Music genre classification using 1D and 2D CNN models

Liviu-Ștefan Neacșu-Miclea

Department of Computer Science, Babeș-Bolyai University
1, M. Kogalniceanu Street, 400084, Cluj-Napoca, Romania
E-mail: nlic3194@scs.ubbcluj.ro

Abstract – *Music is a part of entertainment in our everyday lives. With the increasing volume of songs and albums comes the need to quickly look for audio pieces matching one's taste and preferences. One way to index songs is by tagging them according to genres, a task which can be automatized using machine learning approaches. This study analyzes the performance of convolutional neural networks in music genre classification by working with 1D and 2D representations of data. This work performs a thorough data analysis on MagnaTagATune dataset and trains a sample-based simple 1D CNN, 1D CRNNs (RNN/LSTM/GRU), a chunk CNN and 2D CRNNs on the top-10 genres classification. The results are evaluated using accuracy, precision, recall, AUC-ROC and AUC-PR. The study concludes that GRU-based methods work best, while the others are still comparable with state-of-the-art.*

1. Introduction

Music is a form of artistic expression that relies on harmonious and rhythmical use of voice and instruments sound. The style of musical interpretation is a testament to the human creativity, sensitive and exploring nature, in a permanent need to find the artistic solvent medium which holds the emotion and meaning in way that feels right to both the composer and the audience. This idea sparks a colorful palette of genres, subgenres and crossovers. In our modern technologically developed world where information is everywhere and for all tastes, being capable to create good music is futile unless there is a way that allows people to find it easily in the never ending sea of multimedia libraries. This cements the importance of approaching the music genre classification problem, which tries to find a way to extract information and features from audio data that hints to its inclusion into a certain genre [Gho+23]. This work is assessing the performance of a few common deep network architectures on music genre classification. It is also questioned whether data representation (either audio samples or spectrogram) has the impact on the classification task. In the next sections, literature approaches will be presented, then the models for evaluation will be introduced, using 1DCNN, 2DCNN, RNN/LSTM/GRU and CNN-RNN combinations. The experimental results will be discussed and compared with literature. It will be concluded that GRU-based models perform the best, although the performance gap between selected models is negligible and influenced by the particularities of the classified genre.

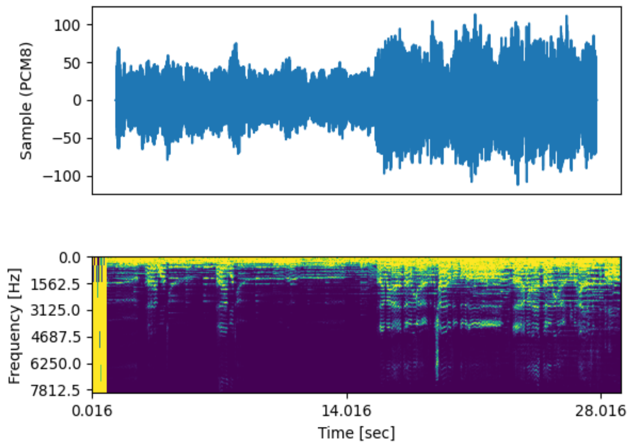

Fig 2.1. Audio sample wave and spectrogram

2. Related work

2.1. Audio signal processing

The output of a digital recording device is a sequence of numbers interpreted as discrete observations of sound wave amplitude changes captured at a constant interval of time, called the sample rate [Déf+22]. A Fourier transform is a method used to map a sequence from time domain $x(t)$ into the frequency domain $X(\omega)$. For a given audio signal, the evolution of frequencies obtained by applying discrete Fourier transform (DFT) across consecutive time window subintervals can be visualized using a spectrogram (*Fig.2.1*). A spectrogram in a two-dimensional heat map, where the axis represent time and frequencies. The intensity of the color at a given time and frequency denotes the amplitude of the frequency at that particular time [OS10].

2.2. Song datasets

The publicly available labeled song datasets share some common points. The audio tracks are around 30 seconds long each and the number of genres can vary from tens to up to two hundreds, with subsets of a smaller number of genres being extracted from the huge original datasets for ease of use and prototyping purposes. The GZTAN dataset contains 1000 song extracts classified into 10 popular genres (e.g. blues, classical, hip-hop, metal etc.) with balanced distributions [Gho+23]. The Free Music Archive (FMA) has a total of 106574 audio files and 161 unbalanced classes [Gho+23]. The Million Song Dataset (MSD) provides a contemporary music data in the form of chroma and timbre vectors, along with artist and song related metadata [Gho+23].

The MagnaTagATune (MTAT) dataset is a collection of 16kHz, 32kbps, mono-channel, MP3, 30 seconds long tracks. The 25863 sound clips coming from 5405 songs were labeled with 188 different tags, with multiple labels per instance [Law+09]. The collection process was undertaken in the format of an online game, TagATune, where users could link a song with a tag. The validity of a label was accorded if more than two players applied the same tag to a song excerpt [Law+09]. The MTAT dataset will be used in the experiments described in this work.

2.3. Evaluation metrics

Since a song could belong to multiple genres at the same time, this is not a categorical classification. It is enough to train a model to predict whether a song has a certain tag or not, therefore leading to a number of binary classification sub-problems. Metrics for binary classification tasks are employed : accuracy, precision, recall, F-1 score, area under receiver operating curve (ROC-AUC) and area under precision-recall curve (PR-AUC), which is considered more descriptive for the case of unbalanced datasets [Won+20].

2.4. Approaches in literature

Usual classifier methods have been considered, such as k-NN [MZ18], Random Forest [MZ18], Support Vector Machines [Gho+23] [MZ18], Logistic Regression [Gho+23] [MZ18], reportedly giving poor performances (Gosh et al., 46% overall test accuracy [Gho+23]). As expected, research focus shifted to deep learning methods as they provided a more appealing playground for this kind of classification task [KKB21]. For example, Gosh et al. also evaluate a 3-layer ReLU-activated ANN to achieve 67% accuracy [Gho+23].

Alternatively, seeing the input as one-dimenional data (in the case of amplitude samples representation), or a sequence of time-dependent vectors (in the case of the spectrogram), one convenient idea is resorting to 1-D convolutional neural networks (1D CNNs). Moreover, treating the spectrogram itself as an image suggests using a 2-dimensional CNN for feature extraction, basically converting the task into a computer vision problem [KKB21].

On the other hand, another natural intuition on sequencial data leads to solutions involving recurrent neural networks. Kostrzewa et al. tries both 1D CNN and a 2D CNN followed by two LSTM layers on the Mel spectrogram representations of the FMA dataset, along with ensemble networks of 1D or 2D CNNs, RNNs and CRNNs, obtaining up to 57% accuracy with their best ensemble model and 52% with individual model (2D CNN) [KKB21]. Gosh et al. obtain 77% accuracy with one-dimension CNN, 90% CRNN and 88% with a parallel CRNN on Mel spectrograms of 480 instances from the Million Song dataset [Gho+23]. A similar study is performed by Matocha and Zieliński on FMA using 2D convolutions on Mel spectrograms, reaching 60% accuracy on eight genres [MZ18]. Allamy and Koerich mention the possibility of extracting additional data from audio signals (timbral texture, beat, pitch), as also encountered in MSD, which may serve as input for neural networks [AK21]. Their work features training a 1D ResNet CNN on GZTAN, scoring an overall 72% accuracy and 74% with augmentation that involve noise and pitch alterations [AK21]. Other CNN approaches in music genre classification include fully convolutional networks (FCN), Musicnn (which uses horizontal and vertical filter to capture timbre and temporal patterns), Sample-level CNN (the normal 1D CNN), Harmonic CNN (which introduces stacked harmonic blocks instead of the usual CNN operations to predict a distribution of Discrete Cosine Transform outputs) [Won+20]. This problem has also seend transfer learning approaches [Cho+17].

Won et al. conducted a thorough study on CNN architectures trained on MTAT, MSD and the higher quality MTG-Jamendo Dataset [Won+20]. The results can be consulted in *Table 2.1*.

The MTAT dataset is used by Stastny et al. in their work analyzing performances of semisupervised approaches involving Self Organizing Maps (SOM), k-means, Multilayer Perceptron (MLP) on feature vectors obtained from audio instances, followed by Learning Vector Quantization (LVQ) as a fine tuning step [SSF13]. Their results range between 85.2% and 92.0% accuracy [SSF13].

