     Valid loss : 1.20     Train acc : 0.65     Valid acc : 0.59
Epoch: 142     Train loss : 1.04     Valid loss : 1.20     Train acc : 0.66     Valid acc : 0.61
Epoch: 143     Train loss : 1.04     Valid loss : 1.31     Train acc : 0.62     Valid acc : 0.57
Epoch: 144     Train loss : 1.13     Valid loss : 1.45     Train acc : 0.56     Valid acc : 0.53
Epoch: 145     Train loss : 1.33     Valid loss : 1.36     Train acc : 0.59     Valid acc : 0.56
Epoch: 146     Train loss : 1.22     Valid loss : 1.24     Train acc : 0.64     Valid acc : 0.60
Epoch: 147     Train loss : 1.07     Valid loss : 1.21     Train acc : 0.66     Valid acc : 0.60
Epoch: 148     Train loss : 1.05     Valid loss : 1.22     Train acc : 0.65     Valid acc : 0.60
Epoch: 149     Train loss : 1.02     Valid loss : 1.25     Train acc : 0.65     Valid acc : 0.59
Epoch: 150     Train loss : 1.06     Valid loss : 1.26     Train acc : 0.62     Valid acc : 0.57
Epoch: 151     Train loss : 1.08     Valid loss : 1.28     Train acc : 0.63     Valid acc : 0.57
Epoch: 152     Train loss : 1.09     Valid loss : 1.32     Train acc : 0.59     Valid acc : 0.56
Epoch: 153     Train loss : 1.16     Valid loss : 1.23     Train acc : 0.65     Valid acc : 0.59
Epoch: 154     Train loss : 1.05     Valid loss : 1.19     Train acc : 0.67     Valid acc : 0.60
Epoch: 155     Train loss : 0.99     Valid loss : 1.17     Train acc : 0.68     Valid acc : 0.61
Epoch: 156     Train loss : 0.96     Valid loss : 1.17     Train acc : 0.66     Valid acc : 0.61
Epoch: 157     Train loss : 0.98     Valid loss : 1.22     Train acc : 0.66     Valid acc : 0.59
Epoch: 158     Train loss : 1.02     Valid loss : 1.20     Train acc : 0.66     Valid acc : 0.59
Epoch: 159     Train loss : 1.01     Valid loss : 1.32     Train acc : 0.61     Valid acc : 0.57
Epoch: 160     Train loss : 1.14     Valid loss : 1.24     Train acc : 0.65     Valid acc : 0.58
Epoch: 161     Train loss : 1.05     Valid loss : 1.24     Train acc : 0.64     Valid acc : 0.60
Epoch: 162     Train loss : 1.05     Valid loss : 1.19     Train acc : 0.66     Valid acc : 0.60
Epoch: 163     Train loss : 0.97     Valid loss : 1.16     Train acc : 0.69     Valid acc : 0.63
Epoch: 164     Train loss : 0.95     Valid loss : 1.16     Train acc : 0.68     Valid acc : 0.62
Epoch: 165     Train loss : 0.95     Valid loss : 1.22     Train acc : 0.66     Valid acc : 0.59
Epoch: 166     Train loss : 1.00     Valid loss : 1.31     Train acc : 0.61     Valid acc : 0.55
Epoch: 167     Train loss : 1.11     Valid loss : 1.27     Train acc : 0.65     Valid acc : 0.58
Epoch: 168     Train loss : 1.06     Valid loss : 1.12     Train acc : 0.70     Valid acc : 0.63
```

```
Epoch: 178          Train loss : 0.92          Valid loss : 1.16          Train acc : 0.70          Valid acc : 0.63
Epoch: 179          Train loss : 0.92          Valid loss : 1.19          Train acc : 0.68          Valid acc : 0.60
