AUDIO FEATURES FOR MUSIC GENRE CLASSIFICATION

Nicklas Hansen (s153077), Bragi Marinosson (s185510)

Technical University of Denmark

ABSTRACT

Music genre classification is an inherently difficult task for machines as categorization is subjective and based on human perception. Typically, hand-engineered audio features are used in models that aim to automate genre annotation, but it is unclear which features best describe a genre and to which extent. In this study, we provide a holistic view on common audio features and apply a backward feature selection using a k-nearest neighbors (KNN) classifier [1] and features extracted from a slightly imbalanced 7-genre subset (6573 audio files) of the Free Music Archive (FMA) dataset [2]. We then train a 20-layer DenseNet [3] on the best-performing feature subset – 32 Mel Frequency Cepstral Coefficents (MFCCs) – and achieve a F_1-score of 0.60, which is comparable to other studies [4] but below human-level performance measured under similar conditions [5] [6]. It is observed that classification performance is highly genre-dependent, which suggests that certain genres may have similar profiles in the frequency domain.

1. INTRODUCTION

In recent years, there has been an abundance of freely available music on the World Wide Web and the volume steadily increases every year. This has lead to a strong demand for indexing and categorization of music to help navigate large collections of musical pieces, which is what researchers in the field of Music Information Retrieval (MIR) attempts to solve. One of the oldest and most widely used categorizations are *music genres* (*genres* from here on), which is an ever-evolving categorization based on human perception.

Due to the inherently subjective nature of genres, the problem of music genre classification is especially difficult. In fact, [5] found that the inter-agreement of non-expert music listeners was 76% when listening to $30s$ excepts and tasked to distinguish between 6 genres, while [6] found that the average human has a mere 55% accuracy when given 19 options of genres in a comparable setting. Additionally, [6] found that human classification performance does not increase significantly beyond an excerpt length of $3s$.

This study aims to provide a brief presentation of two classification techniques – k-Nearest Neighbors (KNN) and DenseNet [3] based on Convolutional Neural Networks (CNNs) – and their application to music genre classification, as well as a holistic view of the predictive ability of audio features commonly used for MIR by benchmark of a KNN classifier trained using a backward feature selection scheme. The features considered are Mel-Frequency Cepstral Coefficients (MFCCs), Chroma features, Crest factor as well as tempo (beats per minute, *bpm*) and audio duration.
Models are trained and evaluated on 3 s excerpts of music from a slightly unbalanced subset of the Free Music Archive (FMA) dataset [2] – summarized in Table 1 – and predictions are made on a total of 7 genres. Finally, opportunities for future work are briefly touched upon in section 7.
Implementation is available at:
https://github.com/nicklashansen/music-genre-classification.

Genre	Song count
Electronic	1000
Hip-Hop	1000
Rock	1000
Folk	1000
Pop	1000
Classical	1000
Jazz	573

Table 1: Genre distribution of a subset of the FMA dataset [2]. A total of 6573 audio files consisting of 7 genres are used in this study.

2. DATASET

The original FMA dataset [2] used in this study contains a total of $106,574$ raw MPEG-3 encoded audio tracks from a collection of 161 genres. As the genre distribution in FMA is highly imbalanced, we use a 7-genre subset where genres are selected based on frequency and popularity, which leaves us with 6573 audio files (see Table 1). 5916 files are used for training and the remaining 657 files are used for testing. Each audio file contains a $30s$ excerpt of the original song. It is possible to obtain audio files in their entirety but that has two implications: 1) it results in a class imbalance due to different audio durations, and 2) intro- and outro segments will be difficult to classify, artificially lowering the performance. As such, we have decided to only address the $30s$ excerpts in this study.

