Music genre classification with deep neural networks
Audio Pattern Recognition Project

Paolo Cortis
Università Statale di Milano
Dipartimento di Informatica

***Abstract*—Music genre classification is a relevant task which stands as a basis for many real world applications, ranging from automatic labelling of audio tracks on streaming services, management of large databases of music files, audio retrieval systems and recommendation algorithms.**

In this work various machine learning algorithms are applied to this task on the GTZAN dataset. While performing experiments to understand how the analysis setup influenced the prediction performance of the methods, the focus was on determining the best features for the task and the correct methods to extract said features, as well as comparing models' performance with appropriate figures of merit. A Multi Layer Perceptron (MLP) architecture was used to handle the vectors of audio features, Convolutional Neural Networks (CNN) were instead trained on the STFT Spectrogram, Mel Spectrogram and Mfccs, and, in conclusion, a Multi Modal Neural Network (MMNN) was designed to capture both data modalities. While designing the MLP architecture Bayesian optimization was employed for the hyperparameter tuning phase.

Overall Mfccs proved to be the most effective feature for the music genre classification task, in particular when considering a representative metric such as the mean or variance.

I. Introduction

A music genre classification system is capable, given parts of a music line or an entire track, to classify it into some music class. As the quantity of songs being released on a daily basis continues to grow, manual curation becomes unfeasible and the need for accurate meta-data required for database management and storage purposes climbs in proportion. Being able to instantly classify songs in any given playlist or library by genre is an important functionality for any music streaming service, and the capacity of a system to automatically output correct and complete labeling of music and audio provides great value. In fact, as an example, such a model can serve as the basis for recommendation systems, where the genre is a crucial feature to advise users new compositions. This being an onerous, time-consuming and complex task, combined with the huge amount of data available, as well as a good numerical representation of the main features useful for the task, makes machine learning perfectly suited for the challenge. Furthermore, the study of automatic music genre classification provides a framework for developing and evaluating features for any type of content-based analysis of musical signals. A musical genre is hard to define but it can be characterized by the common characteristics shared by its tracks. These are typically related to the instrumentation, rhythmic structure, and harmonic content of the music. Data representation is crucial to convert audio data into a discrete format, this will in fact determine how much relevant information is retained. To this, end beats per minute, loudness, acoustics, energy, tempo, are all valid metrics, in this project, also spectral information was greatly considered, as it is known to be very much correlated with the genre of the track and works well with the model types employed.

II. Existing works

Machine learning techniques have been used for music genre classification for decades now. In 2002, G. Tzanetakis and P. Cook [1] used both the Mixture of Gaussians model and K-Nearest Neighbours along with three sets of carefully hand-extracted features representing Timbral Texture, Rhythmic Content and Pitch Content. The result was a model with 61% accuracy; more importantly they found out that Mfccs were the best performing feature. Their findings shaped the literature, as essentially all following works transformed audio data in a similar manner.

In a work of the REVA University [2] simple machine learning algorithms such as Logistic Regression, K-Nearest Neighbours, Random Forests, Support Vector Machines and Artificial Neural Networks, along with dimensionality reduction techniques, namely PCA, were applied on the GTZAN dataset. The results found that the combination of K-NN with PCA provided the highest accuracy of 77.41%.

In 2018 Mingwen Dong titled his work "Convolutional Neural Network Achieves Human-level Accuracy in Music Genre Classification" [3], proposing a simple Convolutional model to classify a short segment of the music signal that achieved human-level accuracy. It also found that the filters learned in the CNN resemble the spectro-temporal receptive field in the auditory system, essentially proving that CNN's filters transform the original Mel Spectrogram into a representation where the data is linearly separable for classification. Convolutional Neural Networks have shown to be incredibly accurate music genre classifiers [4], with excellent results reflecting both the complexity provided by having multiple layers and the ability of convolutional layers to effectively

identify patterns in Mel Spectrograms and Mfccs. This proves how these are solid features for the task and are better handled with this type of network, leading to better performances than Support Vector Machines and K-Nearest Neighbours.

Inspired from these works, this project compares audio Timbral Texture features, especially STFT Spectrogram, Mel Spectrogram and Mfccs to determine their level of relevance for the task when handled with a CNN architecture. Importance was also given to obtaining good results, exploring proper data processing and feature extraction techniques as well as adopting hyperparameter tuning to optimize the networks' architecture.