Epoch: 180          Train loss : 0.96          Valid loss : 1.20          Train acc : 0.67          Valid acc : 0.62
Epoch: 181          Train loss : 0.97          Valid loss : 1.15          Train acc : 0.69          Valid acc : 0.61
Epoch: 182          Train loss : 0.93          Valid loss : 1.13          Train acc : 0.69          Valid acc : 0.63
Epoch: 183          Train loss : 0.92          Valid loss : 1.13          Train acc : 0.70          Valid acc : 0.60
Epoch: 184          Train loss : 0.91          Valid loss : 1.17          Train acc : 0.69          Valid acc : 0.63
Epoch: 185          Train loss : 0.93          Valid loss : 1.14          Train acc : 0.70          Valid acc : 0.63
Epoch: 186          Train loss : 0.90          Valid loss : 1.15          Train acc : 0.68          Valid acc : 0.63
Epoch: 187          Train loss : 0.95          Valid loss : 1.19          Train acc : 0.70          Valid acc : 0.62
Epoch: 188          Train loss : 0.92          Valid loss : 1.16          Train acc : 0.68          Valid acc : 0.61
Epoch: 189          Train loss : 0.95          Valid loss : 1.12          Train acc : 0.71          Valid acc : 0.63
Epoch: 190          Train loss : 0.88          Valid loss : 1.12          Train acc : 0.72          Valid acc : 0.62
Epoch: 191          Train loss : 0.86          Valid loss : 1.13          Train acc : 0.72          Valid acc : 0.63
Epoch: 192          Train loss : 0.87          Valid loss : 1.08          Train acc : 0.72          Valid acc : 0.64
Epoch: 193          Train loss : 0.85          Valid loss : 1.15          Train acc : 0.71          Valid acc : 0.61
Epoch: 194          Train loss : 0.90          Valid loss : 1.15          Train acc : 0.70          Valid acc : 0.63
Epoch: 195          Train loss : 0.91          Valid loss : 1.12          Train acc : 0.71          Valid acc : 0.64
Epoch: 196          Train loss : 0.87          Valid loss : 1.17          Train acc : 0.70          Valid acc : 0.62
Epoch: 197          Train loss : 0.90          Valid loss : 1.19          Train acc : 0.68          Valid acc : 0.60
Epoch: 198          Train loss : 0.94          Valid loss : 1.25          Train acc : 0.65          Valid acc : 0.58
Epoch: 199          Train loss : 1.01          Valid loss : 1.19          Train acc : 0.67          Valid acc : 0.60
Epoch: 200          Train loss : 0.98          Valid loss : 1.12          Train acc : 0.71          Valid acc : 0.63
Epoch: 201          Train loss : 0.87          Valid loss : 1.16          Train acc : 0.70          Valid acc : 0.62
Epoch: 202          Train loss : 0.91          Valid loss : 1.14          Train acc : 0.72          Valid acc : 0.62
Epoch: 203          Train loss : 0.86          Valid loss : 1.09          Train acc : 0.72          Valid acc : 0.65
