**REPUBLIC OF TURKEY**
**YILDIZ TECHNICAL UNIVERSITY**
**DEPARTMENT OF COMPUTER ENGINEERING**


# MUSIC GENRE CLASSIFICATION WITH VISION TRANSFORMER AND CNN ARCHITECTURES USING DIGITAL SIGNAL PROCESSING TECHNIQUES

17011071 — Hatice DEMİR
17011057 — Tarık AYTEK

**SENIOR PROJECT**

Advisor
Dr. Ahmet ELBİR

Jan, 2024

# ACKNOWLEDGEMENTS

We would like to thank Dr. Ahmet ELBİR for his precious time and help.

Hatice DEMİR
Tarık AYTEK

# TABLE OF CONTENTS

# LIST OF ABBREVIATIONS

| | |
|---|---|
| AI | Artificial Intelligence |
| CNN | Convolutional Neural Network |
| ConvNet | Convolutional Neural Network |
| CV | Computer Vision |
| DSP | Digital Signal Processing |
| FMA | Free Music Archive |
| LSTM | Long-Short Time Memory |
| MFCC | Mel-frequency Cepstrum Coefficient |
| MGC | Music Genre Classification |
| MIR | Music Information Retrieval |
| NLP | Natural Language Processing |
| ReLU | Rectified Linear Activation |
| RGB | Red Green Blue |
| RNN | Recurrent Neural Network |
| RMSProp | Root Mean Squared Propagation |
| SGD | Stochastic Gradient Descent |
| STFT | Short-Time Fourier Transform |
| ViT | Vision Transformer |

# LIST OF FIGURES

# LIST OF TABLES

# ABSTRACT

## MUSIC GENRE CLASSIFICATION WITH VISION TRANSFORMER AND CNN ARCHITECTURES USING DIGITAL SIGNAL PROCESSING TECHNIQUES

Hatice DEMİR
Tarık AYTEK

Department of Computer Engineering
Senior Project

Advisor: Dr. Ahmet ELBİR

The aim of this project is to classify music data according to their genres using Convolutinal Neural Network (CNN) and Vision Transformer (ViT) architectures and to examine the effects of various parameters of the used architectures on the result. The project emerged with the motivation to evaluate the ViT architecture as an alternative to the CNN architectures frequently used in this field, to observe by making comparisons how solutions can be applied with various data sets, and to analyze the parameters of the models. The fact that music has a wide range of classifications, such as being classified according to its genres, according to the mood it creates, and according to its acoustic features, shows the diversity of topics that research can shed light on.

For this purpose, the music dataset was processed with digital signal processing (DSP) methods and converted into image data. After performing various pre-processing on the obtained image data, many CNN and ViT models were trained with different parameters and these models were subjected to comparative analysis. The advantages and disadvantages of the models and parameters compared to each other have been observed.

**Keywords:** Librosa, Music, Genre, CNN, ViT, Classification, DSP

# ÖZET

## SAYISAL İŞARET İŞLEME TEKNİKLERİ KULLANARAK VISION TRANSFORMER VE CNN MİMARİLERİYLE MÜZİK TÜRÜ SINIFLANDIRMA

Hatice DEMİR
Tarık AYTEK

Bilgisayar Mühendisliği Bölümü
Bitirme Projesi

Danışman: Öğr. Gör. Dr. Ahmet ELBİR

Bu projede amaç CNN ve ViT mimarilerini kullanarak müzik verilerini türlerine göre sınıflandırmak ve kullanılan mimarilerin çeşitli parametrelerinin sonuca etkilerini incelemektir. Proje, bu alanda sıkça kullanılan CNN mimarilerine bir alternatif olarak ViT mimarisini değerlendirmek, çeşitli veri setleriyle nasıl çözümler uygulanabileceğini karşılaştırmalar yaparak gözlemlemek, modellerin parametreleri alanında analiz yapmak motivasyonlarıyla ortaya çıkmıştır. Müziğin türlerine göre sınıflandırılmasının yanında kişide yarattığı duygulara göre, akustik özelliklerine göre sınıflandırılması gibi çok geniş bir yelpazeye sahip olması, araştırmanın ışık tutabileceği konuların çeşitliliğini göstermektedir.

Bu amaç doğrultusunda müzik veriseti dijital sinyal işleme metotları ile işlenmiş ve görüntü verilerine çevrilmiştir. Elde edilen görüntü verileri üzerinde çeşitli ön işlemler yapılmasının ardından değişik parametrelerle çok sayıda CNN ve ViT modelleri eğitilmiş, bu modeller karşılaştırmalı analize tabi tutulmuştur. Modellerin ve parametrelerin birbirlerine kıyasla avantaj ve dezavantajları gözlemlenmiştir.

