Music genre classification using an audio spectrogram transformer

Project report

https://github.com/sbajamy/Music-Genre-Classification-Using-Audio-Spectrogram-Transformer

1. Introduction

In order to amplify user experience, music streaming platforms estimate users' preferences and suggest music other users with similar preferences like. Music genre classification is an important feature for this task. Thus, many methods try and automate the music genre classification sub-task.

In recent years, deep-learning-based methods are trying to deal with the fact that supervised learning on classification tasks with a high sample diversity, require a large labeled dataset in order to generalize well. Self-supervised learning [2,3], transfer learning [4,5,7] and data augmentation [4,6] were suggested as methods to immensely reduce the size of the required labeled dataset.

Mel-spectrogram of an audio signal can efficiently calculate the signal's frequencies as a human perceive them throughout time. Hence, intuitively it is a good starting point for a deep learning architecture that is trying to mimic a human labeling the genre of a music signal.

In this project we will utilize an audio spectrogram transformer trained on an extensive everyday sounds' classification dataset. We will use transfer learning methods and sound augmentations to adapt it to the music classification task using a small music genre classification dataset. Then we will test the final trained architecture generalization capabilities.

2. Background

2.1 Mel-spectrogram

Fourier transform on an audio signal, reveals its frequency content. The frequency changes through time though are hidden in this representation. Using Fourier transform on short overlapping time intervals of the signal (STFT), we can compensate for this predicament, trading some of the frequency resolution away (Figure 1a). Each time interval produces a frequency vector (Taking only the magnitude in decibels from the complex elements of the vector). Placing these window frequency vectors as columns of a matrix according to the widows' order in time creates a spectrogram.

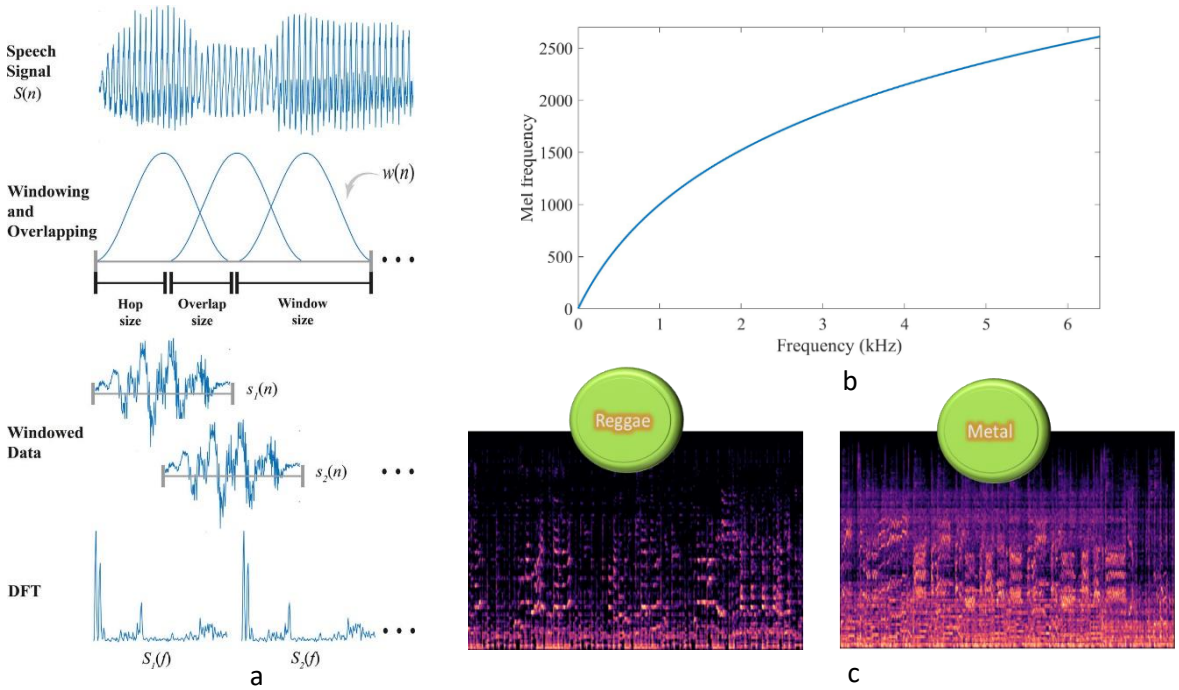

Figure 1

Humans' perceived frequency change logarithmically with the actual physical frequency (Figure 1b). When listening to two 100Hz distanced notes, the perceived frequency change decrease as the interval's lower frequency increase. The spectrogram's frequency axis is transformed to compensate for this discrepancy, creating a spectrogram with a linear perceived frequency axis- The Mel-spectrogram.