Method	**MTAT**		**MSD**		**MTG-Jamendo**	
	ROC-AUC	**PR-AUC**	**ROC-AUC**	**PR-AUC**	**ROC-AUC**	**PR-AUC**
FCN (Choi et al.)	0.9005	0.4295	0.8744	0.2970	0.8255	0.2801
FCN (w/ 128 Mel bins)	0.8994	0.4236	0.8742	0.2963	0.8245	0.2792
Musicnn (Pons et al.)	0.9106	0.4493	0.8803	0.2983	0.8226	0.2713
Musicnn (w/ 128 Mel bins)	0.9092	0.4546	0.8788	0.3036	0.8275	0.2810
Sample-level (Lee et al.)	0.9058	0.4422	0.8789	0.2959	0.8208	0.2742
Sample-level+SE (Kim et al.)	0.9103	0.4520	0.8838	0.3109	0.8233	0.2784
CRNN (Choi et al.)	0.8722	0.3625	0.8499	0.2469	0.7978	0.2358
CRNN (w/ 128 Mel bins)	0.8703	0.3601	0.8460	0.2330	0.7984	0.2378
Self-attention (Won et al.)	0.9077	0.4445	0.8810	0.3103	0.8261	0.2883
Harmonic CNN (Won et al.)	0.9127	0.4611	0.8898	0.3298	0.8322	0.2956
Short-chunk CNN	0.9126	0.4590	0.8883	0.3251	0.8324	0.2976
Short-chunk CNN + Res	0.9129	0.4614	0.8898	0.3280	0.8316	0.2951

Table 2.1. State of the art in CNN music genre classification [Won+20]

3. Proposed method

3.1. Data analysis

3.1.1. Labels analysis

Figure 3.1.1 shows the tags distribution of all the 25863 instances. Also, this dataset defines a multi-class multi-label classification problem, as *Figure 3.1.2* suggests. A common way to prepare the dataset is to select the top 50 most popular classes [AK21]. *Figure 3.1.3* displays the correlation matrix between the first 50 most numerous tags, which reveals high correlations between some classes which induce the idea of labelling redundancies. *Table 3.1.1* summarizes the pairs of genres with the highest Pearson correlation index. It can be observed that while certain tag pairs are explainable by

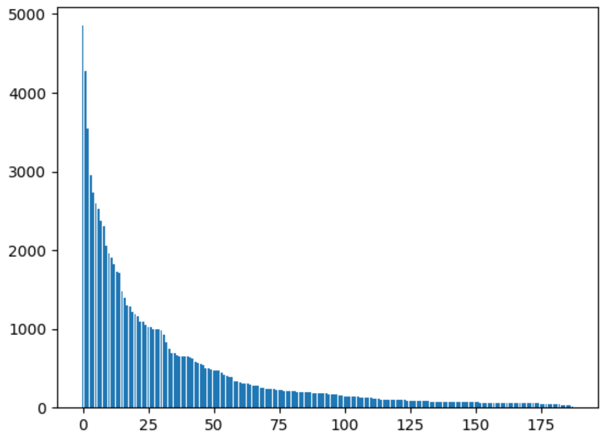

Fig.3.1.1. MTAT labels distributions

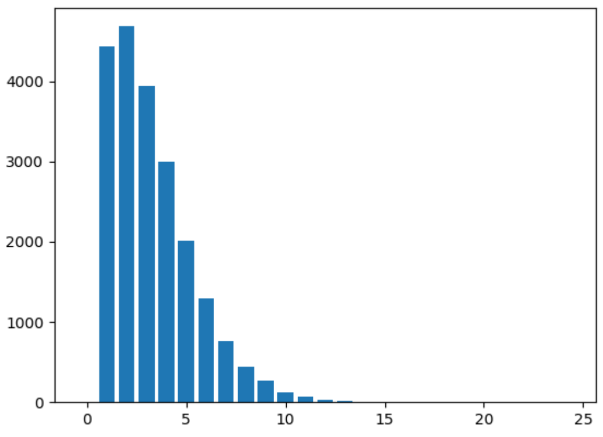

Fig.3.1.2. MTAT instances count per number of tags

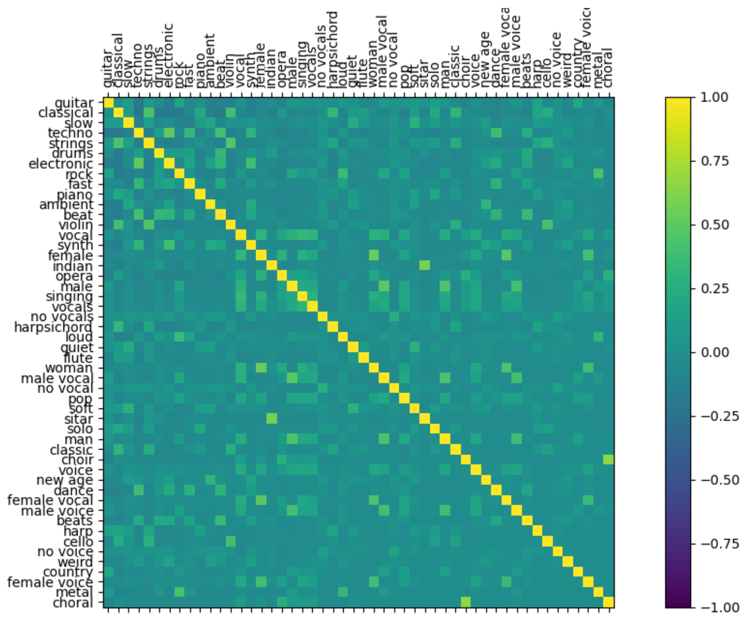

Fig.3.1.3. MTAT top-50 genre correlation matrix

Genre 1	**Genre 2**	**Corr.**
choir	choral	0.6573
indian	sitar	0.5910
female	woman	0.5402
electronic	techno	0.5107
female	female vocal	0.4896
male	male vocal	0.4886
strings	violin	0.4535
male	man	0.4403
woman	female vocal	0.4349
metal	rock	0.4263
classical	strings	0.4247
man	male vocal	0.4246
cello	violin	0.4071

Table 3.1.1. Top highly correlated tags

real facts (e.g. sitar is an Indian instrument, or a cello, also called violoncello, is a type of violin and is also associated with classical music, or that techno and electronic, or rock and metal are closely related terms that may coexist, but are not completely interchangeable), other labels may be purely redundant (female/woman/female vocal and male/man/male vocal, or the choir is the one that sings choral music). Won et al. holds an insightful discussion on music tagging and multi-instance problem, stating that, aside from current observations, certain tags may be valid for limited parts of the audio instead of the entire song, therefore tags can be predicted for both the whole instance or for short chunk splits followed by a global aggregation [Won+20].

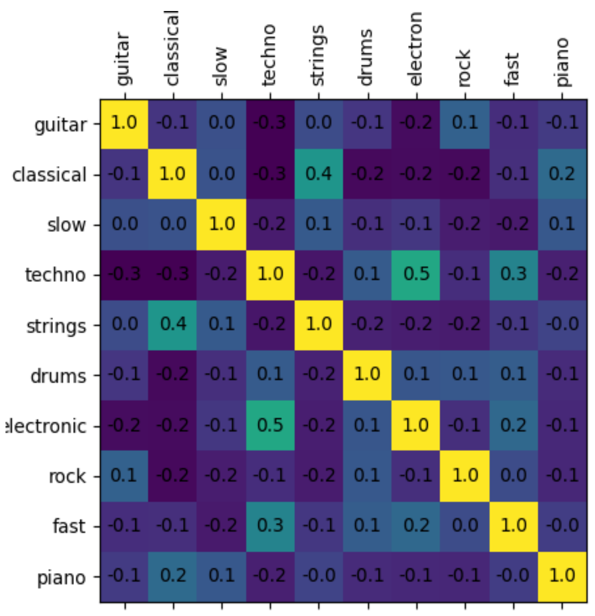

Fig.3.1.4. MTAT top-10 genre correlation

Instances count	Genre
4852	guitar
4274	classical
3547	slow
2954	techno
2729	strings
2598	drums
2519	electronic
2371	rock
2306	fast
2056	piano

Table 3.1.2. MTAT top-10

In order to avoid inconsistencies as much as possible while also keeping reasonable preprocessing effort requirements, the number of labels used in the current work was chosen to be 10. This decision is also encouraged by the fact that this is the maximum number of tags for which each tag will have at least 2000 associated song instances, thus minimizing the difference between the most and least numerous tag while also having sufficient data to work with for a proper training. The correlation matrix between the top 10 tags is depicted in *Figure 3.1.4*, while *Table 2* shows the number of instances for each genre. High correlation values are still visible, like electronic-techno or strings-classical, but they are clearly separable concepts as opposed to what was found in the top-50 correlations analysis.