III. System overview

A. Dataset

The GTZAN dataset was initially proposed by G. Tzanetakis and is still one of the most popular music record dataset used for genre classification. It contains 1000 music track records coming from numerous sources like CDs, DVDs, radios and microphone recordings, these are 30 seconds in length, with 22050 Hz sampling frequency. The dataset comes with audio features (58 attributes such as mean and variance of Mfccs, spectral centroid, bandwidth, chroma, harmony, tempo and more) extracted considering both a 30 seconds window (the whole track) and a 3 seconds window, as well as an alternate representation of tracks as images via 30 seconds Mel Spectrogram. Genres in the GTZAN are blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae, and rock.

An analysis of the GTZAN music genre dataset [5] demonstrated that there are heavy artist repetitions, which can't be considered during the dataset split, additionally not all labels are correct, which makes high accuracy models impossible to obtain. There is also distortion in the audio files, namely clipping and skipping. Furthermore, the size of the dataset is small, which limits the capabilities of deep learning models.

B. Data processing and Feature extraction

The first goal was to extract and process robust features in an appropriate manner, starting with the GTZAN audio features data. Since data normalization is proven to improve the performance of models, a robust scaler was applied to these vectors, subtracting the median and dividing by the standard deviation between the interquantile range, operating a z-scoring while being robust to outliers.

Because data often contains some features that are either redundant or irrelevant and can negatively affect the model performance, the next step was feature selection, to choose a proper subset of the original pool of features. Feature selection reduces noise, improves model accuracy, and reduces training time by cutting the total number of features without incurring in much loss of information. To this end, identifying feature correlation with the output and correlation between features is helpful. This is because features not correlated with the output do not carry relevant information for the model to exploit for the task and can thus be removed; the same goes for features correlated with other features, these carry the same kind of information and result redundant. The correlation tests considered to perform feature selection were the Pearson, Spearman and Maximal Information Coefficient (Mic) test to identify linear, monotonic and non-linear correlation between features and the output.

To gather additional relevant data, feature extraction was the next step. This can be seen as the process of computing a compact numerical representation which can be used to characterize a segment of audio. The choice and design of descriptive features for the music genre classification task is relevant, as once robust features are extracted, standard machine learning techniques independent from the specific application area can be considered. Apart from the data coming with the dataset, the STFT Spectrogram, Mel Spectrogram and Mfccs on a 3s window were extracted directly from the audio files using the Librosa library implementation [6]. The Spectrogram was computed as the Discrete Time Fourier Transform over a 3 seconds window. The Mel Spectrogram represents the same spectral information projected onto the Mel scale, which mimics how the human hearing systems perceives sounds. Mfccs were calculated with the application of the Discrete Cosine Transform on the Mel Spectrogram, the early components are deemed to be the most relevant, especially when handling music, hence, in this project, the first 20 were considered.

Apart from a careful selection of the parameters of the algorithms to extract said metrics, not much can be done in terms of processing for these kind of features, still, Cepstral Mean and Variance Normalization over a sliding window was performed on Mfccs, a technique which is proven to minimize noise distortion by linearly transforming the cepstral coefficients to have the same segmental statistics. The complete processing pipeline of data is summarized in the block diagram (Fig. 1).

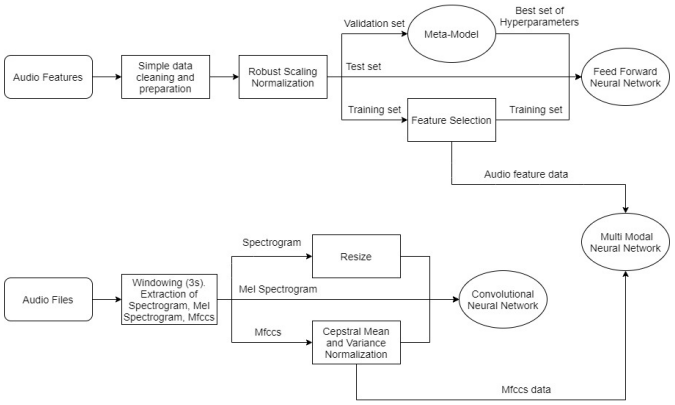

Fig. 1. Data Processing Pipeline

As stated, a relevant part of this project was to determine the best features for this task when considering the models proposed. To this end, the next step was to visualize the feature space, the dataset distribution and evaluate the difficulty of the task, this can also help to guide the development of adequate models. The Principal Component Analysis (PCA)

data decomposition algorithm was applied in order to reduce the high-dimensionality dataset into a lesser number of features to represent it. It is a statistical procedure used to decompose a multivariate dataset in a set of successive orthogonal components that exhibit the maximum amount of variance.