Mel-spectrograms of reggae and metal songs are shown in (Figure 1c). Intuitively the spectrograms of these samples match to the typical music genre properties. Raggae is a chill music genre and respectively its spectrogram content is sparse and concentrates on low frequencies. Metal however is an intense music genre and respectively its spectrogram content is dense and heavy.

2.2 AST: Audio spectrogram transformer [1]

The audio spectrogram transformer presented by Yuan Gong et al. actually has the architecture of a visual transformer (ViT).

Audio spectrogram transformer (Figure 2):

1. Split the spectrogram to 16x16 overlapping patches (stride 10).
2. Each patch is flattened and linearly projected into a 786 long token.
3. A learned class token (CLS) is appended to these tokens.
4. A learned positional encoding is added to each token.
5. The tokens are processed using a transformer encoder.
6. The processed class token is linearly projected into per class score vector.

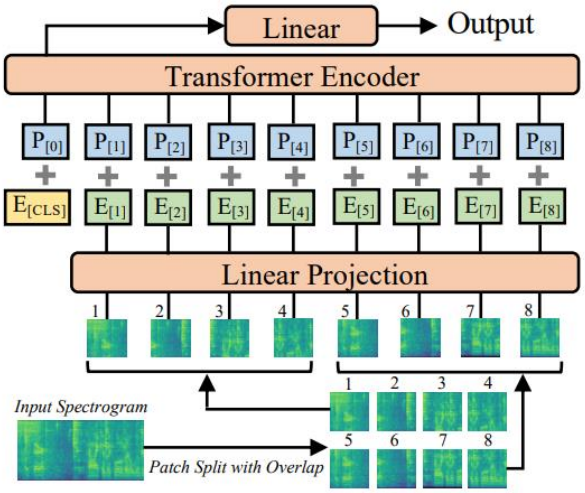

Figure 2

Yuan adjusted the weights of a ViT pretrained on ImageNet with a 224x224 color image as an input to one with an audio signal spectrogram as input (Size 128xt where t is the number time intervals used in the spectrogram calculation). This architecture is then trained on AudioSet dataset which is a large dataset of labeled audio clips excised from YouTube videos. The labels include bark noise, engine noise, music, speech etc.

2.3. DoRA: Weight-Decomposed Low Rank Adaptation[8]

A linear layer fine tuning method. Given a linear layer with weight matrix W_0, DoRA can be used to update them using learned low rank matrices A, B while keeping the overall magnitude of the original W_0:

$$W = \|W_0\| \frac{W_0 + \alpha AB}{\|W_0 + \alpha AB\|}$$

$$W_0 \in \mathbb{R}^{d_{in} \times d_{out}}$$

$$A \in \mathbb{R}^{d_{in} \times r}, B \in \mathbb{R}^{r \times d_{out}}$$

If $r < d_{in}, d_{out}$ then the rank of AB is r thus the learned update matrix has a limited rank with respect to W_0 maximum possible rank. Usually, $r << \frac{d_{in} \times d_{out}}{d_{in} + d_{out}}$ meaning the number of trained weights is significantly lower than W_0 matrix size. r and α are hyperparameters of the method.

3. Method

3.1 Augmentations

Training a model on a specific small dataset can easily lead to of overfitting. Therefore, in order to improve generalization and performance on the test set, we augmented the training set.

On each sample from the training set, we applied audio effects with TorchAudio AudioEffector using? FFmpeg format.

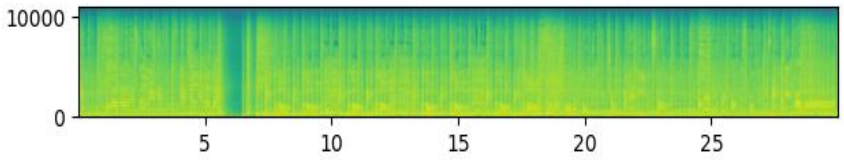

Figure 3: Mel spectrogram of a 30s track of the original sample

- Change of pace (Figure 4): Fast-paced and slow-paced variations.
  In real world scenarios, a sensible adjustment to the music pace won't change its genre class. We modified the tempo of the waveform to a lower or higher range.