3. PRE-PROCESSING

Rather than processing raw audio, one typically extracts auditory features. While human-engineered feature extraction may introduce some bias, they have proven widely useful in past experiments. The most common technique is to compute the STFT of the audio to produce a *spectrogram*. The dimensionality of a spectrogram, however, can be very large, which has a significant impact on processing time. A natural way to transform the spectrogram into a low-dimensional representation is the use of MFCCs. The mel-spectrogram (sometimes denoted the mel-frequency cepstrum in litterature) is a quantized version of the spectrogram (using a number of frequency bands equal to the number of MFCCs), where frequency resolution is high in the low frequencies and gradually lowers in the high frequencies, following a non-linear mel scale. As such, the mel-spectogram can be considered an approximation of the human auditory system, which was the motivation behind its introduction in 1976 [7]. To calculate the MFCC the STFT power spectra is mapped onto the mel scale and for each mel frequency the logarithm of the power is calculated. The discrete cosine transform (DCT) is applied on the mel log powers and the amplitude of the resulting spectrum represents the MFCCs [8].

The chroma features used in this study is the *chromagram*. A chromagram is a spectrogram where all frequencies are mapped to 12-dimensional vectors representing the relative power of 12 pitches in the chromatic scale. A chromagram is computed by taking the STFT of an audio waveform and transforming it to a log-frequency spectrum. The resulting log-frequency spectrum is summed across octaves, resulting in a relative power for each pitch [9].

The tempo of a track is the speed or pace of that track and is often measured in beats per minute (bpm). A beat is a repeating rhythmic stress sequence in a track. To compute the tempo a *tempogram* is generated, which consists of the following steps: a novelty curve representing sudden changes in energy is computed by 1) the STFT of a waveform, 2) taking the logarithm of the magnitude and 3) compute the discrete derivative. The novelty curve is then split into shorter windows and an auto-correlation is done with time-shifted copies of itself. Each frequency bin of the tempogram is then weighted with a log-normal distribution and the maximum value of the weighted tempogram is the tempo of the track. [10]

Crest factor is measured as the ratio of the L_∞-norm and the L_2-norm, or the highest peak and the RMS of the signal. The CFr is always ≥ 1, a low CFr value means that the peaks are small in the signal while a high means large short duration peaks [11]. This can be use to get a measurement of sudden loudness changes in the music.

The FMA dataset also includes meta-data for the tracks. We extract and use audio duration as a feature, as different genres tend to vary in length.

To reduce dimensionality, small excerpts are processed at a time rather than the entirety of an audio track. [6] found that human classification performance does not increase significantly beyond 3s of continuous audio, so we opt for excerpts of a similar length.

4. PROPOSED ARCHITECTURES

Two distinct model architectures are adopted in this study: the traditional KNN classifier and a DenseNet. In the following, we briefly present the two model architectures. If you are unfamiliar with these topics, refer to [1] and [3].

4.1. K-Nearest Neighbors

The k-Nearest Neighbors algorithm is a non-parametric classification method, where a given input sample is classified by majority vote of its k closest training sample neighbors in the feature space. This makes the KNN algorithm particularly elegant and fast as it does not rely on learning anything about its training samples. The major drawbacks of using KNN, however, is that 1) the model needs to store all training samples, 2) not learning anything during training results in large computational cost at inference time and 3) it only approximates a function locally, which makes it very vulnerable to noise.

For above reasons, inputs to the KNN classifier are first transformed to a radically smaller dimensionality. For instance, rather than using the full mel-spectrogram as input we summarize the temporal axis across all frequency bands, which consequently means that we lose all temporal information.

4.2. DenseNet

In recent years, Convolutional Neural Networks (CNNs) have become very popular for visual classification and segmentation tasks, as they have proven effective at capturing spatial information in images. 2-dimensional features such as spectrograms can also be interpreted as images due to their rich spatial information and in this study we thus apply CNNs to find appropriate filters to extract that information for use in classification.