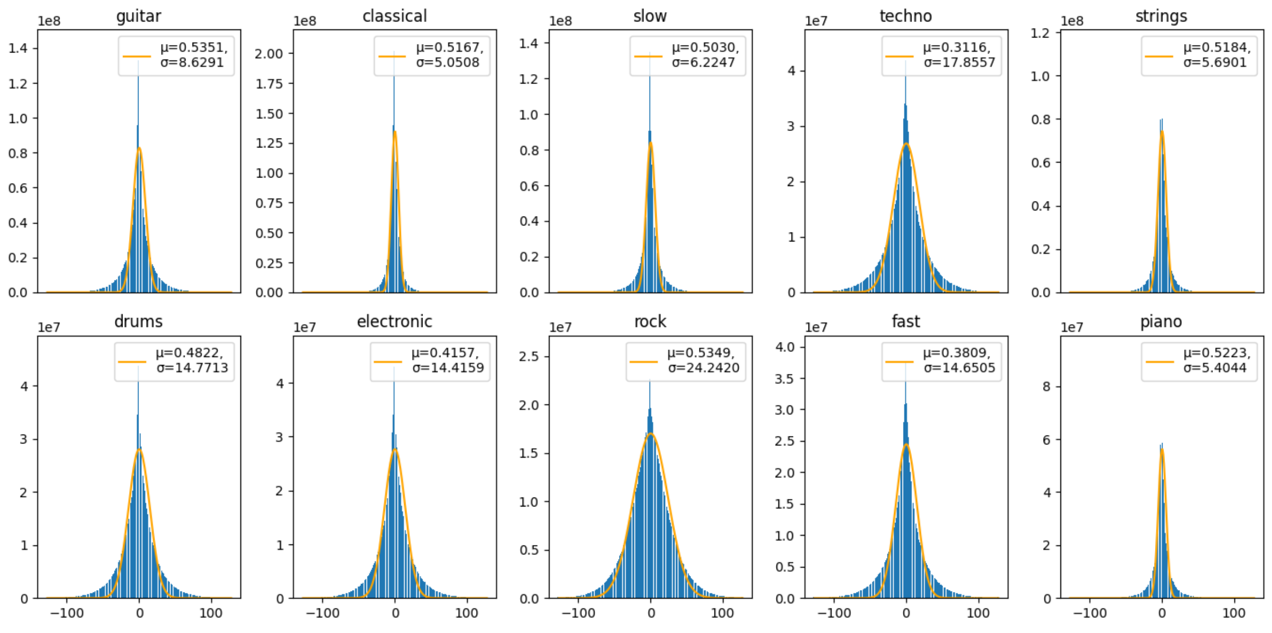

Fig.3.1.5. Samples histogram per genre

3.1.2. Audio samples and spectrogram statistics

One first point of interest would be the sound wave itself. In the following, the sound data, which is originally in MP3 format, has been brought to raw signed 8-bit PCM. That means, each discrete observation of the sound wave would be an integer in the interval $[-128, 127)$, since the ease to quickly manipulate the data is preferable to keeping high audio quality, especially because a DFT would approximate frequencies anyway, therefore it is more meaningful to know whether a sample is low or high rather than its actual high precision value.

Studying the occurrence of wave discrete samples for each genre reveals a tendency to follow a normal distribution which is showcased in *Figure 3.1.5*. While the mean is naturally centered around 0 (because positive samples should be counterweighted by negative signals in order to create the air vibration by physics related reasons), the variance provides meaningful data on the magnitude of sample values each genre employs. As shown in *Figure 3.1.6*, synthesized music (electronic, techno, rock), percussion and fast paced music has higher sample variance, as opposed to slower, harmonious melody like classical music played with traditional instruments (piano, strings, guitar), which feature a lower variance.

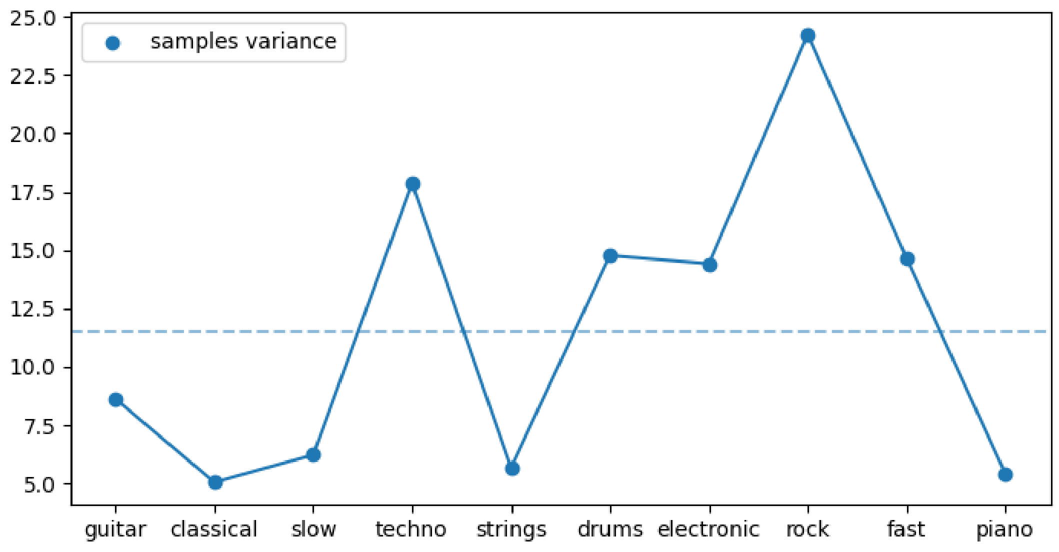

Fig.3.1.6. Sample variance per genre

Additionally, by averaging the spectrogram heat maps across genres and summing their intensities over time, similar remarks can be made about the frequency domain. According to *Figure 3.1.8*, synthesized and fast paced music tends to have a higher frequencies density and range compared to the slow, classical, naturally generated sound. It comes with no surprise, since the effects captured by the spectrograms are easily perceived by the human ear such as calm, noise or rhythm (*Fig 3.1.7*).