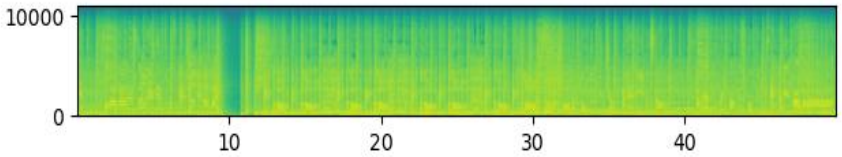

Figure 4: Slow paced original sample. Tempo < 1

- Frequency filters (Figure 5-6): High-pass and low-pass filtered samples.
  We want our model to be able to also perform well on different frequency sets. By filtering we can emphasize waveform information in a certain frequency range. High-pass filtered samples mimic recording like sounds, and low-pass filtered samples mimic hearing music far from its source or through sound barriers.

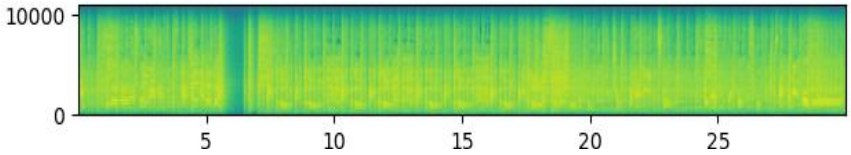

Figure 5: High-pass filter on the original sample

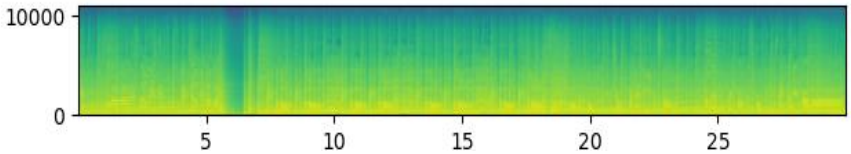

Figure 6: Low-pass filter on the original sample

- Echo effect (Figure 7): Sample with added reflected sound waves. Imitates outdoor acoustics, as expected from audio recordings of a live concert.

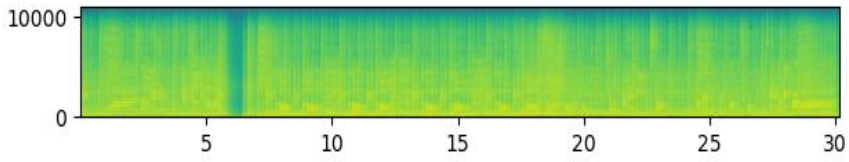

Figure 7: Echo effect applied on the original sample

- Bundle effect: A mix of filtering, repacing and echo effects.

Each effect parameters were chosen randomly for each sample. Some effects applied to the source signal will extend its duration. In these cases, we randomly crop a source duration slice from the extended signal. After accumulating the new variations, we added them to the training set.

3.2 Fine-tuned architecture

In order to fine tune Yuan's architecture on the new augmented training set, the following steps were performed:

1. Upload AST architecture and its AudioSet pretrained weights.
2. Freeze the architecture weights.
3. Replace the last linear layer 1 hidden layer MLP with GELU as non-linearity and dropout before each of its linear layers (Figure 8):

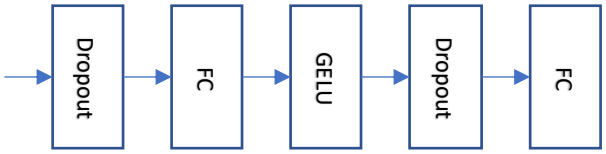

Figure 8

4. DoRA envelops the last linear layer in the feed forward listed encoder layers (Figure 9):

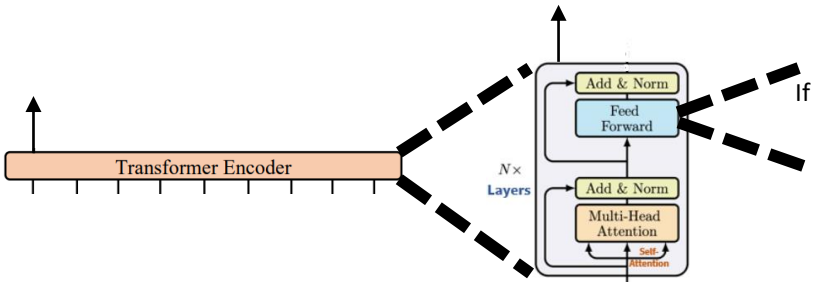

Figure 9

4. Experiments and Results

4.1 GTZAN dataset [9]

"GTZAN is a dataset for musical genre classification of audio signals. The dataset consists of 1,000 audio tracks, each of 30 seconds long. It contains 10 genres, each represented by 100 tracks. The tracks are all 22,050Hz Mono 16-bit audio files in WAV format. The genres are: blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, and rock."[10]

If not stated otherwise, a (8:1:1) train, validation, and test split was used.