The simple clustering algorithm K-means was ran on the Mfccs space and on a different space considering other spectral features, providing valuable insight on the complexity of the classification task and on the quality of the features. The better performances on the Mfccs in terms of Mutual Information score, Adjusted Rand Index score, Mean Squared Error and Silhouette score, further confirm what can already be induced by looking at the two spaces: Mfccs appear to be more discriminating for the task then other features, following the current literature consensus (Fig. 2).

To further analyse the relevance of these features with respect to the task, the Boruta Feature Selection algorithm was run on the 3 seconds window dataset. It is a wrapper built around the random forest classification algorithm which gives a numerical estimate of the feature importance, allowing to separate relevant features from irrelevant ones.

C. Models

All the presented models were developed using Keras [7] with the TensorFlow backend. The models developed were Feed Forward Neural Networks trained audio features data, Convolutional Neural Networks trained on two dimensional Spectrogram, Mel Spectrogram and Mfccs data and Multi Modal Neural Networks to combine the previous two types of network and predict using both kinds of data.

The Feed Forward Neural Network consists of one input and output layer with a certain number of hidden densely connected layers in between. The activation function of these layers was set to the Relu function, while a Softmax function was employed for the output layer, the chosen optimizer was Nadam and the loss function was Categorical Cross-Entropy. To avoid overfitting of the model (learning the training set up to the noise without being able to generalize over unseen data) the regularization technique of dropout was implemented. The idea is to partially limit the network capabilities during training by "shutting off" some neurons. This forces the rest of the network to learn different aspects of the input and pick up more robust features, which in turn helps generalization. When designing a neural network, the choice of the architectural shape (number of layers, neurons, activation functions) and setting of learning hyperparameters (optimizer algorithm, batch size, learning rate) are critical for achieving high and reliable performances. At the state of the art, there is no unified method for finding the appropriate hyperparameters for a given task, and model selection is generally performed by relying on experience, involving empirical tests, or applying automatic methods which explore the hyperparameter space in a bounded domain. Bayesian optimization is a technique that learns from the observed performance of previously tried hyperparameter settings on the current task. This knowledge then helps to build a meta-model that can be used to predict which unseen configurations may work best on the task. It is a sequential design strategy for global optimization of black-box functions, and it is usually employed to optimize expensive-to-evaluate functions, in this case the training of a neural network. This method has proven to be an effective and cost-efficient solution to hyperparameter optimization and it is the hyperparameter tuning method chosen for this project. The hyper model is defined by specifying, for each of the critical hyperparameters, a range of possible values to explore for a set number of trails. The procedure then trains this meta-model using training data and tests it over validation data. Subsequently it returns the set of hyperparameters found that optimized the given evaluation metrics. In the context of this project hyperparameter tuning was applied when designing the MLP architecture for treating audio features data. While it could very well be used to compose the CNN and the MMNN as well, this was deemed not to be worth it as the literature found specific configurations hard to replicate with hyperparameter tuning. Furthermore, the lack of data does not allow for proper training of the meta-models, resulting in poor results with respect to hand made designs. The hyperparameters to set were the number of hidden layers (1-6), the number of neurons per layer (8-256) and the amount of dropout per layer (0.3, 0.4 or 0.5). To evaluate these optimized architectures also fixed architectures were designed: three layers for handling 30 seconds window data and four layers for handling 3 seconds window data. On the quality of features with respect to the task, to determine to what extent is the model exploiting Mfccs related metrics rather than other measures, a MLP was also trained on the same set of features (chroma, rms, spectral centroid, spectral bandwidth, rolloff, zero-crossing rate, etc.) excluding Mfccs.

A Convolutional Neural Network is great at handling spatial information such as images, or in this case Spectrograms and Mfccs. This type of network is based around the operation of convolution, consisting in the application of a filter to the input to create a feature map that identifies patterns and relevant features within the data. The Relu activation function, Softmax output function, Adam optimizer and Categorical Cross-Entropy loss were used for this type of architecture. After each convolutional layer, a max pooling layer is necessary to down sample, keeping only the maximum values within the feature map region, thus, the output is a feature map consisting of the most prominent portions of the previous one. The resulting building block of the Convolutional Neural Network consists of a convolutional layer, followed by a max pooling layer; multiple blocks of this kind compose the full network. The results of the convolutional portion of the network are then passed to a set of dense layers to obtain the final prediction. The architectures handling the Spectrogram, Mel Spectrogram and Mfccs follow this very same general structure, with slight differences in the number of filters per layer depending on the input size.