The second architecture proposed is an implementation of DenseNet [3] that has been adapted to the temporal nature of audio. [12]

A common issue that arises in deep CNNs is that a lot of information about the input tends to vanish by the end of the network (the vanish gradient problem). DenseNet is specifically designed to maximize the information flow through CNN layers by connecting layers via concatenation within blocks of layers and therefore improve the feed-forward properties of the network. The initial layer consist of a dilated convolutional layer with max-pooling, which allows for a broad capture of the moving window while keeping the parameter count relatively low. The output from the dilated convolutional layer is forwarded into two sequentially positioned Dense Blocks

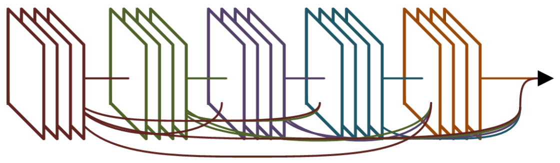


Fig. 1: Illustration of a Dense Block as used in DenseNet. [3]

connected by a Transition layer. The transition layer consists of 1x1 convolution followed by average-pooling. The output of the last dense block is passed through a single convolutional layer with max-pooling and finally a fully connected softmax layer to output the predicted class distribution.
Figure 1 depicts a Dense Block as used in our implementation of the DenseNet.

5. EXPERIMENTAL DETAILS

This section aims to provide the reader with experimental details required to reproduce the results presented in this study. Models are trained on a total of X samples (of which 10% are used for validation in the case of DenseNet) and tested on Y samples. Training of models is highly parallelized (multiple processes in the case of KNN; CUDA for DenseNet). The harmonic mean of precision and recall (denoted the F_1-score) is the primary measure used for performance evaluation. Baselines are established and a backward feature selection is done using the KNN classifier to determine which features best differentiate music genres in the given setting. We then proceed to apply our learnings to a DenseNet as it is a more flexible model and thus more promising in terms of performance.

5.1. Baselines

The following two baselines are established: random prediction weighted by class distribution and prediction by audio duration alone. When predicting by audio duration, samples are standardized:

$$z = \frac{x - \bar{x}}{S}$$

where $\bar{x}$ and S is the sample mean and sample standard deviation, respectively, of samples x. For classification by audio duration, a KNN using $k = 3$, L2 distances and uniform weighting is applied.

5.2. Feature selection

A backward feature selection is performed to pick the most suitable subset of features. MFFCs are first standardized and then re-scaled by min-max normalization:

$$z = \frac{x - \min x}{\max x - \min x}$$

Layer	Architecture	Details
1	2D Convolution	kernel: 6, 64 filters, max-pool
2-9	Dense Block	kernel: 3, 64 to 96 filters
10	Transition Block	kernel: 1, 64 filters, avg-pool
11-18	Dense Block	kernel: 3, 64 to 96 filters
19	2D Convolution	kernel: 1, 96 filters, max-pool
20	Softmax Layer	7 classes

Table 2: Architecture of DenseNet as used in this study. The network consists of a total of 71,904 parameters across 20 layers.

Chromas are already on the $[0; 1]$ scale and need no change. Tempo, Crest and audio duration is re-scaled by min-max normalization.
For the feature selection, a KNN using $k = 30$, L2 distances and weights according to the inverse distance of samples is applied; F_1 is used as performance measure.

5.3. DenseNet

A CUDA-enabled 20-layer (71,904 parameters) DenseNet is trained for 60 epochs on a GTX1080 unit using a batch size of 256. The architecture of the network is described in Table 2. LeakyRELU is used as activation function throughout the network; for regularization, batch-normalization and a dropout of 0.4 is applied to all layers. For fast convergence, the Adam [13] optimizer is used with a learning rate of $5 \cdot 10^{-4}$ and a weight-decay of $5 \cdot 10^{-7}$. Samples are fed to the network as-is; no standardization nor normalization was applied.

6. RESULTS

In this section, we present our experimental results for the two models proposed in this study.
Weighted random prediction achieves a F_1-score of 0.14, whereas prediction by audio duration achieves a F_1-score of 0.17. Thus, a weak correlation between audio duration and genre has been shown. Unsurprisingly, the correlation is highly dependent on the genre in question; while electronic, hip-hop and classical achieve F_1-scores of 0.22, 0.21 and 0.24, respectively, jazz achieves just 0.04, which suggests that it might have a large variance in audio duration. As it turns out, the standard deviation of jazz is 332 s – more than three times larger than that of hip-hop (102 s).