4.2 Hyperparameters search

Hyperparameters search was performed using Optuna.
Training process hyperparameters searched:

- learning_rate- Factor multiplying the estimated weight update vectors.
- warmup_ratio- Ratio of minibatch training steps at the beginning of the training process in which the learning rate change linearly from 0 to the above defined learning rate.
- num_train_epochs - Total number of training epochs to perform.
- weight_decay- Factor of the L2 norm weight regularization loss added to the cross-entropy loss.
- batch_size - Number of samples used in each training step (We used 8 gradient accumulation steps thus the actual minibatch is the $8 \cdot \text{batch\_size}$).

Architecture hyperparameters searched:

- classifier_hidden_layer_size- Number of neurons in the hidden layer of the MLP used after the encoder.
- encoder_update_alpha- Factor of the added matrix (AB) in DoRA enveloped layers in the encoder.
- encoder_update_rank- The second dimension of matrix A and the first of matrix B in DoRA enveloped layers in the encoder.
- encoder_min_layer_to_update- Encoder layers from this layer to the last will have their last linear layer enveloped by DoRA (3.1.4).

We conducted 35 trails using optuna with validation set accuracy as its success metric. 19 of these trails were pruned. The chosen hyperparameters (Table 1):

learning_rate	~0.000132	classifier_hidden_layer_size	200	weight_decay	~0.0676
warmup_ratio	~0.085	encoder_update_alpha	~4.247	batch_size	2
num_train_epochs	13	encoder_update_rank	2	encoder_min_layer_to_update	10

Table1

Optuna estimation for the hyperparameters importance (Figure 10):

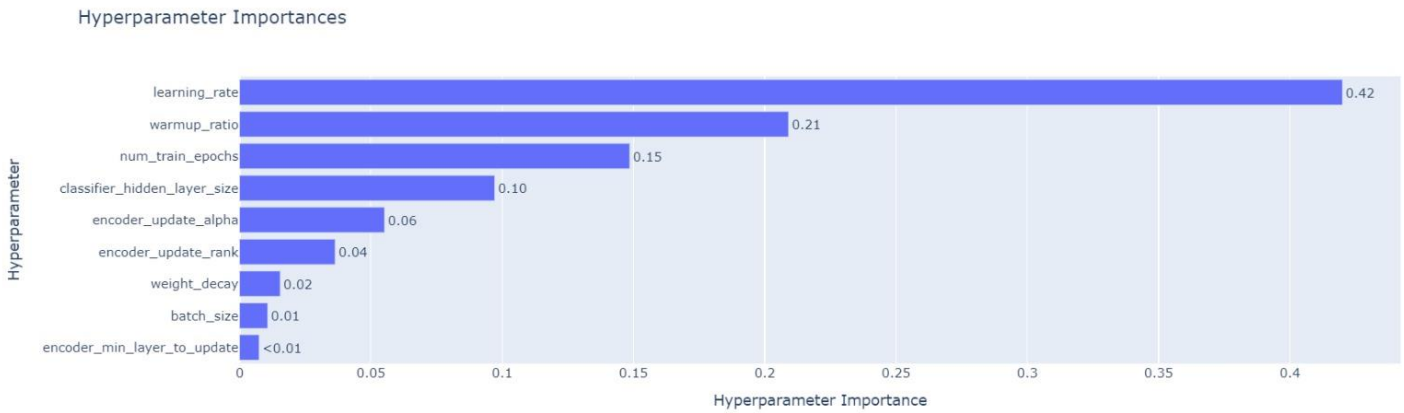

Figure 10

The most important hyperparameters according to the estimation are the learning rate, warmup ratio and the number of epochs.

4.3 Results

Using the hyperparameters from section 4.2, the validation set accuracy achieved is 87-90% and the test set accuracy achieved is 83-87%. The confusion matrix for the test set (Figure 11) shows us that rock is the hardest genre to detect (Consistent through multiple tests).