Data usually comes in different modalities, which carry different information, like in this case, where both audio features and Mel Spectrogram/Mfccs bear significant inputs.

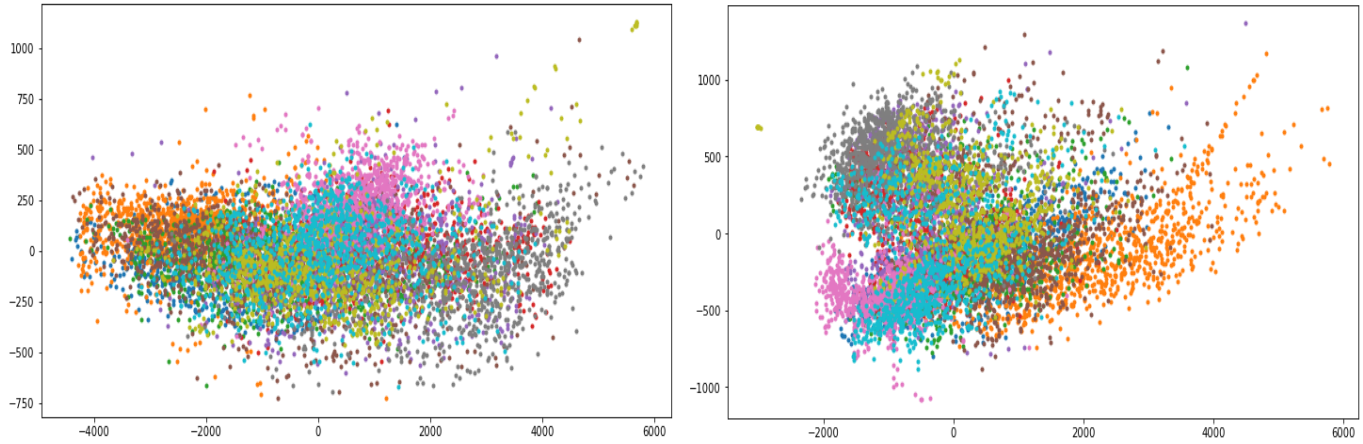

Fig. 2. Feature space (PCA) and Mfccs Space (PCA)

Multi modal learning attempts to model the combination of different modalities of data, in this case via a Multi Modal Neural Network, a model that is able to jointly represent and exploit the information of both data modalities (audio features data and Mfccs data). To shape this network the layers of a MLP and a CNN are concatenated, each receiving their specific input, the outputs are then combined into a final dense layer and output layer, resulting in a single architecture which is trained at the same time as a whole MMNN.

IV. Experimental results

A. Experimental setup

In order to train a model, tune his hyperparameters with a meta-model, and evaluate his performance, the dataset was split into training, validation and testing set. To have a statistically sound estimate of an architecture performance, multiple models are built and trained, each with the same architecture, over different portions of the data (holdouts) and, the average performance of those, is considered as an estimate of the overall performance of the architecture. The technique used to generate the holdouts was the stratified Monte-Carlo method, producing each time a different arrangement of the training, validation and testing set, while keeping roughly the same class balance. 10 holdouts were considered to train and evaluate each model, 20% of the whole dataset was used for testing, 80% for training where 20% of it was reserved for the validation of the meta-model. During the training phase each model went through the dataset for 100 epochs (although early stopping was set) with a batch size of 128. For audio features data, both the removal of uncorrelated features and the hyperparameter tuning step were performed over the training set only, so to never have data leakage and introduce bias into the model. Note that the optimal set of hyperparameters found with the meta-model based on Bayesian optimization depends on the specific training set considered at each holdout and may differ at each iteration of the main loop. For audio features data, correlation analysis was performed, then the model was built and trained over the entire training set with the optimal set of hyperparameters found by the meta-model and tested over the test set.

When evaluating each model performance, a plethora of metrics are available, in this case, the Accuracy, the AUROC (Area Under Receiver Operating Characteristic) and the AUPRC (Area Under the Precision-Recall Curve) were deemed to be the most relevant. The Accuracy returns the number of correct predictions over all samples considered. The AUROC is the area under the curve where the x axis is the false positive rate and the y axis is the true positive rate. The AUPRC is the area under the curve where the x axis is the recall and the y axis is the precision. After the stratified holdout procedure was complete, an average of these values was performed, returning the 10-holdout performance estimate of the architecture.

