EE460 FINAL PROJECT

GENRE GUESSER

Caitlin, Evangelos, Le, Samrit

MAY 7, 2025

INTRODUCTION

OBJECTIVE

Build a machine learning model to **classify songs** into different genres using the GTZAN dataset.

DATASET

GTZAN contains 1,000 audio tracks across **10 genres** (e.g., pop, rock, classical, jazz).
We work with pre-computed 60 **features** and **spectrograms** to represent the audio.

MOTIVATION

Traditional genre classification relies on **handcrafted features** or metadata, which struggle with raw audio.

Our **hybrid approach** would combine:
CNNs to extract spatial patterns from spectrograms.
RNNs to capture temporal dynamics in the audio.

We aim to improve accuracy by training a **deep learning mode**l on 10-second clips from the GTZAN dataset.
We also test the model on newly generated spectrograms for **real-world validation.**

GOALS

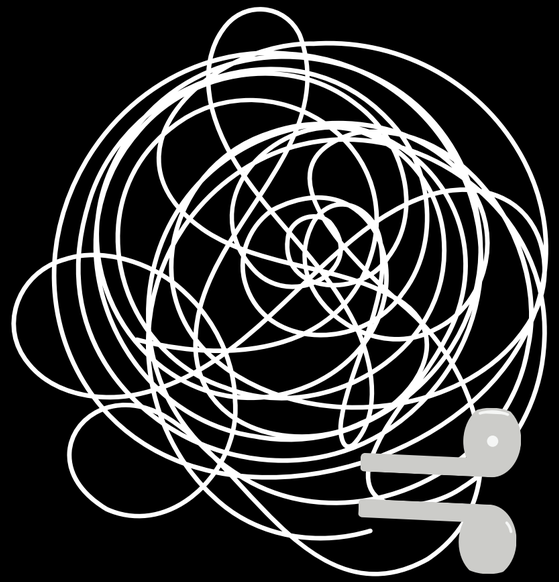
1. Explore different linear and non-linear models
2. Exploring learned features versus hand-designed features
3. Generate our own spectrograms and classify live audio

PIPELINE OF WORKING CNN MODELS

1. Train CNN with GTZAN dataset spectrograms
2. Obtain live audio .wav file
3. Obtain a spectrogram which is input to CNN
4. Get a genre classification

CHALLENGES FACED

Dataset is limited (1000 audio samples lead to around 3000 10-second spectrograms)



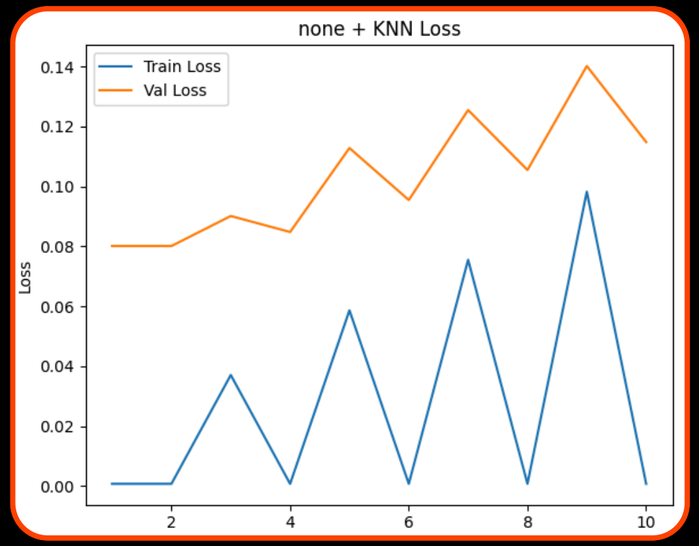
Overfitting of CNNs

Integration of CNN and RNN since output for CNN is supposed to be pulled out manually before Softmax layer and input to RNN