**Anahtar Kelimeler:** Librosa, Müzik, Tür, CNN, ViT, Sınıflandırma, Dİİ

# 1
# Introduction

Music genre classification and the other similar Music Information Retrieval (MIR) tasks are becoming increasingly important these days as social media and streaming platforms have become a big part of our daily lives. These tasks traditionally were done with classical machine learning techniques. As Artificial Neural Networks continue to develop and gain popularity, research efforts are mostly focused on natural language processing (NLP) and computer vision (CV) tasks. Our research aims to find the potential common use of architectures developed for NLP and CV tasks on DSP or directly MIR tasks. The focus is evaluating the performance of ViT and CNN with different parameters on classification of music genres. For achieving this goal, GTZAN dataset will be used. It is a dataset that is divided into 10 different music genres and often used in the field. In the first step, music files will be processed through the Librosa library using DSP techniques, as is generally done in this field. After Fourier transformation and some other steps, 2D STFT and MEL spectrograms with time and frequency axes will be obtained as image files from the 1D signal data. In this way, spectrogram images with higher information will be used as the data set. Various ViT and CNN architectures will then be designed by determining the parameters that can be tested on the created data set. Comparison tables will be prepared based on determined parameters in order to obtain comments on various perspectives for both types of architecture. After the comparative analysis, observations on optimum approaches will be obtained by determining the specific success and failure points of the models and parameters. It will be aimed to shed light on the use of architectures evaluated in this way on similar topics and similar data sets.

# 2
# Literature Review

## 2.1 Studies Done

In the article titled "Vision Transformer for Music Genre Classification using Mel-frequency Cepstrum Coefficient," published in 2021, a similar study was done [1]. This study focused on the Mel-frequency Cepstrum Coefficient (MFCC) feature of music tracks, and the classification was performed based on this feature. ViT, CNN, and Long-Short Time Memory (LSTM) architectures were implemented with fixed parameters, aiming to show the better performance of ViT model. Analysis of different modeling approaches is done and results are presented in Table 2.1.

Another article published in 2023, titled "Music Audio Sentiment Classification Based on Improved Vision Transformer," implemented the ViT architecture for classification task [2]. In this study, tracks were initially classified into four categories based on their tone and then further classified into genres, aiming to achieve high accuracy rates. Classifying tracks into the types of sentiments which are quiet, passionate, sad, relaxed made surprisingly helpful impact on final score. They finally had around %90 accuracy rate.

There is another article published in 2020, titled "Music Genre Classification: A Comparative Study Between Deep-Learning And Traditional Machine Learning Approaches" [3]. The paper compares deep learning and traditional machine learning in music genre classification. It highlights the challenge of defining music genres due to their abstract nature and evolving styles. The research explores spectrograms and content-based features on the GTZAN dataset. Motivated by the need to manage vast

**Table 2.1** Comparison of Accuracy and Loss of Various Models

| Architecture | Test Accuracy | Test Loss |
|---|---|---|
| Convolutional Neural Network | 51.59% | 1.3884 |
| Vision Transformer | 56.85% | 1.2845 |
| RNN-LSTM | 57.13% | 1.2636 |

music databases in digital services, the study investigates using audio signal waves as spectrogram images for deep learning. It hypothesizes that CNNs using these images would outperform traditional methods. The study conducts music genre classification using both spectrograms and content-based features. It experiments with 30-second and 3-second feature sets, finding that shorter duration improve classification accuracy across models. Results emphasize the superiority of 3-second feature sets in achieving better classification accuracy. This research significantly contributes to advancing music genre classification methods.

There is another article published in 2020, titled "Deep attention based music genre classification" [4]. This study focuses on the application of CNN in music genre classification. The similarity between audio spectrograms and RGB images has encouraged the use of various deep learning models in music genre classification (MGC) tasks. However, assuming equal importance for spectra at different temporal steps contradicts the theory of processing bottleneck in psychology, as well as observations from audio spectrograms, leading some researchers to find it flawed. Therefore, considering the differences in spectra, a new model is proposed that incorporates an Attention Mechanism based on Bidirectional Recurrent Neural Networks. Additionally, two attention-based models, namely serial attention and parallelized attention, are suggested. Parallelized attention demonstrates greater flexibility and superior performance in experimental results over serial attention, notably CNN-based parallelized attention models taking STFT spectrograms as input, surpassing prior work. The study underscores the significance of music genre classification within music information retrieval. It explores the use of CNN-based models alongside traditional machine learning methods, highlighting the effective utilization of different spectrogram types (STFT, MFCC) in deep learning. Furthermore, it validates the performance enhancement in music classification through attention mechanisms (serial and parallel). By presenting the methodology, experimental results, and comparisons, the study elucidates the impact of attention mechanisms coupled with deep learning models on music genre classification.

There is another article published in 2017, titled "Transfer learning for music classification and regression tasks" [5]. In this paper, a transfer learning approach is introduced for tasks involving music classification and regression. The proposal involves utilizing a pre-trained convolutional neural network (convnet) feature, which comprises a concatenated feature vector generated from the activations of feature maps across multiple layers in a trained convolutional network. This convnet feature is suggested as a versatile representation for music in general. The experiments conducted train a convnet specifically for music tagging and subsequently apply it to other music-related classification and regression tasks. The results indicate that

the convnet feature surpasses the baseline MFCC feature in all the tasks considered. Additionally, it outperforms several previous methodologies that aggregate MFCCs alongside low and high-level music features.

Also there are some other works that are implementing CNNs or ViTs for this classification task or the other similar signal processing tasks.

## 2.2 Hypothesis

Even though various parts of the planned research have been done in different ways, it's noticed that there hasn't been an overall comparison study, and no work has been done on spectrograms. In [1], there was a comparison between ViT and CNN, but they used MFCC features as the data, and they didn't try different parameters. In [2], they worked on spectrograms, but they only coded the ViT architecture with fixed parameters, without making any comparisons. Similar limitations are observed in other studies as well. The main goal of our study is to understand where the ViT architecture might be successful in signal processing tasks with various parameters, comparing it with the success of the CNN under different parameters.