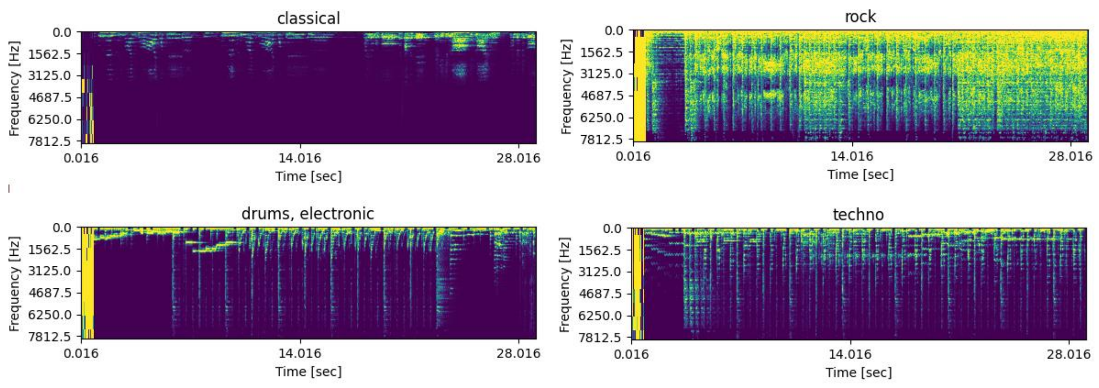

Fig.3.1.7. Spectrogram examples of various genres

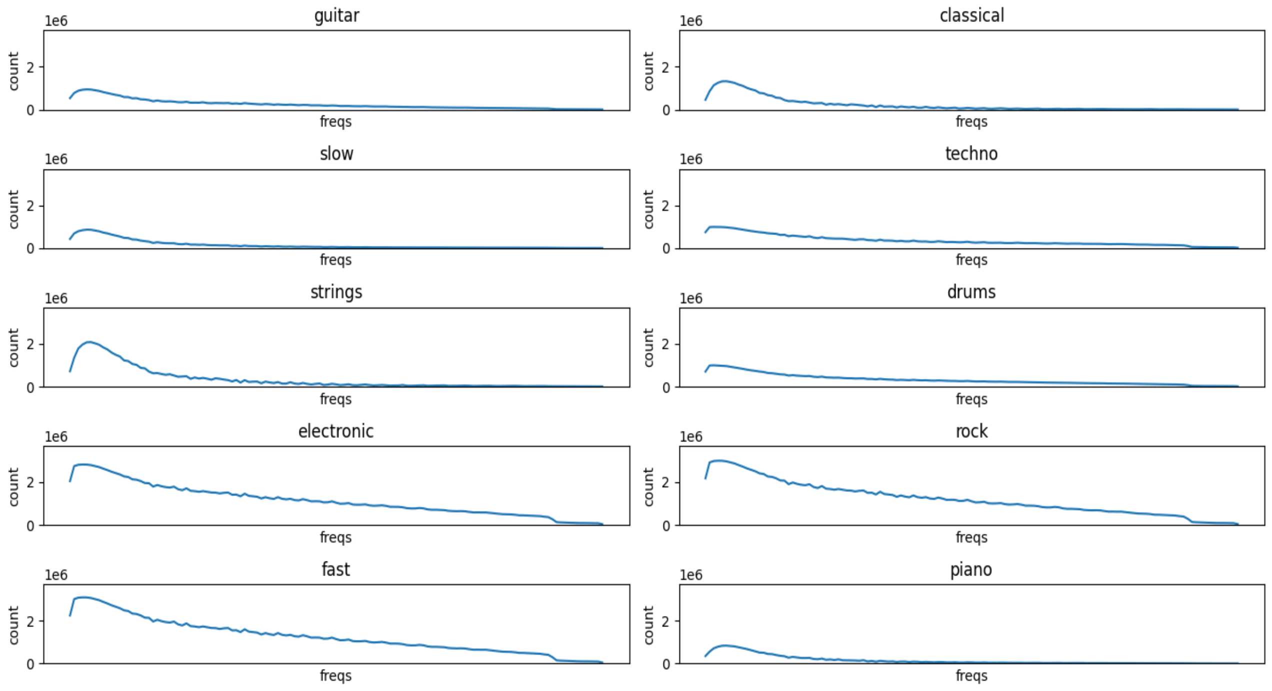

Fig.3.1.8. Time-average frequency distribution per genre

3.1.3. Audio data visualization

Dimensionality reduction techniques applied on raw PCM data (*Figure 3.1.9*) do not reveal any useful information. The messy output, sometimes presenting clear outliers (like in PCA), can be explained by the complex relationship that characterizes the data, in the sense that samples are not features in themselves, but a result of other phenomena possibly visible in other spaces. At the same time, dimensionality reduction applied on the spectrograms, while not defining clean clusters separation,

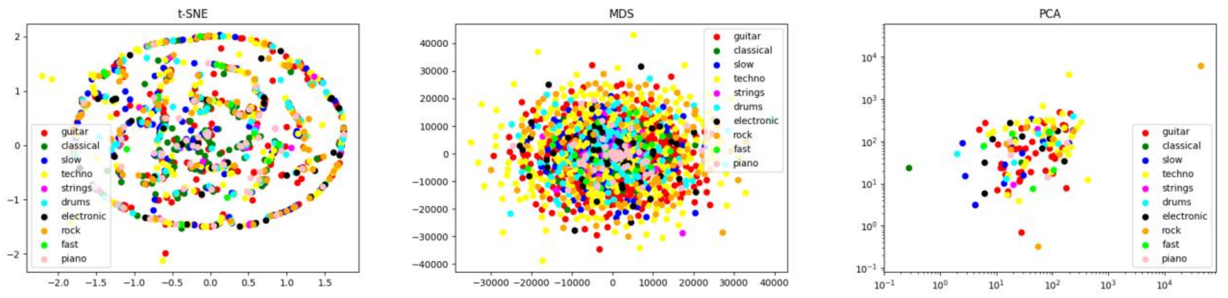

Fig.3.1.9. Sample wave projections

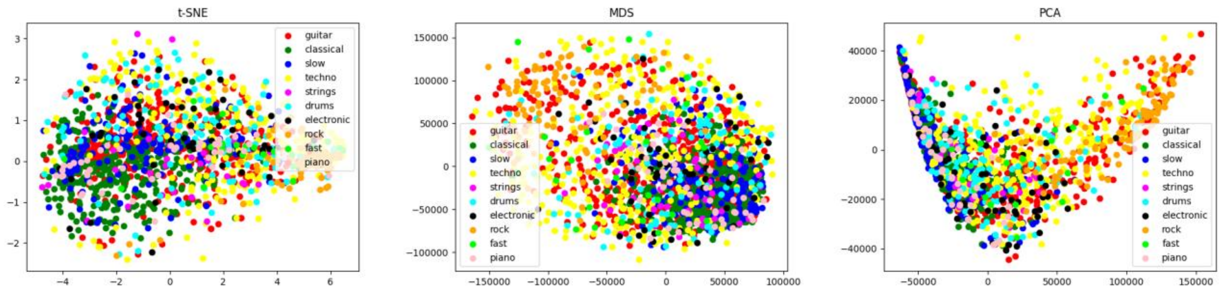

Fig.3.1.10. Spectrogram projections

are able to differentiate between the two categories of genres mentioned in the previous subsection. For instance, in *Figure 3.1.10*, PCA and MDS could easily tell apart fast and slow music, depicted in blue and orange colors.

3.1.4. Extracted features analysis

The MTAT dataset comes with per-clip automatically pre-extracted audio features using the Echo Nest API 1.0. Such features include track-level descriptors (loudness, tempo, key, mode), bars and beats, and per-segment descriptors (loudness, 12 classes of pitch, 12 timbre dimensions). This kind of data is easier to use than the raw audio data because of its smaller dimensions and summarizing nature. For the purpose of this analysis, the track features were selected along with averaged values of beat and each of the 12 classes of pitch and timbre. A correlation matrix (*Figure 3.1.11*) reveals strong correlation (0.94) between loudness and one timbre feature (timbre_0), and high inverse correlation between timbre_4 and loudness.

Additionally, by performing the feature importance using least squares linear regression (*Figure 3.12*), it can be observed that the timbre vector has little to no impact over genre classification, while certain pitch classes may have a strong decision factor favoring a specific genre (e.g. pitch_0 for techno, pitch_4 for rock, or pitch_9 for classical music).