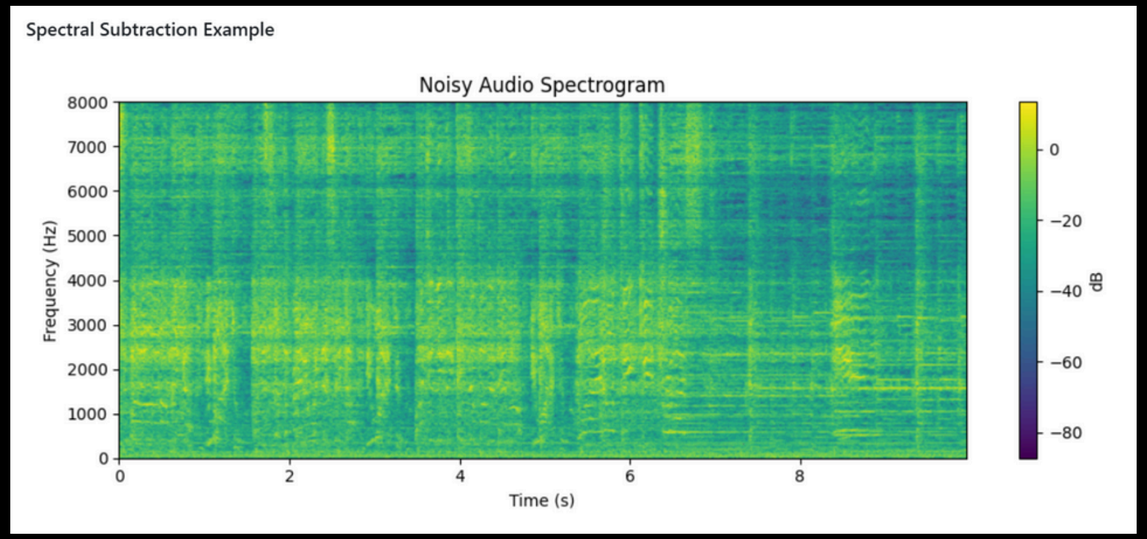
Model complexity and training time takes very long and limits the amount of models we could train.