# 3
# Feasibility

This section includes the technical feasibility, legal feasibility, economic feasibility, labor and time planning and gantt diagram of the project.

## 3.1 Technical Feasibility

The software and hardware feasibility of the system is listed in Section 3.1.1 and Section 3.1.2.

### 3.1.1 Software Feasibility

The elements to be used to create the system are listed below.

- Python
- Librosa – Digital Signal Processing
- MatPlotLib
- NumPy
- Tensorflow – Neural Network Implementation
- Online Jupyter Notebook Editor (Google Colab)

### 3.1.2 Hardware Feasibility

- Remote Runtime Environment (Google Colab) – T4 GPU

## 3.2 Legal Feasibility

Since the study used open source software and was not motivated by profit, there were no legal problems.

The licenses of the programs used in the project are listed below.

- Librosa: ISC License
- Tensorflow: Apache License 2.0

## 3.3 Economic Feasibility

In the study; Since it uses a free public data set, Python libraries, and there is no institutional/company study held to a separate standard, there is no economic burden other than our own computers that we use during the study. Table 3.1 and 3.2 clearly show the expenses.

**Table 3.1** Operating Budget Table

| Job Title | Employee Count | Weekly Budget |
|---|---|---|
| Software Developer | 1 | 30000 TL |
| Systems Analyst | 1 | 20000 TL |
| Data Scientist | 1 | 12000 TL |
| Project Manager | 1 | 45000 TL |
| Total | | 107000 TL |

**Table 3.2** System and Tools Budget Table

| | | |
|---|---|---|
| Development Computer | Computer | 40000 |
| Remote Runtime Environment | Google Colab | Free |

## 3.4 Labor and Time Planning

The project is divided into work packages. Workforce and time planning has been made for each work package. A gantt diagram has been prepared for existing work packages. This project is aimed to be completed in one period (3 months). Two fourth-year computer engineering undergraduate students will take part in the completion of the project. Table 3.3 shows those students' information and positions.

**Table 3.3** Employee Title Table

| Student Number | Name - Surname | Roles |
|---|---|---|
| 17011071 | Hatice Demir | Software Developer, Project Manager |
| 17011057 | Tarık Aytek | System Analyst, Data Scientist |

## 3.5 Gantt Diagram

The threads and scheduling of the system can be examined in Figure 3.1. In the project, wide time intervals are given to the work pieces. The project could be completed within the expected time.

| ID | Name | Day | Start | Finish |
|---|---|---|---|---|
| 1 | Determining The Requirements of The Project | 5 | 02.10.2023 | 06.10.2023 |
| 2 | Literature Review | 3 | 09.10.2023 | 11.10.2023 |
| 3 | Determination of Technologies to Be Used in The Project | 5 | 16.10.2023 | 20.10.2023 |
| 4 | Examining The Documentation of The Necessary Libraries | 5 | 23.10.2023 | 27.10.2023 |
| 5 | Obtaining Spectrograms and Mel-Spektrograms From Audio Data | 4 | 30.10.2023 | 02.11.2023 |
| 6 | Implementation CNN model | 4 | 06.11.2023 | 09.11.2023 |
| 7 | Training CNN models | 6 | 13.11.2023 | 18.11.2023 |
| 8 | Implementation ViT model | 5 | 20.11.2023 | 24.11.2023 |
| 9 | Interim Report Writing | 3 | 29.11.2023 | 01.12.2023 |
| 10 | Training ViT models | 7 | 04.12.2023 | 10.12.2023 |
| 11 | Training Models with Different Parameters | 5 | 11.12.2023 | 15.12.2023 |
| 12 | Coding for Measurement Compration The Performances of CNN and ViT Architectures | 4 | 18.12.2023 | 21.12.2023 |
| 13 | Comparison of The Performances of CNN and ViT Architectures | 5 | 25.12.2023 | 29.12.2023 |
| 14 | Final Report Writing | 5 | 01.12.2023 | 05.12.2023 |

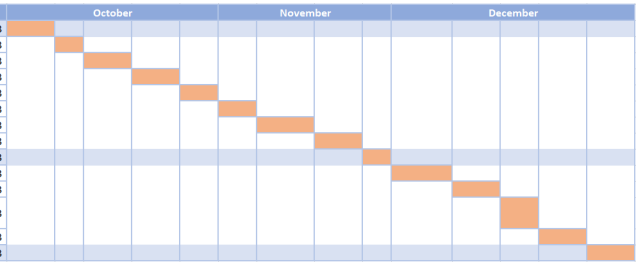


**Figure 3.1** Gantt Chart

# 4
# System Analysis

## 4.1 Goals of the Study

In this project, which aims to make a comparative analysis of CNN architectures and ViT architectures with different parameters, the process of obtaining a suitable data set must be applied first. Then, the architectures will be designed and comparison tables will be obtained. By examining the comparison tables, a study will be carried out on the usage areas of the architectures. Important target steps for the success of the study are listed below.

- Digital Signal Processing – Obtaining Image Data Set from Audio Data Set:

Obtaining spectrogram and mel-spectrogram images in png format from wav format data using signal processing techniques through the Librosa library

- CNN Training – The Process of Training and Testing the CNN Model with the Data Set:

The Process of Training and Testing the CNN Model with the Data Set Creating CNN models via the Tensorflow library, determining various model parameters, feeding the created models with the data set and measuring their performance

- ViT Training – Training and Testing the ViT Architecture with the Dataset:

Creating ViT models via the Tensorflow library, determining various model parameters, feeding the created models with the data set and measuring their performance.