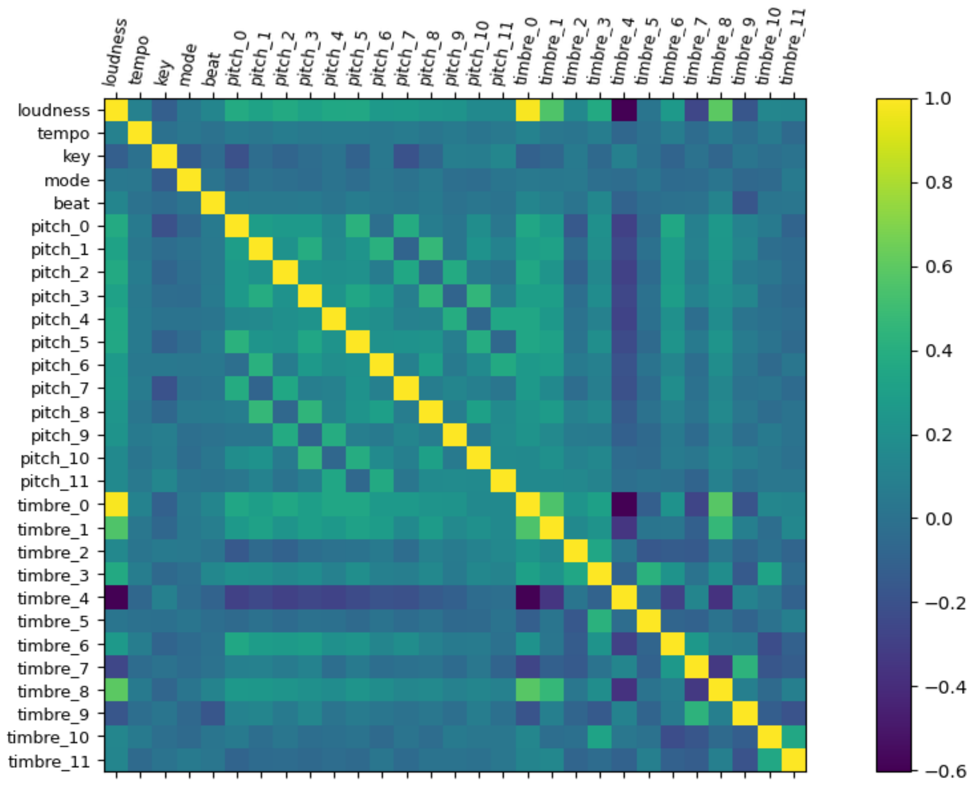

Fig.3.1.11. Audio features correlation matrix

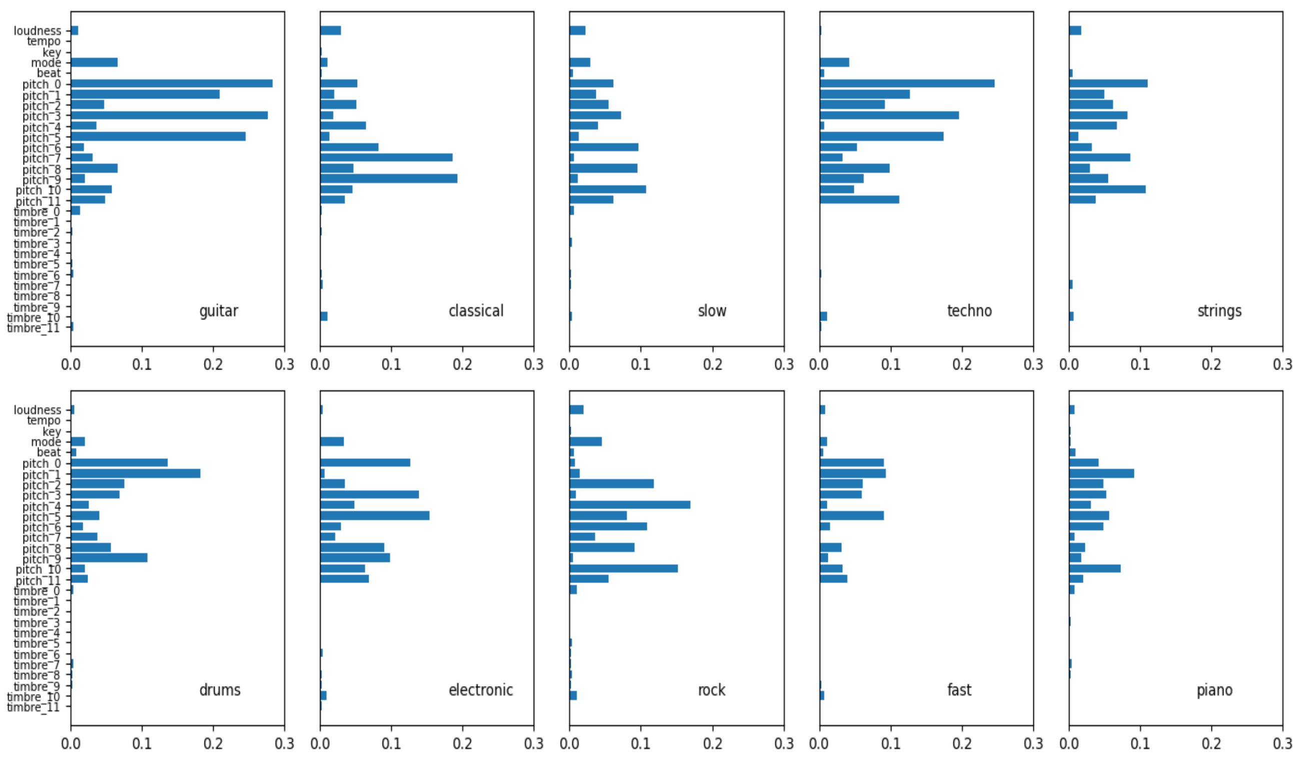

Fig.3.1.12. Audio features importance per genre

3.2. Experimental setup

Given the fact that the task is a multi-class multi-label problem with weak connections between the classes, it is a natural idea to split it into 10 per-tag binary classification sub-problems. This work evaluates the performance of 8 neural network models on the audio samples and PSD spectrograms to study the impact of both one and two dimensional representations of the raw data. An experiment workflow summary is illustrated in *Figure 3.2.1*.

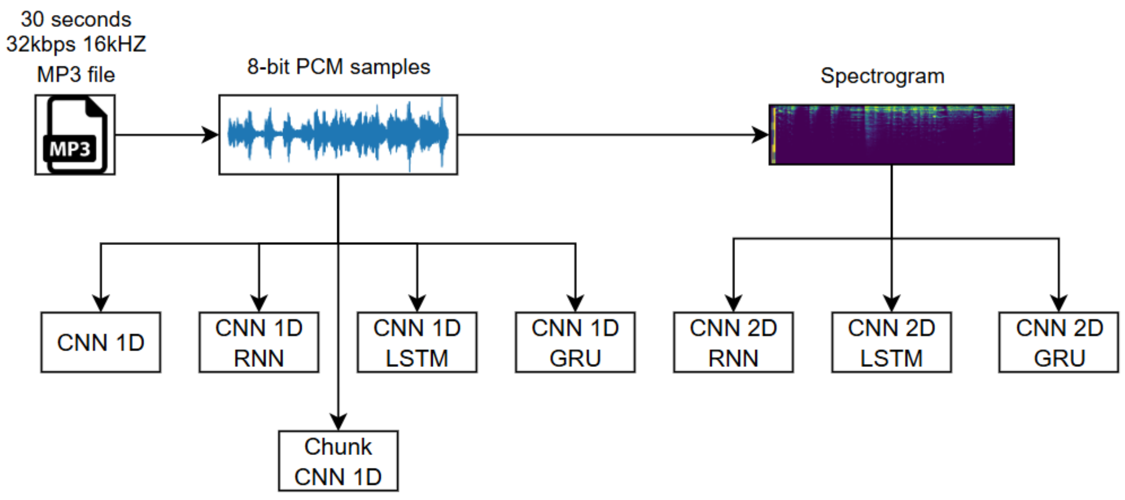

Fig.3.2.1. Experiment workflow

3.2.1. Data preparation and preprocessing

The original 16 kHz MP3 instances are converted to raw PCM-8 audio, represented as a 1-D vector of integer samples between $[-128, 127)$ which will be rescaled to $[-1,1)$ before passing to the model. The same integer vector is used to generate a spectrogram using scipy's default configurations: 0.25 Tukey window, 256 segment length, density scaling and PSD mode. The spectrogram will be treated as a grayscale image during the training process. It was observed that spectrogram intensities can reach very high values (see *Figure 3.1.8*), therefore, for ensuring training stability and for the ease of visualization, the spectrogram matrix was clamped to a maximum value of 512 and normalized into the [0,1] interval. The shapes summary are presented in *Table 3.2.1*. The MTAT dataset was filtered to include the clips that belong to at least one of the 10 selected tags, leaving 21108 out of 25863 original instances. A train/validation split was randomly created with ratio 0.9, conducting to 19473 train and 1635 validation instances.

Data	**Shape**
Input 1D samples	465984x1
Input 2D spectrogram	129x2080x1
Output logits	10

Table.3.2.1. Input and output shapes

3.2.2. Network architectures

Four main classes of neural networks are considered: sample-based 1D CNN (a), 1D CRNN (b) and chunk max pooling 1D CNN (c), and spectrogram based 2D CRNN (d) with the recurrent network acting along the temporal axis. The CRNN models have versions for each of the popular neural network architectures: Simple RNN, Long-Short Term Memory (LSTM) and Gated Recurrent Unit (GRU).

Model (a) architecture: Sample-based 1D CNN	
Layer	**Output shape**
Input	465984x1
Conv1D+MaxPooling1D+Dropout	155328x32
Conv1D+MaxPooling1D+Dropout	51776x64
Conv1D+MaxPooling1D+Dropout	17258x128
Conv1D+MaxPooling1D	5752x128
Conv1D+MaxPooling1D+Dropout	1917x256
Conv1D+MaxPooling1D+Dropout	639x256
Conv1D+MaxPooling1D+Dropout	213x512
Conv1D+BatchNorm.+MaxPooling1D+Dropout	71x512
Conv1D+BatchNorm.+MaxPooling1D+Dropout	23x1024
Conv1D+BatchNorm.+Dropout	23x1024
Flatten	23552
Dense+LeakyReLU	100
Dropout (0.1)	100
Dense+Sigmoid	10

Table.3.2.2. 1D CNN (a) model architecture
All 1D convolutions have 3 sized kernel, same padding and Leaky ReLU activations.
All Max Pooling 1D layers have pool size 3.
Dropout rate is 0.2 unless otherwise specified.

Model (b) architecture: 1D CRNN	
Layer	**Output shape**
Input	465984x1

Conv1D+MaxPooling1D+Dropout	155328x32
Conv1D+MaxPooling1D+Dropout	51776x64
Conv1D+MaxPooling1D+Dropout	17258x128
Conv1D+MaxPooling1D	5752x128
Conv1D+MaxPooling1D+Dropout	1917x256
Conv1D+MaxPooling1D+Dropout	639x256
Conv1D+MaxPooling1D+Dropout	213x512
RNN/LSTM/GRU	256
Dense+LeakyReLU	100
Dropout(0.1)	100
Dense+Sigmoid	10

Table.3.2.3. 1D CRNN (a) model architecture
All 1D convolutions have 3 sized kernel, same padding and LeakyReLU activations.
All MaxPooling1D layers have pool size 3.
Dropout rate is 0.2 unless otherwise specified.
The recurrent network has 256 cells and the default tensorflow configuration.