Epoch: 204          Train loss : 0.84          Valid loss : 1.11          Train acc : 0.73          Valid acc : 0.64
Epoch: 205          Train loss : 0.83          Valid loss : 1.16          Train acc : 0.71          Valid acc : 0.63
Epoch: 206          Train loss : 0.88          Valid loss : 1.12          Train acc : 0.72          Valid acc : 0.63
Epoch: 207          Train loss : 0.85          Valid loss : 1.13          Train acc : 0.70          Valid acc : 0.64
Epoch: 208          Train loss : 0.89          Valid loss : 1.12          Train acc : 0.73          Valid acc : 0.63
Epoch: 209          Train loss : 0.84          Valid loss : 1.06          Train acc : 0.73          Valid acc : 0.65
Epoch: 210          Train loss : 0.79          Valid loss : 1.04          Train acc : 0.76          Valid acc : 0.65
Epoch: 211          Train loss : 0.77          Valid loss : 1.12          Train acc : 0.73          Valid acc : 0.64
Epoch: 212          Train loss : 0.82          Valid loss : 1.28          Train acc : 0.66          Valid acc : 0.59
Epoch: 213          Train loss : 1.02          Valid loss : 1.19          Train acc : 0.69          Valid acc : 0.62
Epoch: 214          Train loss : 0.93          Valid loss : 1.09          Train acc : 0.73          Valid acc : 0.64
Epoch: 215          Train loss : 0.83          Valid loss : 1.15          Train acc : 0.71          Valid acc : 0.62
Epoch: 216          Train loss : 0.87          Valid loss : 1.13          Train acc : 0.71          Valid acc : 0.63
Epoch: 217          Train loss : 0.86          Valid loss : 1.09          Train acc : 0.73          Valid acc : 0.64
Epoch: 218          Train loss : 0.83          Valid loss : 1.09          Train acc : 0.74          Valid acc : 0.62
Epoch: 219          Train loss : 0.82          Valid loss : 1.07          Train acc : 0.75          Valid acc : 0.64
Epoch: 220          Train loss : 0.78          Valid loss : 1.17          Train acc : 0.71          Valid acc : 0.61
```

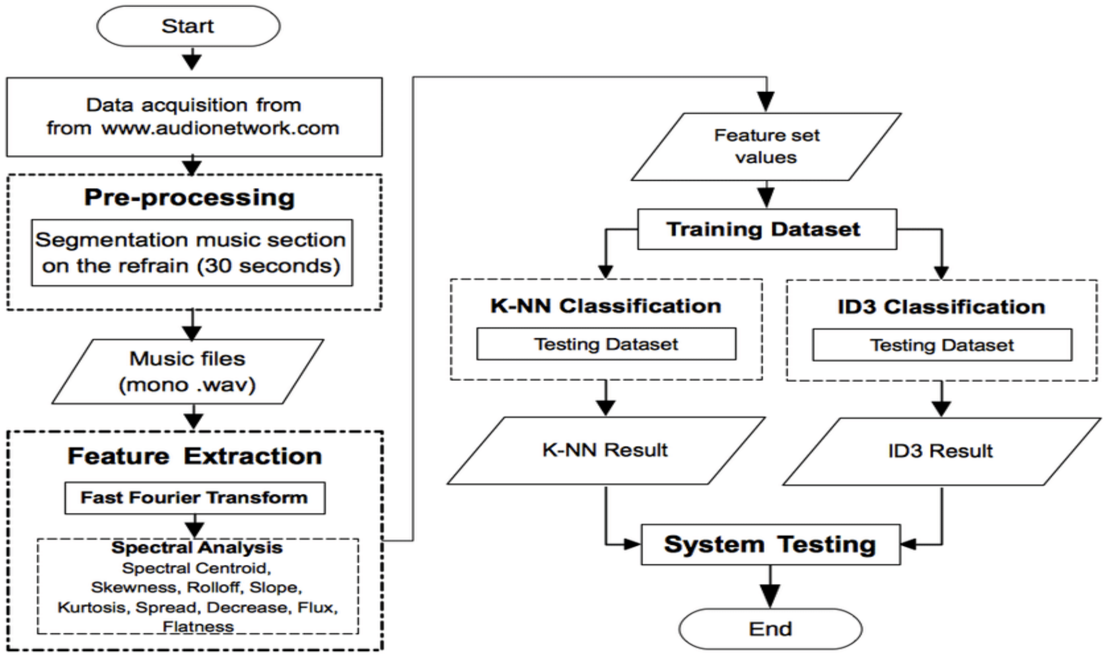
Chapter 5

5.1 Conclusions

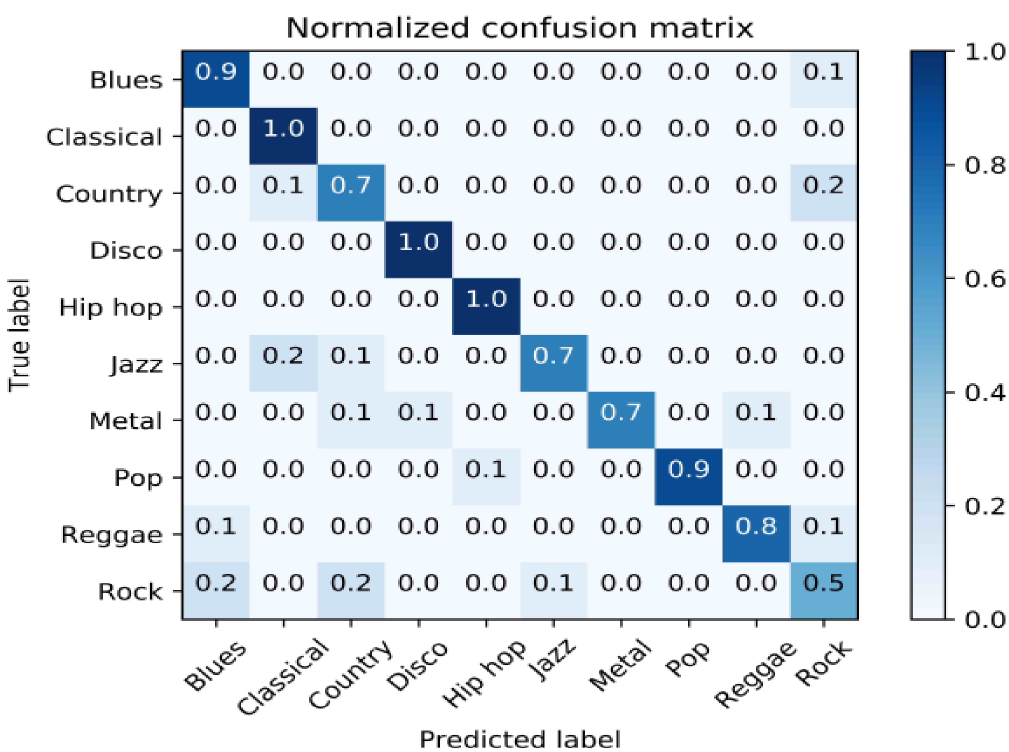
CNN performed the best, as predicted. It also takes the most time to train, but the higher accuracy more than compensates for the additional computational cost. The similarity in accuracy between the KNN, SVM, and feed-forward neural networks, on the other hand, astounded us. When performance data are reviewed, some comparable music genres, such as Jazz and Classic, may result in misclassification and misrecommendation. To enhance previous findings, we wish to construct more extended deep neural network models and use different data models as inputs in future studies, in addition to only employing the Mel spectrogram. In music genre recommendation systems, big data processing techniques and technologies can be used for feature extraction and model construction.