- CNN vs ViT – Comparison of Performances on GTZAN Dataset

Comparing CNN and ViT performance using parameters in the architectures as independent variables

## 4.2 Requirements Analysis

First, it is crucial to determine the dataset to work on. The main datasets that are used in the academic world were considered for this project. FMA, Million Song Dataset, and GTZAN datasets were those which were evaluated. In this project, the requirement is not large amount of data to achieve high accuracy. As the main aim of the project is comparing different architectures and parameters, a relatively small but effective dataset would fit better. It is concluded that GTZAN dataset best meets these requirements. It is a dataset that consists of a collection of 10 genres with 100 audio files each, all having a length of 30 seconds. It can be said that it is the most used dataset in the field.

Second important point is to decide which language and libraries should be used. Since Python is the main language used in this field and is easy to implement related architectures and visualize the content; it was the best choice. Although C++ or R could be alternatives, they were not chosen because the code does not need to be exceptionally fast or efficient for this specific task.

For signal processing, Librosa was selected because it is basic and effective enough for this project. Essentia was another option, but Librosa was a better choice for using in Python and Windows.

Library for implementing neural networks was also a crucial point. The main options were PyTorch and Tensorflow, and we decided to go on with Tensorflow as we had some experience coding with this library. There are no significant differences between the two libraries for this project.

## 4.3 Performance Metrics

Accuracy will be used as the performance measure for this task because it's reliable and precise. Comparisons will be based on this metric.

# 5
# System Design

System design was grouped under 4 main headings. It was grouped under 4 main headings. Details of each section are included in subsections.

## 5.1 Obtaining Spectrogram and Mel-Spectrogram from Audio Data

Mel-Spectrogram: It is obtained using STFT. An example of a Mel-Spectrogram is presented in Figure 5.1.

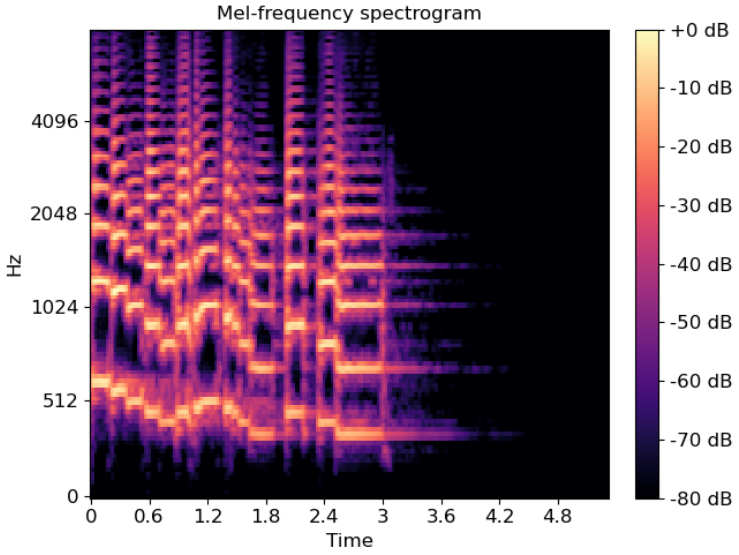


**Figure 5.1** Mel-Spectrogram Example [6]

Audio data (an array representing a time series) and sampling rate (the number of samples per second) are captured.

librosa.feature.melspectrogram(y=y, sr=sr): This function creates a mel-spectrogram using the audio data called y and the sampling rate called sr. Mel-spectrogram visualizes how frequency changes with time. Mel-spectrogram represents spectral

content using the frequency scale that is more suitable to human hearing perception.

`librosa.power_to_db(spectrogram, ref=np.max)`: This function converts the power of the calculated spectrogram into decibels. This transformation allows for better visualization of the sound. The ref=np.max argument converts the power relative to the highest reference value (the value calculated by the np.max function).

`librosa.display.specshow(spectrogram_db, sr=sr, x_axis='time', y_axis='mel')`: This function visualizes the mel-spectrogram. It takes the spectrogram converted to decibels named spectrogram_db and displays the time as the x-axis and the frequency as the y-axis using the sr sampling rate.

Spectrogram: It is obtained using STFT.

Audio data (an array representing a time series) and sampling rate (the number of samples per second) are captured.

y, sr = librosa.load(audiofile.wav'): The command loads the audio data from the specified file into the y variable and the sampling rate into the sr variable. sr specifies the sampling rate (such as 22050 Hz or 44100 Hz).

D = np.abs(librosa.stft(y)): The command STFT the audio data. This operation returns an array containing time and frequency information. With np.abs() we get the absolute value of this transformation because we are usually only interested in the magnitude of the frequency. `librosa.amplitude_to_db()`: The function transforms the output of the STFT on a logarithmic scale relative to a given reference level (here the highest value with np.max). This provides better visualization of the spectrogram.

librosa.display.specshow(): The function is used to visualize the spectrogram. The parameters x_axis='time' and y_axis='linear' specify that the x-axis will display time and the y-axis will display frequency [7].