Model (c) architecture: Chunk 1D CNN	
Layer	**Output shape**
Input	465984x1
Conv1D+MaxPooling1D+Dropout	155328x16
Conv1D+MaxPooling1D+Dropout	51776x16
Conv1D+MaxPooling1D+Dropout	17258x32
Conv1D+MaxPooling1D+Dropout	5752x64
Conv1D+MaxPooling1D+Dropout	1917x128
Conv1D+BatchNorm.+MaxPooling1D+Dropout	639x128
Conv1D+BatchNorm.	639x128
GlobalMaxPooling1D	128
Dropout (0.1)	128
Dense+Sigmoid	10

Table.3.2.4. Chunk 1D CNN (c) model architecture
All 1D convolutions have 3 sized kernel, same padding and Leaky ReLU activations.
All MaxPooling1D layers have pool size 3.
Dropout rate is 0.2 unless otherwise specified.

The sample-based 1D CNN (model (a) – *Table 3.2.2*) is a classical deep convolutional features extractor followed by fully two connected layers for classification. A pool size of 3 was used for quick squeezing of the time dimension while the number of convolutional filters slowly increases with the depth. Dropout layers (just like in the other models) and batch normalization are used for regularization purposes.

The 1D CRNN (model (b) – *Table 3.2.3*) takes its base from model (a) and replaces the last three convolution blocks with one of the three recurrent layers whose output is the last processed state of the sequence. Alternatively, returning the entire sequence from the recurrent layer is a perfectly valid method and may help identify genre patterns in every small window frame, consequently combining these results into a full clip prediction. The reasoning behind choosing the last state approach is that, unlike in speech recognition, there is no real need for per-frame observations, and that errors appearing in classifying the individual frames may affect the final prediction in a destructive way. Recurrent networks should be able to pass dominant tag activations from one state to another, meaning that the state of the last cell should be enough for a relevant classification. Moreover, there is already an aggregation model discussed in this work (model (c)), although not recurrent, which should provide an insight into the effectiveness of this solution. This, however, does not substitute a thorough comparison of using the sequence or the last state of the recurrent output.

Model (d) architecture: Spectrogram 2D CRNN	
Layer	**Output shape**
Input	129x2080x1
Conv2D+MaxPooling2D+Dropout	64x1040x16
Conv2D+MaxPooling2D+Dropout	32x520x16
Conv2D+MaxPooling2D+Dropout	16x260x32
Conv2D+MaxPooling2D+Dropout	8x130x64
Conv2D+MaxPooling2D+Dropout	4x130x128
Conv2D+BatchNorm.+MaxPooling2D+Dropout	2x130x128
Conv2D+BatchNorm.+MaxPooling2D+Dropout	1x130x128
Reshape	130x128
RNN/LSTM/GRU	256
Dense+Sigmoid	10

Table.3.2.5. Spectrogram 2D CRNN (d) model architecture
All 1D convolutions have 3x3 sized kernel, same padding and Leaky ReLU activations.
All Max Pooling 2D layers have pool size 2x2, except for the last one, which is of size 1x2.
Dropout rate is 0.2.

The chunk 1D CNN (model (c) – *Table 3.2.4*), resembling the short chunk CNN from the literature [Won+20], presents an usual feature extractor followed by a global max pooling layer which reduces the time dimension, keeping the mostly activated features. The classification result is obtained by passing the pooling result to a sigmoid activated fully connected layer.

Finally, the spectrogram 2D CRNN (model (d) – *Table 3.2.5*) perform deep CNN features extraction on the spectrogram activations before feeding the feature maps into a recurrent network with the same properties as model (b).

3.2.3. Training setup

For a better comparison, all models use the same training configuration. The models are trained for 10 epochs each with a batch size of 16. The Adam optimizer with a learning rate of 10^{-3} is used to minimize the binary cross-entropy loss. A binary accuracy metric is used to observe the per-epoch model's behavior at train and validation time.

4. Experimental Results

4.1. Training evaluation and comparison

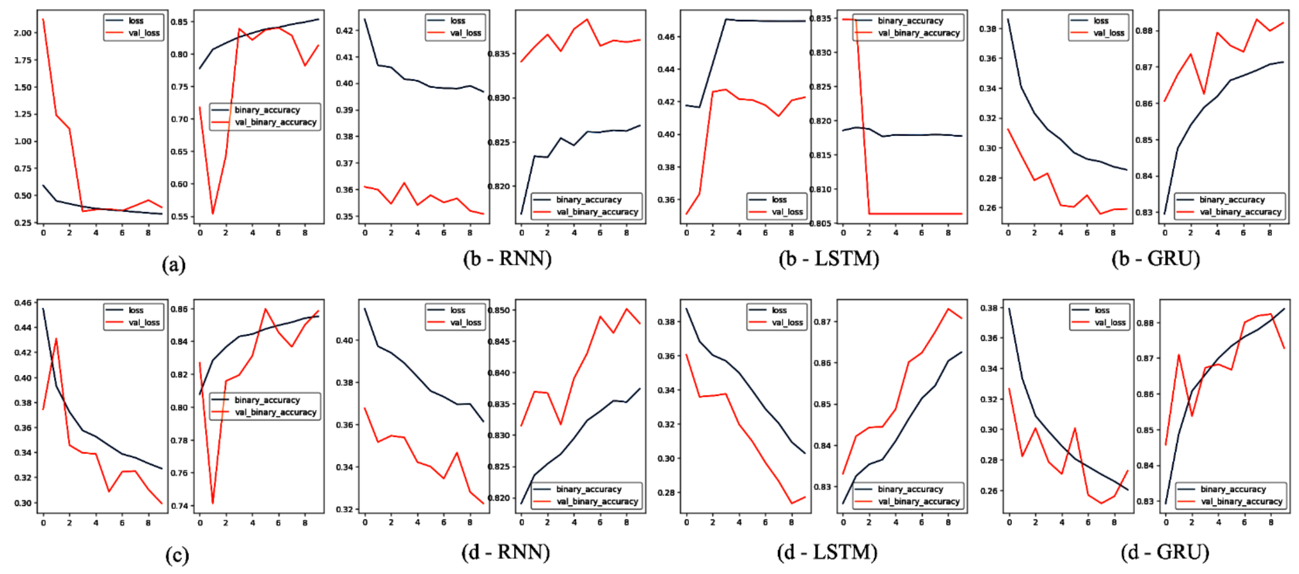

Figure 4.1. Training process evolution of loss and accuracy for train and validation sets

The training plots of loss and accuracy per epoch are shown in *Figure 4.1*. The loss and accuracy at the end of the training can be found in *Table 4.1*. It can be noted that the CNN-GRU models based on (b) and (d) have the best convergence, while the worst performing model is 1D CRNN model (b) with LSTM. The dropout has helped in keeping the train and validation measures close to each other, thus successfully reducing overfitting. Most models (b-GRU, c, d) show an accentuate tendency in loss decreasing rate, which means that reaching better results as the epochs increase is a plausible statement.

Model	**Train loss**	**Val loss**	**Train Acc.**	**Val acc.**
(a)	0.3315	0.3888	0.8514	0.8126
(b–RNN)	0.3978	0.3507	0.8262	0.8365
(b–LSTM)	0.4728	0.4226	0.8175	0.8063
(b–GRU)	0.2867	**0.2590**	0.8705	**0.8821**
(c)	0.3311	0.2991	0.8528	0.8584
(d–RNN)	0.3638	0.3227	0.8350	0.8478
(d–LSTM)	0.3062	0.2772	0.8602	0.8708
(d–GRU)	**0.2649**	0.2729	**0.8813**	0.8728

Table 4.1. Train and validation metrics for each model at the end of last epoch

4.2. Quantitative and qualitative performance analysis

The per-tag AUC-ROC and AUC-PR are shown in *Figures 4.2* and *4.3*, while numerical values of the area under curves are extracted in *Table 4.3* along with the classification metrics values at threshold 0.5. It is easy to observe that b-GRU and d-GRU models have the best overall performance, even surpassing some of the state of the art models shown in *Table 2.1*. The failure of b-LSTM manifests through the model only predicting zero labels (maximum recall) and may be caused by the inability of the 1D CNN to extract meaningful features that serve the timewise pass of LSTM. This particular experiment was repeated multiple times with the same results. It indicates that Conv1D and LSTM is only a bad combination for this dataset, considering that LSTM works well on spectrogram derived maps and the 1D convolution leads to acceptable results both on itself and paired with RNN or GRU.