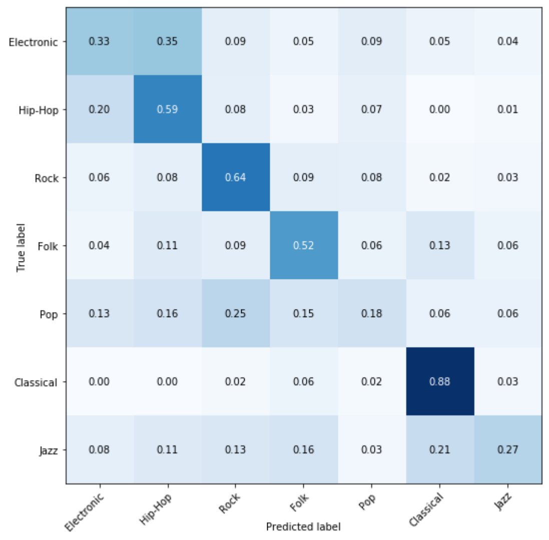
Running the backward feature selection produces results as shown in Table 3. Based on this experiment, we can conclude that MFCCs alone model the problem the best and that inclusion of additional features does not aid the KNN classifier significantly in terms of overall performance. It is also observed that removing chroma features significantly improves performance of a KNN, which suggests that chromas may not be suitable for music genre classification despite its musical origins. Similarly, a majority vote ensemble of

Feature set	F_1
11111	0.3903
01111	0.3331
10111	0.4605
11011	0.3713
11101	0.3833
11110	0.3807
00111	0.2542
10011	0.4741
10101	0.4681
10110	0.4469
00011	0.2174
10001	0.4814
10010	0.4623
10000	0.4851

Genre	F_1
Electronic	0.61
Hip-Hop	0.72
Rock	0.67
Folk	0.60
Pop	0.26
Classical	0.81
Jazz	0.39
Overall	**0.60**

Table 3: *Left:* F_1-scores from backward feature selection using the KNN classifier. 1 denotes that a feature is present and 0 denotes that the feature is absent. Sequence of features: MFCCs, chromas, tempo, crest factor, audio duration. *Right:* Per-genre F_1-scores for DenseNet.

KNNs trained on individual features and weighted by their F_1-scores achieves a score of just 0.45, significantly lower than the MFCCs alone.
A normalized confusion matrix for the MFCC-based KNN is displayed in Figure 2. It can be seen that electronic and hip-hop is commonly confused and both pop and jazz is difficult to distinguish from other genres.



True label \ Predicted label	Electronic	Hip-Hop	Rock	Folk	Pop	Classical	Jazz
Electronic	0.33	0.35	0.09	0.05	0.09	0.05	0.04
Hip-Hop	0.20	0.59	0.08	0.03	0.07	0.00	0.01
Rock	0.06	0.08	0.64	0.09	0.08	0.02	0.03
Folk	0.04	0.11	0.09	0.52	0.06	0.13	0.06
Pop	0.13	0.16	0.25	0.15	0.18	0.06	0.06
Classical	0.00	0.00	0.02	0.06	0.02	0.88	0.03
Jazz	0.08	0.11	0.13	0.16	0.03	0.21	0.27

Fig. 2: Confusion matrix for the KNN classifier using MFCCs as features.

Loss and accuracy during training is depicted in Figure 3. It can be seen that validation performance begins to stagnate around the 40-epoch mark. Test results of the DenseNet implementation is shown in Table 3. It can be concluded that music genre classification performance using MFCCs is highly dependent on the genre in question. The network models classical, hip-hop and rock extremely well (0.81, 0.72 and 0.67, respectively), while pop and jazz is confused with other genres (0.26 and 0.39, respectively). These results are in line with the per-genre performance of KNN, which also showed significantly worse performance for pop and jazz as opposed to the remaining genres, suggesting that it may be a model-agnostic issue more-so related to an overlap in the frequency contents of those genres.