## 5.2 CNN Models and Parameters

CNN: is a type of artificial neural network that is widely used in the field of deep learning and is especially effective in image recognition problems. Basic components and operating principles of CNNs:

- Convolution:

Convolution layers are used in CNNs. These layers apply one or more filters on input data (for example, an image). These filters slide over the input data to extract features. For example, edges, corners or other features can be extracted by this convolution process.

- Pooling:

Pooling layers are used to reduce the feature map produced by convolution layers and highlight important features. This reduces the overlap of features while reducing the computational load of the network.

- Activation Functions:

An activation function is often used after the convolution layers. Especially ReLU (Rectified Linear Activation) is widely preferred. Activation functions are used to increase the learning ability of the network and model nonlinear relationships.

- Fully Connected Layers:

CNNs often result in fully connected layers. These layers are used to classify features obtained from previous layers or for tasks such as regression.

CNNs are used successfully in visual data analysis problems, especially image recognition, object recognition, face recognition. However, it can also be applied in different fields such as time series data, text data. Layers of CNNs are trained by updating their weights throughout the learning process. In this training process, features of the input data are extracted and then used to perform a specific task such as classification or regression. In conclusion, CNNs are a deep learning architecture that can handle complex data structures and is effective in feature extraction and pattern recognition.

Parameters:

Batch Size: Specifies the size of each mini-batch used during training. Instead of processing all of the training data at once, it is processed in small mini-batches. It optimizes the training process by reducing memory usage and processor/GPU load. The values to be used in this project are 32 and 64.

Height and Width of Images: Specifies the height and width of images. It is adjusted according to the sizes of the images in the dataset being studied. The dimensions to be used in this project are 224, 224.

Number of Classes: The number of classes in the data set is 10. validation_split: It is used to divide the training data into training and validation sets. This provides a dedicated data set to monitor the model's performance during training. In this project, the data set will be used as 80

Activation Function: Specifies the activation function used in neural network layers. For example, activation functions such as ReLU, sigmoid, tanh can be used. The ReLU function will be used in this project.

Optimizer: It refers to the optimization algorithm used in training the model. For example, optimization algorithms such as Stochastic Gradient Descent (SGD), Adam, Root Mean Squared Propagation (RMSProp) can be used. The Adam algorithm will be used in this project.

Epochs: It refers to the optimization algorithm used in training the model. For example, optimization algorithms such as SGD, Adam, RMSprop can be used. The Adam algorithm will be used in this project.

Number of Layers: It refers to the total number of layers in the neural network model. This includes all layers including hidden layers and output layer. The number of layers to be used in this project is 9.

## 5.3 ViT Models and Parameters

ViT: is a deep learning model used to process visual data and for tasks such as image classification. Unlike traditional CNNs, ViT relies entirely on the attention mechanism to process visual data. The structure of the ViT architecture is presented in Figure 5.2.

Its main features are:

Attention Mechanism: ViT processes image data using the attention mechanism. Images are divided into pieces (patch) and these pieces interact with each other through the attention mechanism. In this way, it can model long-range relationships without using pixel-wise convolution.

Patched Input: Images are divided into small pieces and these pieces are converted into a text-like sequence. This fragmented input is processed by the attention mechanism.

Pre-trained Large Models: In particular, ViT models are pre-trained on large sizes and often large data sets. This may increase generalization abilities.

Less Convolution Usage: Unlike traditional convolutional neural networks, ViT has

fewer convolution layers. Instead, the attention mechanism focuses on learning relationships within the image.

Suitability for Image Classification and Other Tasks: ViT models are particularly effective for image classification tasks. It can also be used in transfer learning and different tasks.

ViT models can better capture long-range relationships thanks to the processing of visual data by the attention mechanism. This can provide better performance, especially on large-scale datasets and a variety of visual tasks [8].

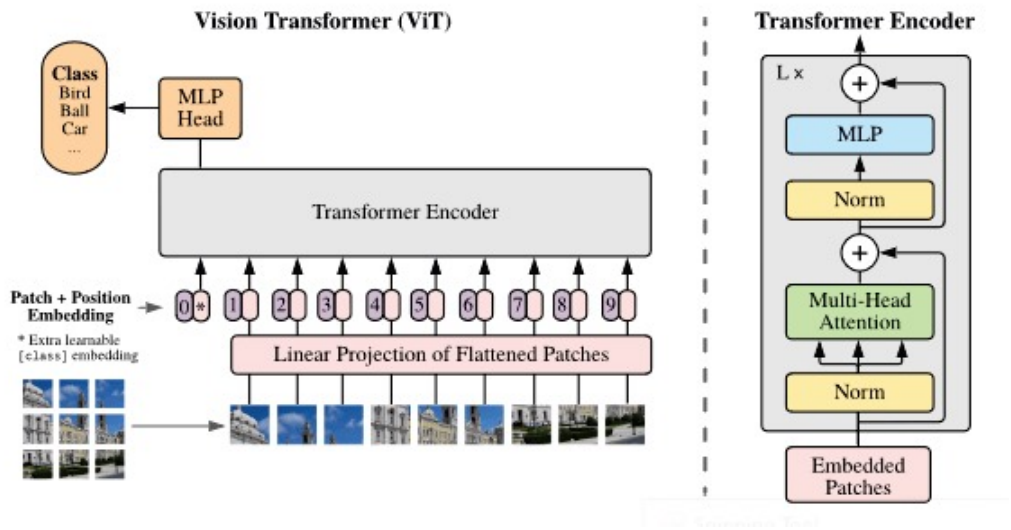


**Figure 5.2** Vision Transformer Design [8]

Parameters:

Patch Size: Specifies the window sizes used to split the image into segments. This is used to activate the attention mechanism by breaking the image into small parts.