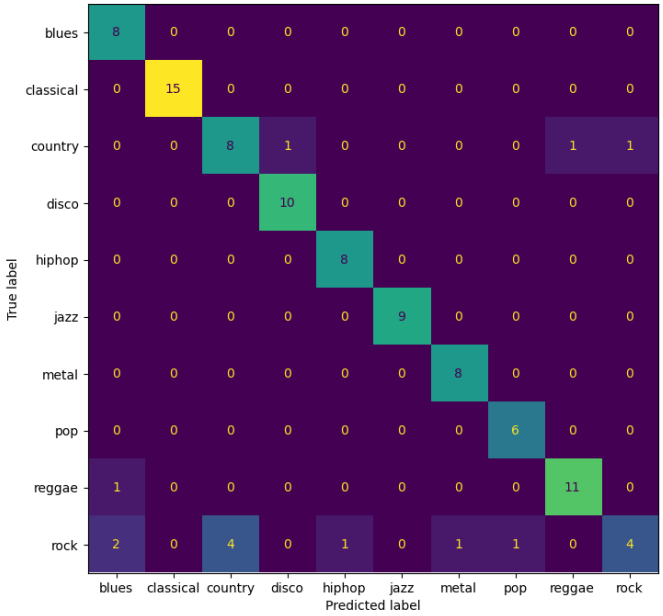

Figure 11

As far as we've seen, in articles [2,3], GTZAN data split into sets isn't presented. Setting this aside, our results are at least in par with their presented results (79.7-83.9% test set accuracy), though we haven't used any self-supervised learning techniques.
Comparing to former students' works (Table 2):

	Data Split (train:val:test)	Their test Accuracy (%)	Our 3 repetitions test accuracy interval (%)
Allouche&Katav[7]	6:2:2	77%	86-87%
Hoffman&Hadar[4]	4:1:- 4:1:5	78% (Val results) -	87-87% (Val results) 84-84%
Cohen&Shalit [5]	~5:2:3	70%	84-86%

Table 2

4.4 Generalization

In order to check our final method generalization capabilities, we tested its performance on songs from Jamendo website (A free music streaming platform).

Given an audio signal of a song:

1. Randomly sample 20 chunks 30 second each.
2. Calculate each chunk Mel spectrogram.
3. Calculate the model's response to the sampled chunks.
4. Calculate each chunk most likely class.
5. Calculate the majority class.
6. Calculate average chosen class probability from the chunks that predicted the majority class as a confidence measure.

Song	Genre	Estimated Genre	Confidence (%)
Burning You	Blues	Country	58%
Classical Clair De Lune	Classical	Classical	87%
Golden Standard	Country	Country	84%
Magenta Six	Disco	Pop	68%
Royalty	Hiphop	Rock	40%
For the Fifth	Jazz	Jazz	83%
Metal After Us	Metal	Metal	76%
Love You Anymore	Pop	Pop	96%
The River	Reggae	Pop	88%
Molotov Heart	Rock	Country	67%

Table 3

In Table 3 one can qualitatively see that our model is far from a perfect model in its generalization capabilities (5 correct out of 10 estimations). Self-supervision method used in [2,3] on a large music dataset may be used to alleviate this issue.

5. Conclusions and Future work

In this project we've shown SoTA in-par results can be achieved on a small music genre classification dataset utilizing MEL-spectrograms of the audio signal and a visual transformer architecture. These results were achieved only by applying audio augmentation and transfer learning techniques.

Possible directions for future work:

- Adding DoRA envelops to other linear layers in the encoder (First linear layer of the feed forward layer, key matrices, value matrices and query matrices).
- Hyperparameter search on the architecture of the MLP used after the encoder.
- Ensembling the suggested architecture with architectures that use the raw audio signal as input and architectures like CNN that inherently has an inductive bias.
- Using more audio augmentations or adding spectrogram augmentations.
- Randomly sample each minibatch augmentation rather than adding augmented samples in advance.
- Using self-supervision techniques like those presented in [2,3] on a large dataset before fine tuning on the small labeled dataset.

6. Ethics statement

6.1 Introduction

Project title: Music genre classification using an audio spectrogram transformer.

In order to amplify user experience, music streaming platforms estimate users' preferences and suggest music other users with similar preferences like. Music genre classification is an important feature for this task. We suggest a solution to this feature estimation sub-task using an audio spectrogram transformer on a small dataset utilizing transfer learning and audio augmentation techniques.

6.2 LLM answers

6.2.a Types of stakeholders affected by the project

- **Music Streaming Platform Users:** These are the end-users who will directly benefit from more accurate music recommendations.