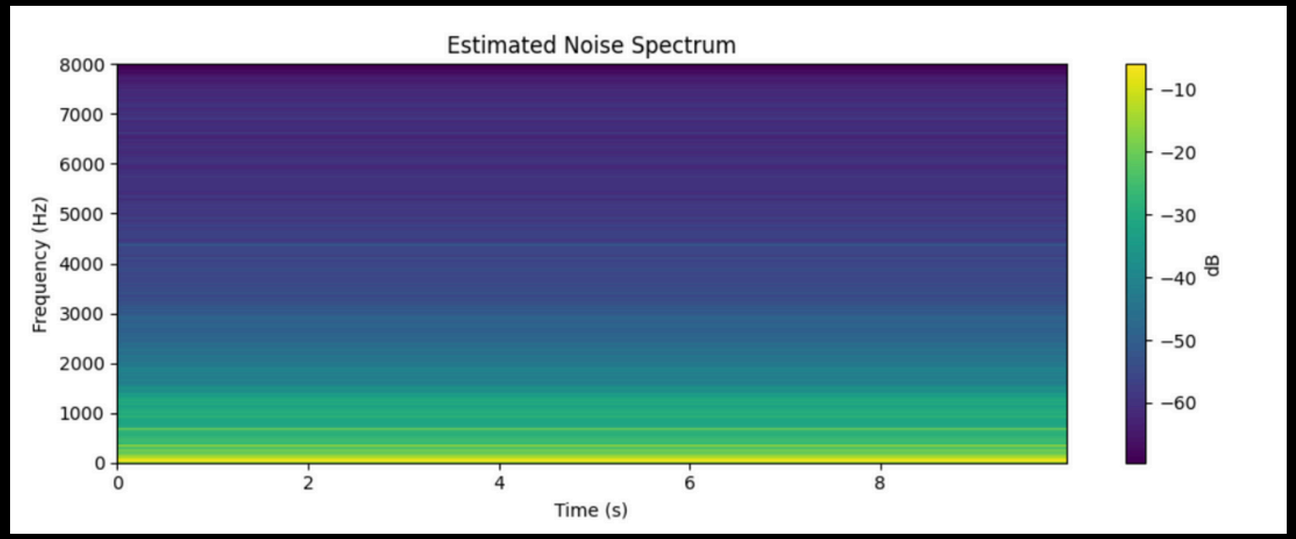
RESULTS – MODEL EXPLORATION

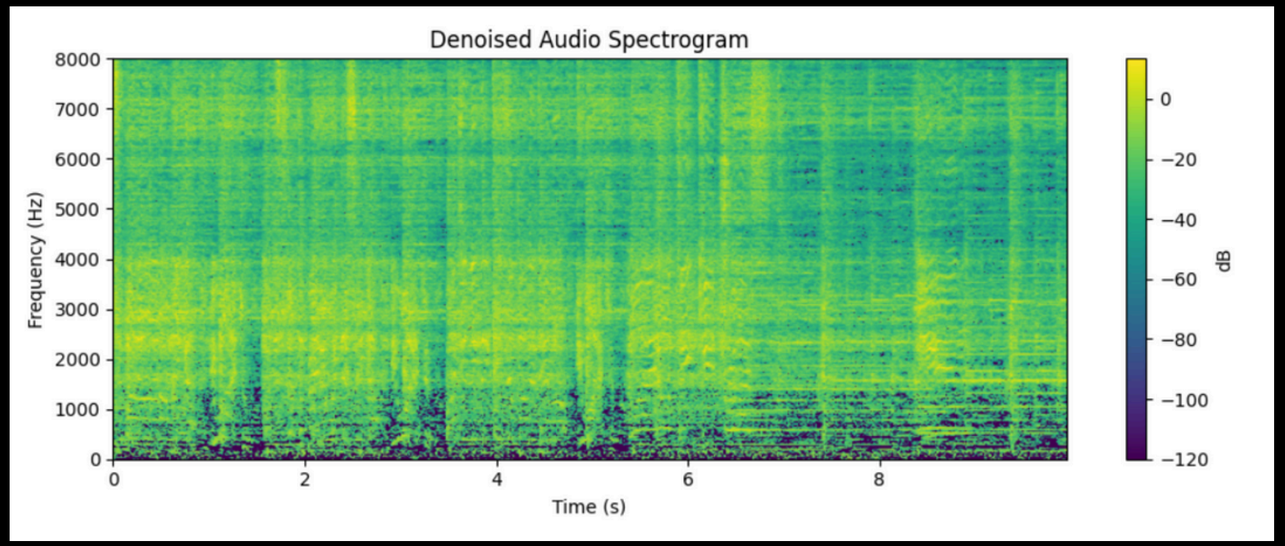
	None	PCA	RBF Kernel	Polynomial Kernel
Logistic regression	0.7131	0.6865	0.6111	**0.8853**
SVM	0.7532	0.7265	0.6284	**0.8846**
Perceptron	0.6418	0.5677	0.2175	0.6364
Ridge	0.6264	0.5657	0.5777	**0.8779**
Decision Tree	0.6384	0.6037	0.6144	0.6731
Random Forest	**0.8699**	**0.8626**	0.7545	**0.8466**
KNN	**0.9313**	**0.9159**	0.7799	**0.9293**
Neural Network	**0.9053**	**0.8746**	0.2141	**0.8859**