Input Size: Determines what type of input sizes the model expects. For example, the width, height and number of channels of images.

Number of Attention Heads: Determines what type of input sizes the model expects. For example, the width, height and number of channels of images.

Number of Transformer Layers: Specifies the total number of attention layers in the model.

Hidden Size: It refers to the hidden dimension in each attention heading or each attention layer.

Number of Classes: It refers to the total number of classes to be classified.

Learning Rate: Specifies the learning rate to be used by the optimization algorithm. It affects how fast or slow the model learns.

Training Epochs: Specifies how many times the model will pass the training data.

A pre-implemented ViT model was chosen for our task. Even though transformers are not known as best for small datasets, training a small and efficient ViT from scratch was enough for purpose of comparison. After couple initial tests, it was seen that spectrogram images that are in shape of 775x308 were not optimal for feeding into transformer. The reason for this, input of ViT should be square shaped. Only scaling the rectangular shaped images would result in losing lots of information in spectrograms. So, we moved forward with splitting images into 2 pieces which are more square shaped. Another important consideration was the sequencing of patches that are fed into the model. As vertical axis is frequency and horizontal axis is time in spectrograms, we decided to split images into patches in two different way: one is patching in vertical order, another is horizontal order. Other parameter to use for comparison was chosen as patch size with the dimensions of 16x16 and 32x32.

## 5.4 CNN vs ViT Comparison

In this project, CNN and ViT performances will be compared depending on the parameters used as independent variables.

# 6
# Implementation

This section includes examples of generated data obtained so far in the study. Figures 6.1, 6.2, 6.3, 6.4, 6.5 and 6.6 are examples from the created data set. As can be seen, every figure of genre has their own characteristics. Also there is important difference between Mel and STFT spectrograms.