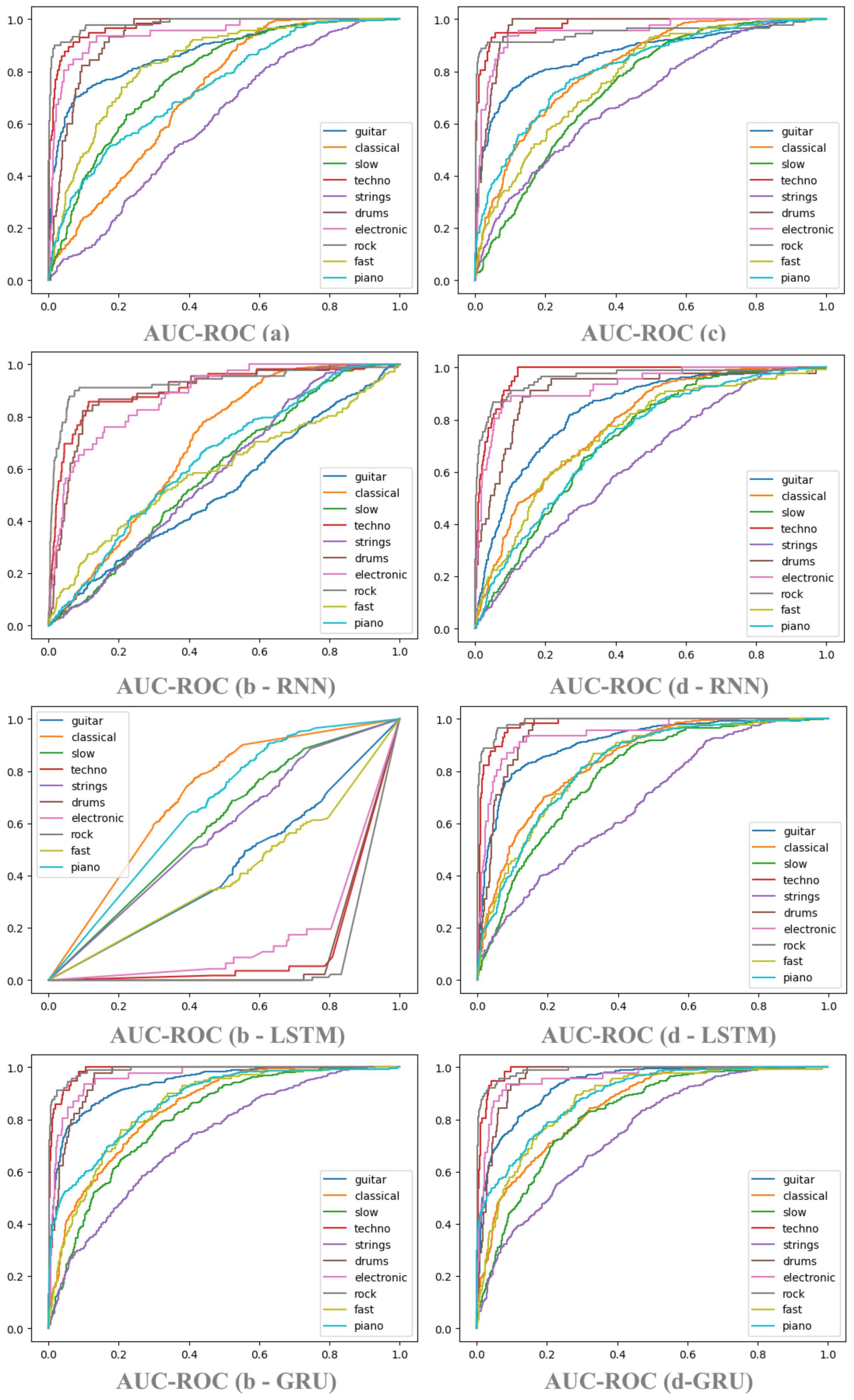

Figure 4.2. Per-tag area under receiver operating curve for each model

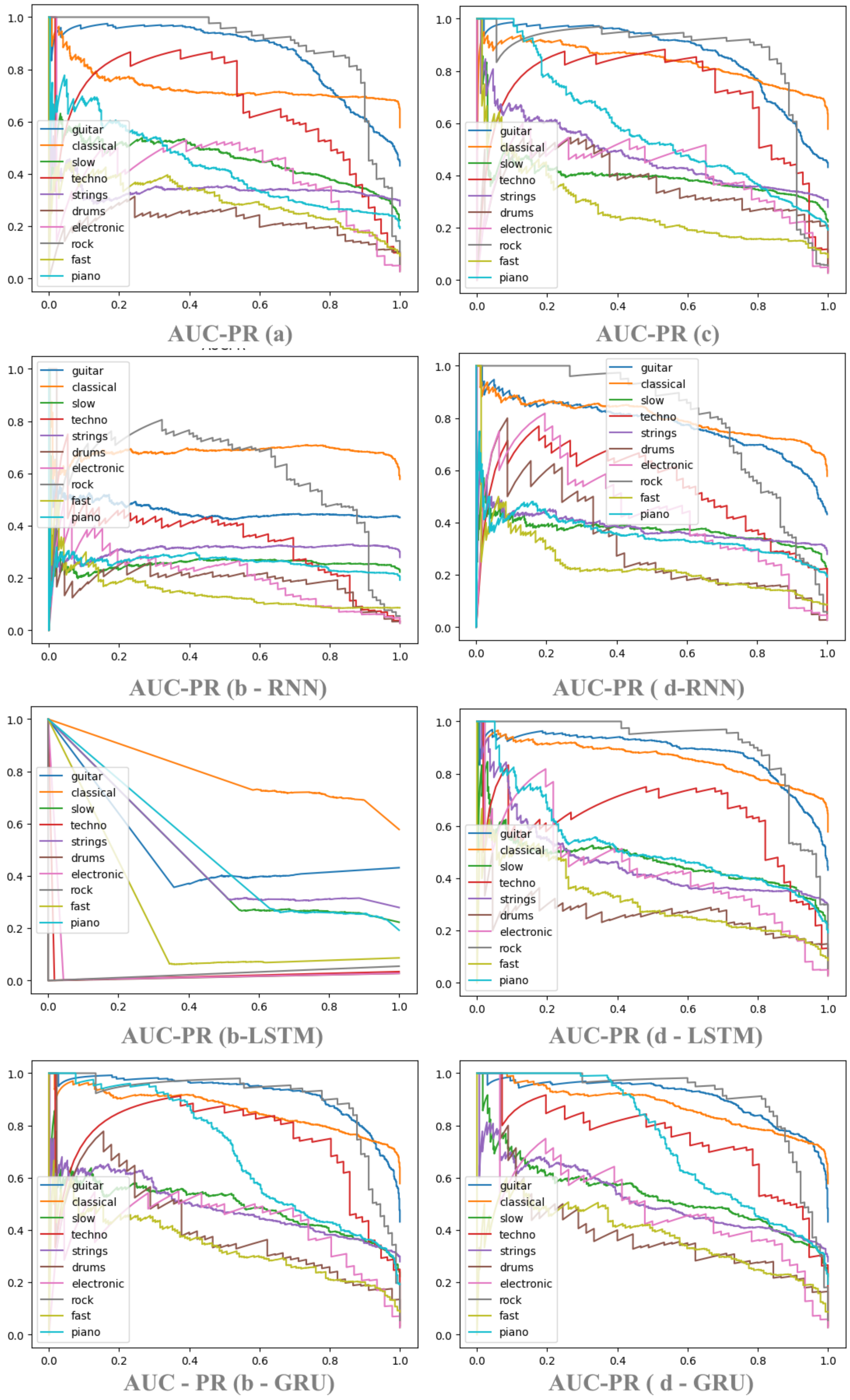

Figure 4.3. Per-tag area under precision-recall curve for each model

Tag	a	b-RNN	b-LSTM	b-GRU	c	d-RNN	d-LSTM	d-GRU
Accuracy								
guitar	0.7755	0.5694	0.5681	**0.8685**	0.7376	0.7363	0.8470	0.8110
classical	0.4611	0.7070	0.4220	0.7700	0.7412	0.6856	0.7633	**0.7706**
slow	0.5932	0.7767	0.7767	0.7914	0.7474	0.7761	0.7785	**0.7975**
techno	0.9706	0.9651	0.9657	**0.9834**	0.9779	0.9706	0.9785	0.9755
strings	0.7198	0.7204	0.7204	0.7400	0.7204	0.6691	0.6776	**0.7522**
drums	0.9645	0.9724	0.9724	0.9718	0.9718	**0.9743**	0.9645	0.9602
electronic	0.9730	0.9718	0.9718	0.9724	0.9706	0.9718	**0.9749**	0.9681
rock	0.9798	0.9614	0.9449	**0.9853**	0.9785	0.9737	0.9834	0.9816
fast	0.9082	0.9131	0.9131	**0.9137**	0.9137	0.9131	0.9119	0.9125
piano	0.7798	0.8073	0.8073	0.8238	**0.8244**	0.8073	0.8275	0.7981
Avg.	0.8125	0.8364	0.8062	**0.8820**	0.8583	0.8477	0.8707	0.8727
Precision								
guitar	0.8393	0.5714	0.5681	0.8466	**0.8654**	0.7085	0.8273	0.7637
classical	0.4381	**0.8781**	0.4220	0.8651	0.7405	0.6108	0.8136	0.7569
slow	**0.9481**	0.7767	0.7767	0.8132	0.8134	0.7776	0.7823	0.8360
techno	0.9916	0.9697	0.9657	0.9874	0.9800	0.9848	0.9898	**0.9929**
strings	0.7205	0.7204	0.7204	0.7698	0.7775	0.7591	0.7770	**0.7778**
drums	0.9645	0.9724	0.9724	0.9806	0.9765	0.9748	0.9781	**0.9841**
electronic	0.9867	0.9718	0.9718	0.9806	0.9724	0.9718	0.9777	**0.9891**
rock	0.9846	0.9707	0.9449	**0.9922**	0.9928	0.9807	0.9890	0.9915
fast	0.9199	0.9131	0.9131	0.9167	0.9157	0.9141	0.9176	**0.9343**
piano	0.8647	0.8073	0.8073	0.9056	0.8718	0.8073	0.8365	**0.9372**
Avg.	0.8658	0.8551	0.8062	**0.9057**	0.8906	0.8489	0.8888	0.8963
Recall								
guitar	0.7481	0.9687	**1.0000**	0.9386	0.6372	0.9106	0.9235	0.9666
classical	0.9797	0.3550	**1.0000**	0.5391	0.5956	0.7028	0.5695	0.6724
slow	0.5039	**1.0000**	**1.0000**	0.9496	0.8755	0.9968	0.9905	0.9196
techno	0.9778	0.9949	**1.0000**	0.9955	0.9974	0.9848	0.9879	0.9816
strings	0.9983	**1.0000**	**1.0000**	0.9117	0.8573	0.7920	0.7750	0.9185
drums	0.9849	**1.0000**	**1.0000**	0.9905	0.9949	0.9993	0.9855	0.9748
electronic	0.9855	**1.0000**	**1.0000**	0.9911	0.9981	**1.0000**	0.9968	0.9779
rock	0.9941	0.9889	**1.0000**	0.9922	0.9844	0.9915	0.9935	0.9889
fast	0.9852	**1.0000**	**1.0000**	0.9959	0.9973	0.9986	0.9926	0.9725
piano	0.8621	**1.0000**	**1.0000**	0.8727	0.9174	**1.0000**	0.9772	0.8037
Avg.	0.9019	0.9307	**1.0000**	0.9176	0.8855	0.9376	0.9192	0.9176
AUC-ROC								
guitar	0.8764	0.5223	0.4325	**0.9386**	0.8706	0.8385	0.9109	0.9344
classical	0.7161	0.6883	0.7055	0.8441	0.8189	0.7832	0.8415	**0.8504**
slow	0.7830	0.5859	0.5921	0.8042	0.7400	0.7256	0.7865	**0.8203**
techno	0.9732	0.9061	0.1219	**0.9874**	0.9793	0.9724	0.9800	**0.9874**
strings	0.6101	0.5835	0.5693	0.7256	0.7004	0.6401	0.6806	**0.7501**
drums	0.9352	0.8956	0.1103	0.9599	0.9677	0.9110	0.9490	**0.9627**
electronic	0.9458	0.8793	0.1694	**0.9627**	0.9526	0.9325	0.9434	0.9596
rock	0.9844	0.9311	0.0863	**0.9897**	0.9503	0.9619	0.9905	0.9887
fast	0.8349	0.5919	0.4075	0.8494	0.7791	0.7524	0.8281	**0.8722**
piano	0.7394	0.6380	0.6602	**0.9709**	0.8140	0.7265	0.8248	0.8908
Avg.	0.8399	0.7222	0.3855	**0.9032**	0.8572	0.8244	0.8735	0.9016
AUC-PR								
guitar	0.8617	0.4577	0.5023	0.9215	0.8645	0.7767	0.8792	**0.9076**
classical	0.7359	0.6833	0.7935	0.8618	0.8353	0.8031	0.8577	**0.8749**
slow	0.4545	0.2573	0.4643	0.4919	0.3996	0.3691	0.4663	**0.5381**
techno	0.6545	0.3390	0.0264	0.7406	0.6986	0.5284	0.6240	**0.7423**
strings	0.3396	0.3100	0.4867	0.4890	0.4802	0.3814	0.4643	**0.5238**
drums	0.2097	0.2024	0.0138	**0.4096**	0.3683	0.3142	0.2700	0.4017
electronic	0.399	0.2065	0.0367	0.4085	0.4056	0.4356	0.4358	**0.5013**
rock	0.8918	0.5948	0.0276	0.8989	0.8498	0.7948	**0.9077**	0.9042