SPECTROGRAM BACKGROUND & DENOISING

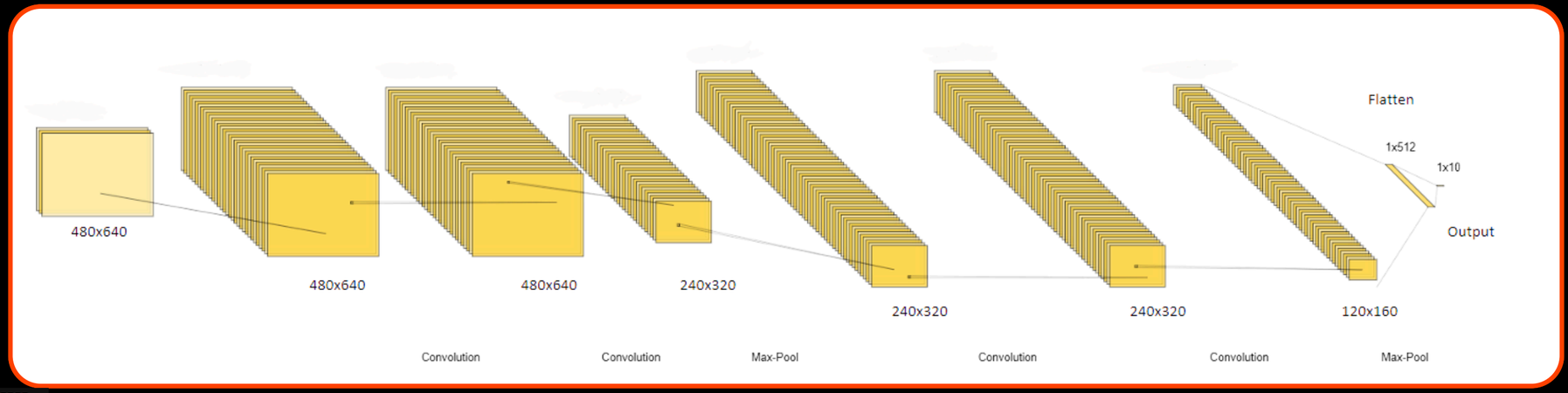






CNN ARCHITECTURE

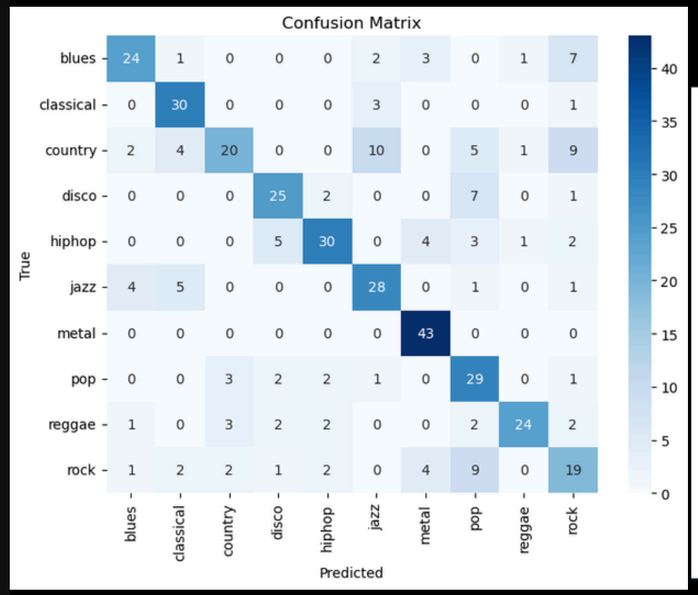
Mini-VGG



v5

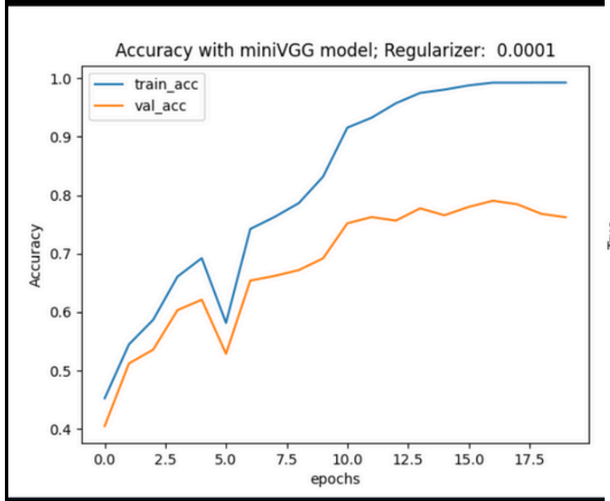
```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1          [-1, 32, 480, 640]             320
              ReLU-2          [-1, 32, 480, 640]               0
       BatchNorm2d-3          [-1, 32, 480, 640]              64
            Conv2d-4          [-1, 32, 480, 640]           9,248
              ReLU-5          [-1, 32, 480, 640]               0
       BatchNorm2d-6          [-1, 32, 480, 640]              64
         MaxPool2d-7          [-1, 32, 240, 320]               0
           Dropout2d-8          [-1, 32, 240, 320]               0
            Conv2d-9          [-1, 64, 240, 320]          18,496
             ReLU-10          [-1, 64, 240, 320]               0
      BatchNorm2d-11          [-1, 64, 240, 320]             128
           Conv2d-12          [-1, 64, 240, 320]          36,928
             ReLU-13          [-1, 64, 240, 320]               0
      BatchNorm2d-14          [-1, 64, 240, 320]             128
        MaxPool2d-15          [-1, 64, 120, 160]               0
          Dropout2d-16          [-1, 64, 120, 160]               0
           Conv2d-17         [-1, 128, 120, 160]          73,856
             ReLU-18         [-1, 128, 120, 160]               0
      BatchNorm2d-19         [-1, 128, 120, 160]             256
           Conv2d-20         [-1, 128, 120, 160]         147,584
             ReLU-21         [-1, 128, 120, 160]               0
      BatchNorm2d-22         [-1, 128, 120, 160]             256
        MaxPool2d-23           [-1, 128, 60, 80]               0
          Dropout2d-24           [-1, 128, 60, 80]               0
          Flatten-25                [-1, 614400]               0
            Linear-26                  [-1, 512]     314,573,312
             ReLU-27                  [-1, 512]               0
      BatchNorm1d-28                  [-1, 512]           1,024
          Dropout1d-29                  [-1, 512]               0
            Linear-30                   [-1, 10]           5,130
================================================================
Total params: 314,866,794
Trainable params: 314,866,794
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 1.17
Forward/backward pass size (MB): 857.83
Params size (MB): 1201.12
Estimated Total Size (MB): 2060.12
----------------------------------------------------------------
Using device: cpu
```