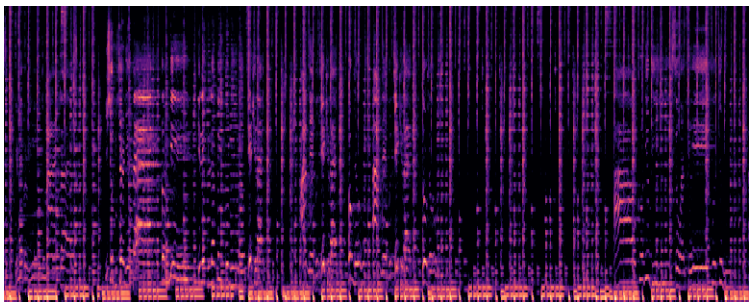

**Figure 6.1** Reggae Mel-Spectrogram

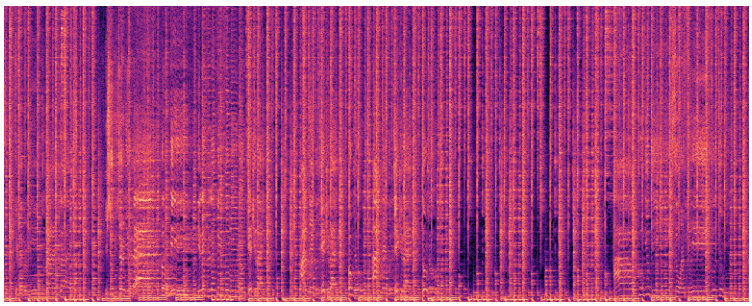

**Figure 6.2** Reggae STFT Spectrogram

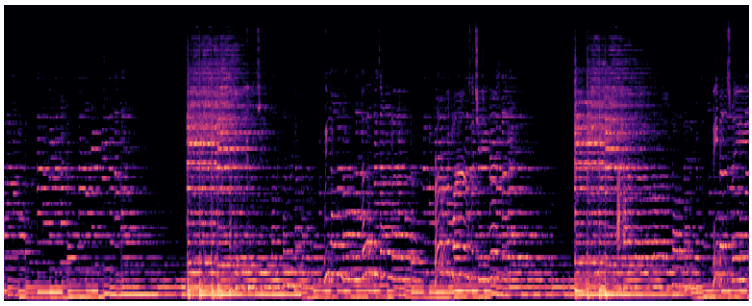

**Figure 6.3** Rock Mel-Spectrogram

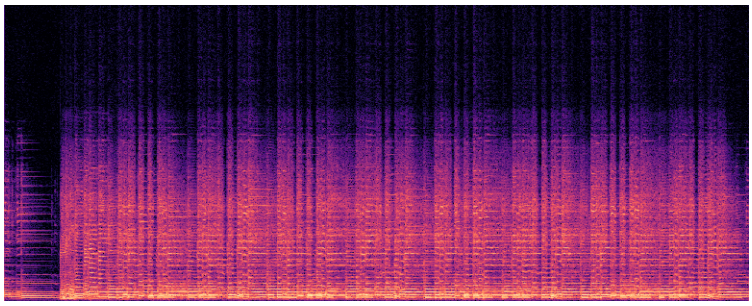

**Figure 6.4** Rock STFT Spectrogram

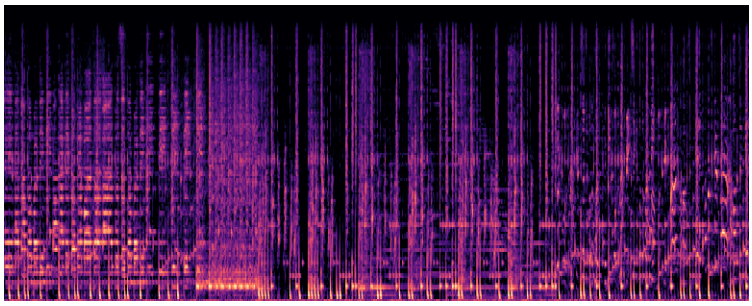

**Figure 6.5** HipHop Mel-Spectrogram

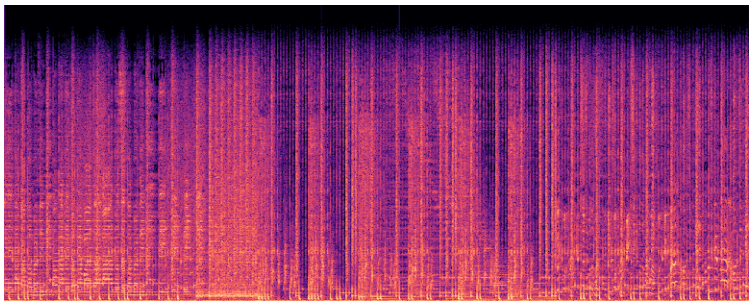

**Figure 6.6** HipHop STFT Spectrogram

# 7
# Experimental Results

## 7.1 CNN

This section contains tables prepared to measure the effects of various parameters on the CNN architecture. In order to apply comparative analysis, in the first stage, correlation analysis was applied and the correlative effect of each parameter on the result was determined. Later, in order to make comparisons, comparative results were prepared for each parameter by keeping the other parameters that gave the most successful results constant. The tables containing the results and the inferences made are as follows.

The effect of parameters on the models is shown in Table 7.1.

**Table 7.1** Impact on Test Accuracy

| | |
|---|---|
| img_size | 0.18 |
| num_conv_layers | -0.47 |
| num_filters | 0.16 |
| kernel_size | -0.25 |

The comparison result of image size is given in Table 7.2. For comparison, other hyper-parameters set their best as following:
Kernel Size: 3x3
Number of filters: 32
Number of convolutional layers: 3
It can be seen that the higher resolution results as higher accuracy. It also increases training time dramatically as it can be seen. It can be interpreted as trade-off between cost and performance.

The comparison result of number of convolutional layers is given in Table 7.3. For comparison, other hyper-parameters set their best as following:
Kernel Size: 3x3
Number of filters: 32

**Table 7.2** Impact of Image Size on Test Accuracy

| img_size | train_acc | test_acc | train_time | f1 | precision | recall |
|---|---|---|---|---|---|---|
| (256, 128) | 0,954 | 0,503 | 65,5 | 0.510 | 0.510 | 0.508 |
| (256, 256) | 0,996 | 0,593 | 99,4 | 0.593 | 0.598 | 0.599 |
| (128, 128) | 0,854 | 0,487 | 48,9 | 0.490 | 0.491 | 0.488 |

Image size: 256x256
It can be seen that 3 layers gives the best result. This can be interpreted as 2 layers are not enough model to generalize, but also 4 layers make it too complex for the task and start to over-fit the training data.

**Table 7.3** Impact of Number of Convolutional Layers on Test Accuracy

| num_conv_layers | train_acc | test_acc | train_time | f1 | precision | recall |
|---|---|---|---|---|---|---|
| 2 | 1 | 0,548 | 108.9 | 0.550 | 0.550 | 0.548 |
| 3 | 0,996 | 0,593 | 99,4 | 0.594 | 0.601 | 0.599 |
| 4 | 0,88 | 0,513 | 98.2 | 0.513 | 0.513 | 0.513 |

The comparison result of number of filters is given in Table 7.4. For comparison, other hyper-parameters set their best as following:
Kernel Size: 3x3
Number of convolutional layers: 3
Image size: 256x256
We observe that the success rate decreases when the number of filters doubles. The model becomes over-complex and starts to over-fit the training data as well.

**Table 7.4** Impact of Number of Filters on Test Accuracy

| num_filters | train_acc | test_acc | train_time | f1 | precision | recall |
|---|---|---|---|---|---|---|
| 32 | 0,996 | 0,593 | 99,4 | 0.599 | 0.591 | 0.592 |
| 64 | 0,976 | 0,487 | 234,6 | 0.489 | 0.488 | 0.487 |

The comparison result of kernel size is given in Table 7.5. For comparison, other hyper-parameters set their best as following:
Number of convolutional layers: 3
Number of filters: 32
Image size: 256x256
It can be seen that increasing kernel size results in losing information and decreases overall accuracy.

**Table 7.5** Impact of Kernel Size on Test Accuracy

| kernel_size | train_acc | test_acc | train_time | f1 | precision | recall |
|---|---|---|---|---|---|---|
| (3, 3) | 0,996 | 0,593 | 203 | 0.590 | 0.589 | 0.591 |
| (5, 5) | 0,975 | 0,462 | 206 | 0.462 | 0.462 | 0.461 |

## 7.2 ViT

Despite Vit architectures are not recommended for small dataset problems, the results were quite promising. First of all, splitting spectrogram images into two square-shaped images increased accuracy significantly. Another parameter, sequence type also turned out to be important as can be seen in Table 7.6. It can be interpreted that the model grabbed information better when closer time zones are sequenced and fed into model closer. Additionally, 32x32 size patches resulted in better accuracy which can be interpreted as reducing the patch sizes below this dimension could result in loss of information. Also highly expected over-fitting problem was not observed after multiple trains.