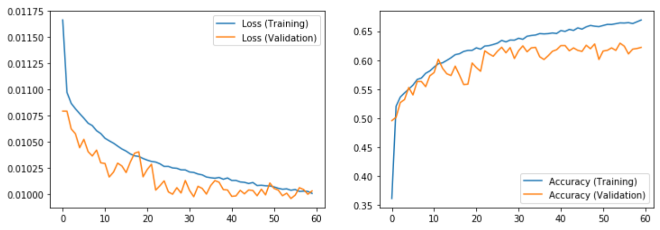


Fig. 3: Loss and accuracy of the DenseNet during training.

7. OPPORTUNITIES FOR FUTURE WORK

There are a number of interesting opportunities for further research in the field of music genre classification based on the findings of this study. As the presented machine learning methods model the problem fairly well based on our experiments, it could be interesting to attempt to reach human-level performance using additional features, larger models, more data and finally by use of an ensemble of developed models rather than relying on a single model.
While this study signifies the importance of MFCCs for use in music genre classification, it could be interesting to consider features that may be complementary to the spectral features used in this study. On the other hand, it could also be interesting to bypass the use of MFCCs and other hand-engineered features entirely and simply model a raw spectrogram at the expense of increased computational complexity. Although MFCCs have their basis in the human auditory system, they may not be the best-suited features for a machine to learn. In fact, it could even be interesting – although perhaps a bit extreme – to explore how well a neural network or similar machine learning methods can model music genres based on raw audio input alone.

8. CONCLUSION

We have shown experimentally that MFCCs are by far the best features for music genre classification among the ones presented in this study. Using 32 MFCCs extracted from a subset of the FMA dataset, a trained KNN and DenseNet achieves F_1-scores of 0.47 and 0.60, respectively. It is observed that classification performance is highly genre-dependent, which suggests that certain genres may have a similar profiles in the frequency domain.

9. REFERENCES

[1] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," in *The American Statistician*, 1992.

[2] Michal Defferrard et. al., "Fma: A dataset for music analysis," 2017.

[3] Gao Huang et. al., "Densely connected convolutional networks," 12 2016.

[4] George Tzanetakis and Perry Cook, "Musical genre classification of audio signals," in *IEEE Transactions onspeech and audio processing*, 2002.

[5] S. et. al. Lippens, "A comparison of humanand automatic musical genre classification," in *ICASSP 2004*, 2004.

[6] Klaus Seyerlehner et. al., "A comparison of human, automatic and collaborative music genre classification and user centric evaluation of genre classification systems," in *Adaptive Multmedia Information Retrieval*, 2011.

[7] Paul Mermelstein, "Distance measures for speech recognition - phychological and instrumental," in *Proceedings of the Joint Workshop on Pattern Recognition and Artificial Intelligence, Held at Hyannis, Massachusetts, June 1-3, 1976*, 1976.

[8] Fang Zheng, Guoliang Zhang, and Zhanjiang Song, "Comparison of different implementations of mfcc," *Journal of Computer science and Technology*, vol. 16, no. 6, pp. 582–589, 2001.

[9] Juan Pablo Bello, "Chroma and tonality," *Music Information Retrieval Course*.

[10] Peter Grosche, Meinard Müller, and Frank Kurth, "Cyclic tempograma mid-level tempo representation for musicsignals," in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2010, pp. 5522–5525.

[11] Stephen Boyd, "Multitone signals with low crest factor," *IEEE transactions on circuits and systems*, vol. 33, no. 10, pp. 1018–1022, 1986.

[12] Nicklas Hansen et. al., "Voice activity detection in noisy environments (https://github.com/nicklashansen/voice-activity-detection)," 2018.

[13] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," 12 2014.