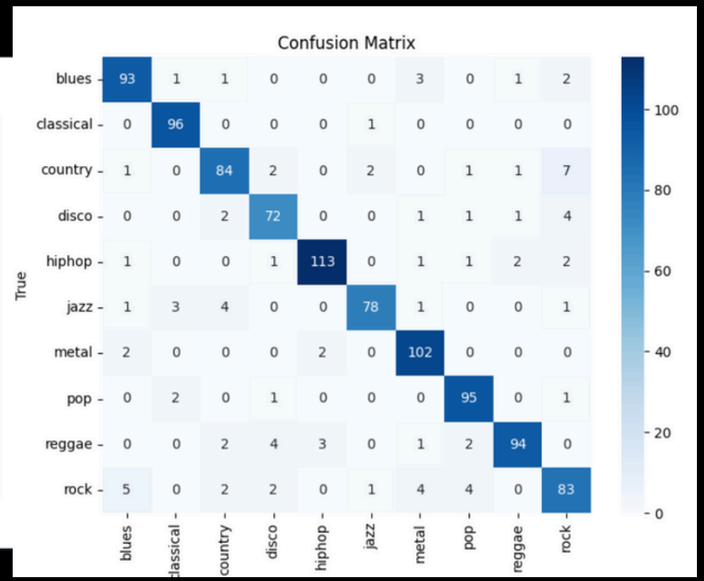
Test accuracy: 0.6865



v6

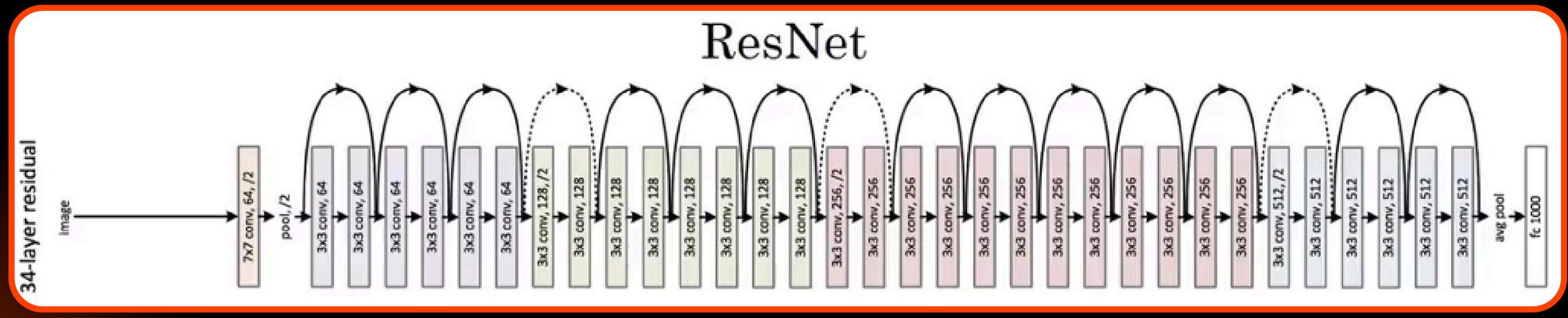
Test accuracy: 0.9056



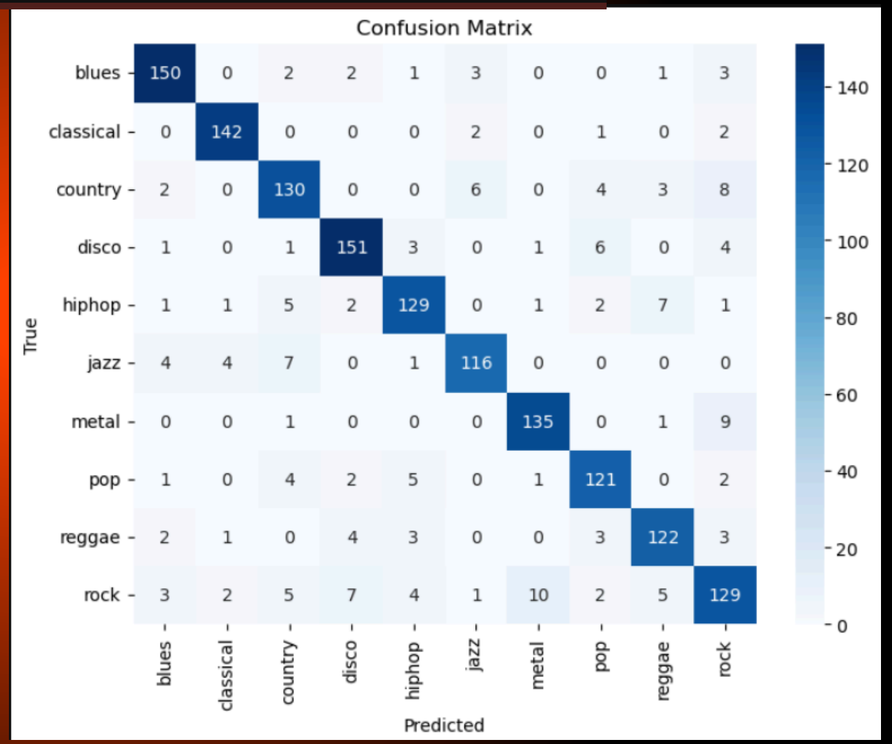


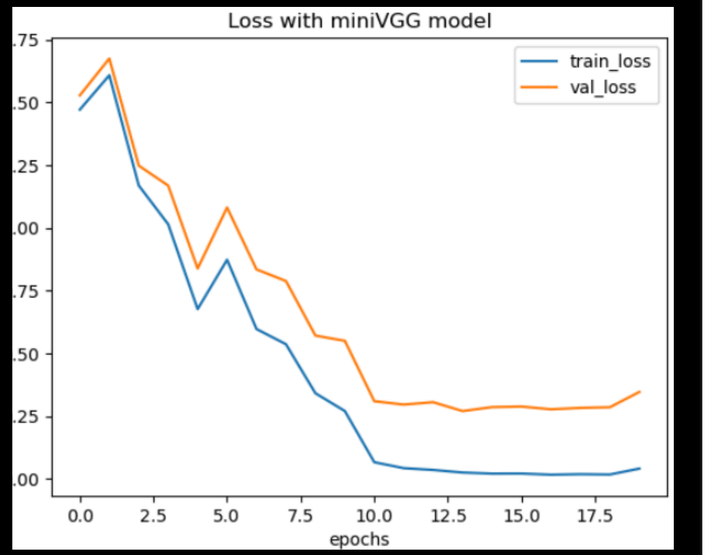
CNN ARCHITECTURE

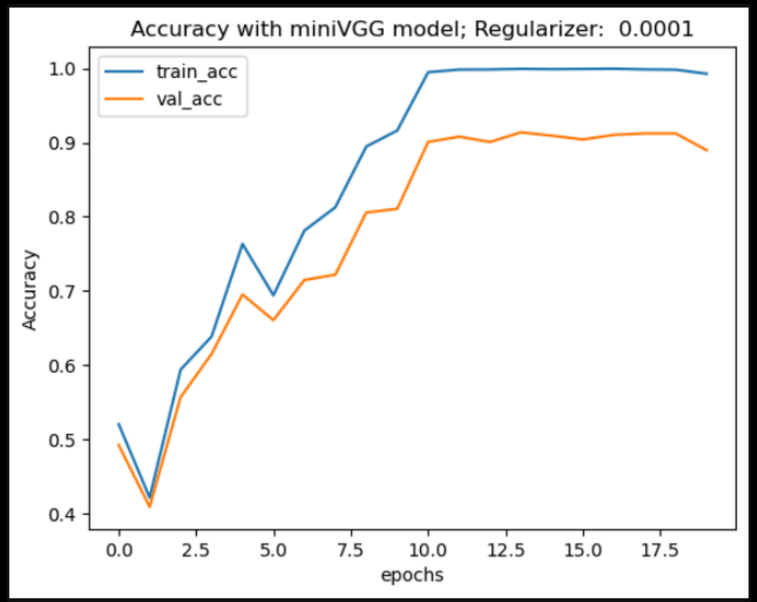
ResNet-34




```
Test accuracy: 0.8847
```







FUTURE WORK

CNN Architecture:
Test deeper CNNs (more VGG blocks, ResNets) to improve accuracy, potentially surpassing KNN.

RNN Integration:
Implement CNN-RNN fusion to model longer audio sequences and improve temporal understanding

Live Inference:
Enable real-time genre prediction from continuous audio, updating as the song plays.

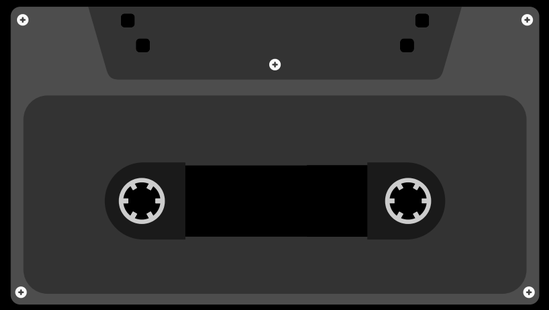

@11sixteen
@goodbyecalev
@dBn
LIVE DEMO

CONCLUSIONS ON FINAL MODELS

The CNN on 10-second spectrograms achieved a test accuracy of ~79%, outperforming many traditional machine learning approaches.

The Linear Model (KNN) on 10-second audiofiles achieved a test accuracy of ~93%, outperforming many traditional machine learning approaches.

Our CNN-based models successfully classified songs into 10 genres using spectrograms from the GTZAN dataset.



The project highlighted the importance of feature quality, data augmentation, and architecture choice in music genre classification tasks.

The hybrid CNN+RNN architecture (planned) is promising for capturing both spatial and temporal patterns in audio, though not fully implemented yet.

Our live audio inference pipeline worked effectively, with real-world spectrograms being denoised, processed, and classified by the trained model.

THANK YOU

Github link:
https://github.com/ccorduroy/genreguessr