**Table 7.6** Accuracy of Vit Models

| **patch_size** | **sequence** | **val_loss** | **val_acc** | **train_time** | **f1** | **precision** | **recall** |
|---|---|---|---|---|---|---|---|
| 16x16 | Horizontal | 1.5594 | 0.5144 | 7m 12s | 0.541 | 0.558 | 0.552 |
| | Vertical | 1.3393 | 0.5553 | 7m 45s | 0.543 | 0.544 | 0.547 |
| 32x32 | Horizontal | 1.2699 | 0.5817 | 5m 59s | 0.548 | 0.572 | 0.562 |
| | Vertical | 1.3266 | 0.5962 | 6m 11s | 0.538 | 0.541 | 0.555 |

# 8
# Performance Analysis

The training times of the CNN and ViT models used in the study differed. It can be clearly said that training ViT models are much more costly than training CNN models. Although, the selected dataset, batch size and the system on which the models run were all the same, CNN models were trained in a much shorter time than ViT models. It was observed that the average training time of CNN models was around 1 minute, while the average training time of ViT models was around 6.5 minutes. It can be predicted that this difference will be dramatic on larger architectures and larger data sets. Also ram usage was dramatically higher in training process of ViT models. In the ViT architecture, we encounter this high usage because the relationships between all patches are determined in the same iteration. Enlarging the patch size can be used as a solution here, but it should be taken into consideration that it will also reduce the meaning to be extracted. That problems could prevent to tune hyper-parameters in the desired way in real life tasks or cause too much computational cost and may not be applicable.

# 9
# Conclusion

The dataset extracted by applying DSP methods on music files was used for training of approximately 100 models consisting of CNN and ViT architectures. Various parameters and approachs were used to train the models and the results were evaluated with different perspectives. As a result of comparisons, important outputs were obtained about the use of these architectures for the relevant task.

The main advantages of CNN architectures obtained from study are that they are much less costly to train, can produce successful results on smaller data sets, and are more explainable. In the cases where there is low computational resources and small amount of data, CNN architectures could be more advantageous to use because of those reasons.

On the other hand, ViT architectures has advantages such as performing better in large data sets, or when a pre-trained model is used, and also being applicable in generative systems. ViT architectures, which have pre-trained models in many areas, including music genre classification, far surpass architectures that use only CNN when fine-tuned with small data sets as in this task. In addition to classification tasks, very good success can be achieved by using ViT's decoder structure in generative artificial intelligence (AI) tasks such as music generation.

Overall, it seems that using the two architectures together with the right structure will give the best results. It can be said that many advantages revealed in this study can be evaluated together with the architecture using CNN in the lower layers and a pre-trained ViT in the upper layers. In addition, since the parameters which CNN and ViT architectures should be built in task were determined in the study, good success can be achieved by using these results.

# References


[1] Y. Khasgiwala and J. Tailor, "Vision transformer for music genre classification using mel-frequency cepstrum coefficient," in *2021 IEEE 4th International Conference on Computing, Power and Communication Technologies (GUCON)*, 2021, pp. 1–5. DOI: 10.1109/GUCON50781.2021.9573568.

[2] C. Zhen and L. Changhui, "Music audio sentiment classification based on improvied vision transformer," *American Journal of Computer Science and Technology*, vol. 6, no. 1, pp. 42–49, 2023.

[3] D. S. Lau and R. Ajoodha, "Music genre classification: A comparative study between deep learning and traditional machine learning approaches," in *Proceedings of Sixth International Congress on Information and Communication Technology: ICICT 2021, London, Volume 4*, Springer, 2022, pp. 239–247.

[4] Y. Yu, S. Luo, S. Liu, H. Qiao, Y. Liu, and L. Feng, "Deep attention based music genre classification," *Neurocomputing*, vol. 372, pp. 84–91, 2020, ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2019.09.054. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231219313220.

[5] K. Choi, G. Fazekas, M. Sandler, and K. Cho, "Transfer learning for music classification and regression tasks," *arXiv preprint arXiv:1703.09179*, 2017.

[6] *Librosa mel spectrogram documentation*, https://librosa.org/doc/main/generated/librosa.feature.melspectrogram.html, Accessed: 27 November 2023.

[7] Librosa Developers, *Librosa*, https://librosa.org/, Accessed: 27 November 2023.

[8] A. Dosovitskiy *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

# Curriculum Vitae

## FIRST MEMBER

**Name-Surname:** Hatice DEMİR
**Birthdate and Place of Birth:** 03.09.1999, İstanbul
**E-mail:** hatice.demir1@std.yildiz.edu.tr
**Phone:** 0506 850 30 36
**Practical Training:** Aktif Yatırım Bankası A.Ş.

## SECOND MEMBER

**Name-Surname:** Tarık AYTEK
**Birthdate and Place of Birth:** 26.06.1999, İstanbul
**E-mail:** tarik.aytek@std.yildiz.edu.tr
**Phone:** 0545 760 19 99
**Practical Training:** Khenda Teknoloji A.Ş

## Project System Informations

**System and Software:** Python
**Required RAM:** 2GB
**Required Disk:** 256MB