B. Results

Table I provides a comprehensive view of all the model results comparing train and test accuracy, AUROC and AUPRC metrics, with a barplot displaying the test accuracy of all models (Fig. 3).

The MLP treating audio feature data showed very good results, proving that even a simple Feed Forward Neural Network is indeed a good fit for the task when handling robust features.

The models trained on the 3 seconds window audio features perform significantly better than those trained on the 30 seconds features (extracted over the entire track). This, probably, has less to do with the 30 seconds features being worthless, and more to do with the amount of data being simply too small to tune the model and learn accurately the task.

The MLP architectures where hyperparameter tuning with Bayesian optimization was performed do not show significant improvements with respect to the hand-made fixed ones. Again, this probably has to do with the size of the dataset; these hyperparameter tuning methods require a significant amount of data to train the meta-model, optimize the network, and provide models with better performances.

TABLE I
MODEL PERFORMANCE COMPARISON

Model	train Accuracy	train AUROC	train AUPRC	**test Accuracy**	**test AUROC**	**test AUPRC**
Fixed MLP 30s	0.750	0.970	0.832	0.641	0.937	0.706
Fixed MLP 3s	0.962	0.999	0.993	0.879	0.989	0.947
Tuned MLP 30s	0.812	0.980	0.887	0.675	0.946	0.746
Tuned MLP 3s	0.951	0.998	0.988	0.864	0.988	0.936
Tuned MLP no Mfccs 30s	0.552	0.904	0.596	0.496	0.879	0.521
Tuned MLP no Mfccs 3s	0.717	0.958	0.798	0.679	0.948	0.757
CNN Mel Spectrogram 30s	0.476	0.871	0.522	0.405	0.840	0.443
CNN Spectrogram 3s	0.870	0.990	0.941	0.705	0.951	0.787
CNN Mel Spectrogram 3s	0.856	0.988	0.935	0.724	0.954	0.804
CNN Mfccs 3s	0.926	0.997	0.978	0.789	0.970	0.868
MMNN	0.942	0.997	0.983	0.863	0.986	0.932

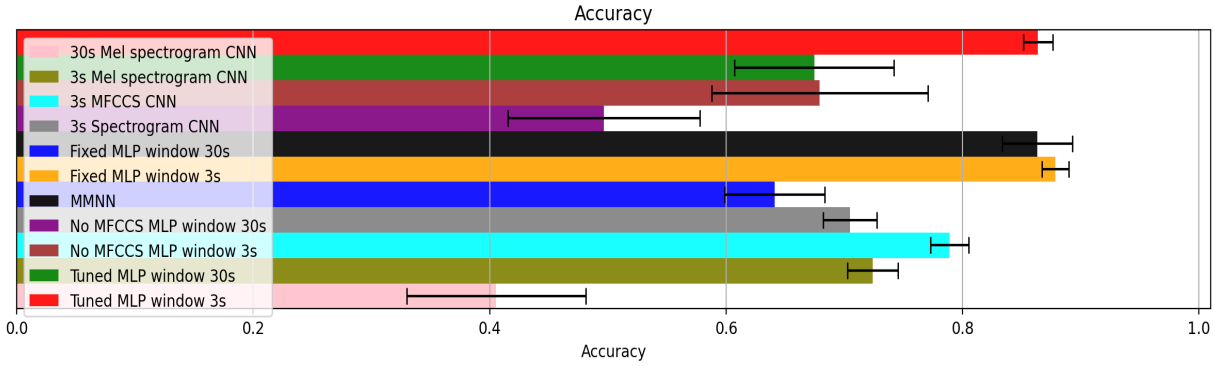

Fig. 3. Model Accuracy Comparison

On the quality of the features with respect to the classification task, Boruta Feature Selection determined that all of them are relevant and can be exploited to predict the genre of the track, with K-means demonstrating better performances in the Mfccs space. An important note is that the audio features include the mean and variance of Mfccs over the considered window. Thus, to delve deeper and determine to what extent is the model exploiting these Mfccs related metrics with respect to other measures, it is to note that the model trained on audio features excluding Mfccs performed significantly worst, with a decrease of -21.41% in test Accuracy, -4.05% in test AUROC and -19.12% in test AUPRC, with respect to the tuned MLP given Mfccs data too, when considering a 3 seconds window.