Several avenues for further investigation present themselves. First, as has been noted, the use of time-based features could allow for classification based on more complex musical properties, as could be performed by algorithms such as recurrent neural networks. Second, other types of features could be investigated in terms of their relevance for this type of classification problem. Only one explicitly rhythmic feature was extracted for this study - average tempo - thus greater emphasis on rhythm may represent an interesting opportunity for researchers in this field. Key signature and melody could also be worthy of greater focus. Finally, more work on curating high-quality datasets could be performed. This could involve working with and reducing the size of pre-existing datasets, combining existing datasets together, or collecting new tracks that are not currently in the public domain. Overall, music genre classification remains an interesting and worthwhile challenge for both academic institutions and businesses alike, and there is plenty of room for further study and analysis.

Flow diagram when using KNN for classification



Confusion matrix for CNN predictions



5.2 Future Works

We intend to experiment with further deep learning algorithms in the future, as they performed the best. Given the nature of the data, an RNN model may perform well (GRU, LSTM, for example). The project's generative elements, such as genre conversion, have piqued our interest (in the same vein as generative adversarial networks which repaint photos in the style of Van Gogh, but specifically for music). Furthermore, we foresee the possibility of transfer learning, such as categorising music by artist or decade. We want to create a web interface for uploading music files and detecting their genres using both front-end and back-end technologies.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software is available from tensorflow.org.
[2] Hareesh Bahuleyan. Music genre classification using machine learning techniques. CoRR, abs/1804.01149, 2018.
[3] François Chollet et al. Keras. https://keras.io, 2015.
[4] Mingwen Dong. The convolutional neural network achieves human-level accuracy in music genre classification. CoRR, abs/1802.09697, 2018.
5] Brian McFee, Matt McVicar, Stefan Balke, Carl Thomé, Vincent Lostanlen, Colin Raffel, Dana Lee, Oriol Nieto, Eric Battenberg, Dan Ellis, Ryuichi Yamamoto, Josh Moore, WZY, Rachel Bittner, Keunwoo Choi, Pius Friesch, Fabian-Robert Stöter, Matt Vollrath, Siddhartha Kumar, nehz, Simon Waloschek, Seth, Rimvydas Naktinis, Douglas Repetto, Curtis "Fjord" Hawthorne, CJ Carr, João Felipe Santos, JackieWu, Erik, and Adrian Holovaty. librosa/librosa: 0.6.2, August 2018. [
[6] 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.
[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M.
[8] Music Genre Classification and Recommendation by Using Machine Learning Techniques Innovations in Intelligent Systems and Applications Conference (ASYU), IEEE. the
Posted: 2018
[9] Basili, R., Serafini, A., Stellato, A.: Classification of musical genre: a machine learning approach. In: ISMIR (2004)
[10] Keunwoo Choi, George Fazekas, Mark Sandler, Kyunghyun Cho. Convolutional Recurrent Neural Networks for Music Classification. ArXiv 1609.04243
[11] Lin Feng, Shenlan Liu, Music Genre Classification with Paralleling Recurrent Convolutional Neural Network, arXiv:1712.08370G. Tzanetakis and P. Cook. Musical genre classification of audio signals. IEEE Transactions on Speech and AudioProcessing, 10(5):293-302, 2002.
[12] Jan W¨ulfing and Martin Riedmiller. Unsupervised learning of local features for music classification. In ISMIR, pages139-144, 2012.
[13] Changsheng Xu, MC Maddage, Xi Shao, Fang Cao, and QiTian. Musical genre classification using support vector machines. In Acoustics, Speech, and Signal Processing, 2003. Proceedings.

(ICASSP’03). 2003 IEEE International Conference on, volume 5, pages V-429. IEEE, 2003.
[14] Spectral Features and Multiple Prototype VectorsRepresentation by Chang-Hsing Lee, Chih-Hsun Chou, Cheng-Chang Lien, and Jen-Cheng Fang Department of Computer Science and Information Engineering Chung HuaUniversity, Hsinchu, Taiwan.
[15] N. Scaringella, G. Zoia, and D. Mlynek. Automatic genre classification of music content: a survey. IEEE SignalProcess. Mag., 23(2):133 – 141, 2006
[16] Derek A. Huang, Arianna A. Serafini, Eli J. Pugh. Music Genre Classification