fast	0.2962	0.1464	0.2323	0.3356	0.2769	0.2352	0.3143	**0.3729**
piano	0.4275	0.2591	0.4963	0.6959	0.5769	0.3522	0.5263	**0.7413**
Avg.	0.5271	0.3456	0.3079	0.6253	0.5755	0.4990	0.5745	**0.6508**

Table 4.2. Classification metrics for each model and class

The ROC curves reveal that some genres are difficult to pinpoint (strings) and this is not related to the lack of dataset instances for that label (for example, piano, the least numerous label, scores well in most evaluations compared to the slow tag, the third most common label). Additionally, the PR curve can be used to fine-tune the classification threshold for each tag if a better precision-recall ratio is desired. A wise choice indicator would be the right-most points of discontinuity which cause the pointy wave appearance of the graphs.

5. Conclusion and future work

This work demonstrates that CNN is a simple reliable tool in music classification genre and work surprisingly well on both raw 1D data and 2D frequency intensity view of the spectrogram. The combination of CNN with other machine learning algorithms like recurrent neural networks could also be beneficial. However, while the results are good, they are not perfect, and there surely is room to explore and improved the results by tweaking the training setup and trying other architectures. It is also important to mention that, due to the author's first contact with the sound processing domain and lack of familiarity with spectrograms, the scipy library's defaults were used to generate the spectrogram. The established literature methods use Mel spectrograms, which are said to better capture the human impression properties of audio [Cho+17]. It remain a future task to find out whether the findings of this study also hold true for Mel spectrograms.

6. References

[AK21] S. Allamy and A. L. Koerich. "1D CNN Architectures for Music Genre Classification". In: *2021 IEEE Symposium Series on Computational Intelligence (SSCI)* (2021), pp. 01–07. url: https://api.semanticscholar.org/CorpusID:234742681.

[Cho+17] K. Choi et al. "Transfer learning for music classification and regression tasks". In: *18th International Society for Music Information Retrieval Conference, ISMIR 2017*. International Society for Music Information Retrieval. 2017, pp. 141–149.

[Déf+22] A. Défossez et al. "High Fidelity Neural Audio Compression". In: *Transactions on Machine Learning Research* (2022).

[Gho+23] P. Ghosh et al. "A Study on Music Genre Classification using Machine Learning". In: *International Journal of Engineering Business and Social Science* 1.04 (2023), pp. 308–320.

[KKB21] D. Kostrzewa, P. Kaminski, and R. Brzeski. "Music genre classification: looking for the perfect network". In: *International Conference on Computational Science. Springer*. 2021, pp. 55–67.

[Law+09] E. Law et al. "Evaluation of algorithms using games: The case of music tagging." In: *ISMIR*. Citeseer. 2009, pp. 387–392.

[MZ18] M. Matocha and S. Zieliński. "Music genre recognition using convolutional neural networks". In: *Advances in Computer Science Research* (2018).

[OS10] A. V. Oppenheim and R. W. Schafer. Discrete-time signal processing. Pearson, 2010.

[SSF13] J. Stastny, V. Skorpil, and J. Fejfar. "Audio data classification by means of new algorithms". In: *2013 36th International Conference on Telecommunications and Signal Processing (TSP)*. 2013, pp. 507–511. doi: 10.1109/TSP.2013.6613984.

[Won+20] M. Won et al. "Evaluation of CNN-based Automatic Music Tagging Models". In: *SMC* (2020).

Acknowledgement:

This work is the result of my own activity, and I confirm I have neither given, nor received unauthorized assistance for this work.
I declare that I did not use generative AI or automated tools in the creation of content or drafting of this document.