The CNN trained on the GTZAN 30 seconds window Mel Spectrogram is the worst performing one overall. This was to expected given the poor quality of these images and the low amount of data.

The CNN handling the 3 seconds window Spectrogram performed slightly worst than the one trained on the 3 seconds window Mel Spectrogram with a decrement of -2.62% in test Accuracy, -0.31% in test AUROC and -2.11% in test AUPRC, proving that the Mel Spectrogram carries more relevant information when handling music for this task.

The CNN treating Mfccs is the best performing Convolutional Network with an increase of +8.98% in test Accuracy, +1.68% in test AUROC and +7.96% in test AUPRC with respect to the one handling the Mel Spectrogram.

Mfccs truly seem to be the most rich feature for this task, as it is with many other audio pattern recognition applications. However, there is an interesting result to report: the MLP trained on the audio features outperformed the CNN trained on Mfccs with an increase of +11.41% in test Accuracy, +1.96% in test AUROC and +9.10% in test AUPRC. This result is much more related to this specific classification task and, indeed, follows the literature consensus stating that it is often desirable to extract metrics representative of the whole music signal with a final long-term averaging operation, highlighting the salient features of the track. In fact, while the Mfccs extracted on the 3 seconds window present the complete Mfccs' sequence, the audio features incorporate the mean and variance of Mfccs across the entire window, which, in turn, provide more information on the track for predicting the genre.

The Multi Modal Neural Network trained over audio features data and Mfccs data results are respectable, but do not improve on previous models. This is probably related to the already known faults of multi modal learning [8]. In fact, even if the model receives more information and should match or outperform its single-modal counterpart, these results confirm the opposite: the best Single Modal Network outperforms the Multi Modal Network. The main causes for this performance drop are that Multi Modal Networks are more likely prone to overfitting, due to increased capacity, and that the two different modalities overfit and generalize at different rates, so training them jointly with a single optimization strategy is sub optimal.

V. Conclusions and future work

After visualizing the feature space and analysing the complexity of the task, different features were considered for training different Neural Network models for the music genre classification task. Related to the GTZAN dataset it was noted how the lesser data availability of the 30 seconds window hindered the performances of models, whereas, when features were extracted over a 3 seconds window, results were much better. Hyperparameter tuning of the model via Bayesian Optimization didn't surpass fixed models, however this could change with an increase in data supply. Mfccs were further proven to be the best feature when classifying musical genre, surpassing the Spectrogram and Mel Spectrogram when handled with a Convolutional Network. A MLP given mean and variance of audio metrics, in particular Mfccs, outperformed a CNN trained on the top 20 Mfccs sequence, suggesting that there is value in condensing measurements in a single representative feature vector of the whole window length. However, trying to combine these two different kinds of audio representation didn't lead to better performances when handled with a Multi Modal Neural Network.

Following this work, similarly to what was done by Gabriel Gessle and Simon Åkesson [9], who performed analysis by comparing a CNN and a LSTM, the next step could be to combine, in a Multi Modal fashion, a Recurrent Neural Network and a Convolutional Neural Network to capture both temporal and spatial information in music, and strive for better results.

References

[1] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. IEEE Transactions on Speech and Audio Processing, 10(5):293-302, 2002

[2] Jeffery Ho Kin Pou, H Keshav Rao, Geetansh Bhambhani, Joel Joseph, Suriya Prakash J. Music Genre Classification using Machine Learning, REVA University, 2021.

[3] Mingwen Dong. Convolutional neural network achieves human-level accuracy in music genre classification. CoRR, abs/1802.09697, 2018.

[4] Derek A. Huang, Arianna A. Serafini, Eli J. Pugh. Music Genre Classification, 2018.

[5] Sturm Bob L. An Analysis of the GTZAN Music Genre Dataset, Aalborg Universitet, MIRUM 2012 - Proceedings of the 2nd International ACM Workshop on Music Information Retrieval with User-Centered and Multimodal Strategies, Co-located with ACM Multimedia, 2012.

[6] McFee, B. et al. Librosa: Audio and music signal analysis in python, 2015.

[7] Francois Chollet et al. Keras. https://github.com/fchollet/keras, 2018.

[8] Weiyao Wang, Du Tran, Matt Feiszli. What Makes Training Multi-modal Classification Networks Hard?, 2020.

[9] Gabriel Gessle and Simon Åkesson. A comparative analysis of CNN and LSTM for music genre classification, KTH Royal Institute of Technology, Stockholm, Sweden, Semantic Scholar, 2019.