- **Music Streaming Platform Management/Executives:** They are responsible for the overall performance of the platform and ensuring that users are satisfied with the service.
- **Data Scientists/Engineers:** These are the technical experts who will work on implementing and maintaining the music genre classification model.

6.2.b Explanation given to each stakeholder

- **Music Streaming Platform Users:** "We're working on improving your music discovery experience by enhancing the way our platform understands music genres. This means you'll get more personalized music recommendations that better match your taste, helping you find new songs and artists that you'll love".
- **Music Streaming Platform Management/Executives:** "We are developing a machine learning-based music genre classification system using audio spectrogram transformers. This will allow us to more accurately classify songs by genre and enhance our recommendation algorithms, leading to increased user engagement and retention".
- **Data Scientists/Engineers:** "We're implementing an audio spectrogram transformer for music genre classification, leveraging transfer learning and data augmentation. The goal is to improve the accuracy of our genre predictions on a small dataset, which will enhance the recommendation system and align with the platform's strategic goals."

6.2.c Responsibility for giving the explanation

- **Music Streaming Platform Users:** The responsibility for communicating with users typically lies with the Product or Marketing Team. They will craft the message in a way that resonates with users and explain the improvements in user experience through product announcements, in-app notifications, or email campaigns.
- **Music Streaming Platform Management/Executives:** The Project Lead or Data Science Team Lead would be responsible for presenting the project's progress and impact to the management team. This explanation might occur during strategy meetings or project reviews, where the focus is on how the model aligns with business objectives.
- **Data Scientists/Engineers:** The Project Lead or Senior Data Scientist would explain the project to the technical team. This would involve detailed discussions about the methodologies being used, the challenges of working with a small dataset, and the expected outcomes. Regular team meetings or technical documentation would be the medium for this explanation.

6.3 Reflecting on results

- "Music Streaming Platform Management/Executives" as a stakeholder type may be generalized to "Music industry" in order to include the artists that created the music in the platform.
- The LLM model completely ignored negative implications on the affected stakeholders. For example: Classifying new genre according to existing genres or classification mistakes, fixing users to a certain pool of genres, ignoring artists that find their music being classified offending, ignoring users that are not willing to share their information or to use other users' information in their playlist suggestions algorithm.

- The explanation to the users should include a consent declaration to the platforms monitoring and suggestion mechanisms and also instructions in case they wish to deactivate each mechanism.
- The artists explanation should include a consent declaration for automatic genre classification showing a concise explanation about the algorithm mistakes rate and instructions if they wish to manually label their music genre.
- The engineers explanation should include detailed information on the generalization capabilities of the model.
- The responsibility for explaining to the music industry about the allowed use limitations should also be handled by a state regulatory body.

7. References

[1] Yuan Gong, Yu-An Chung, James Glass, "AST: Audio Spectrogram Transformer," 2021.

[2] Rodrigo Castellon,Chris Donahue,Percy Liang, "Codified Audio Language Modeling Learns Useful Representations For Music Information Retrieval," in *ISMIR*, 2021.

[3] Daisuke Niizumi, Daiki Takeuchi, Yasunori Ohishi, Noboru Harada, and Kunio Kashino, "Masked Modeling Duo: Learning Representations," in *ICASSP*, 2023.

[4] Yuval Hoffman and Roee Hadar, "Music Genre Classification using Transformers," [Online]. Available: https://github.com/YuvalHoffman/Music-Genre-Classification-using-Transformers.

[5] Omer Cohen and Jonathan Nir Shalit, "MusicGenreClassifier," [Online]. Available: https://github.com/omercohen7640/MusicGenreClassifier.

[6] Ilay Yavlovich and Amit Karp, "Automatic Music Genre Classification," [Online]. Available: https://github.com/IlayYavlovich/MusicGenreClassifier-Technion.

[7] Itai Allouche and Adam Katav, "MusicGenreClassification," [Online]. Available: https://github.com/ItaiAllouche/MusicGenreClassification.

[8] Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng and Min-Hung Chen, "DoRA: Weight-Decomposed Low-Rank Adaptation," in *ICML*, 2024.

[9] George Tzanetakis, Georg Essl, Perry Cook, "Automatic Musical Genre Classification Of Audio Signals," in *ISMIR*, 2001.

[10] Lewis Tunstall, "Marsyas (Music Analysis, Retrieval and Synthesis for Audio Signals)," [Online]. Available: https://huggingface.co/datasets/marsyas/gtzan.