GROOVE ESTIMATION USING DNNs

Alberto Barrera Herrero
Universitat Pompeu Fabra
alberto.barrera01@estudiant.upf.edu

Andrea Sofia Vallejo Budziszewski
Universitat Pompeu Fabra
andreasofia.vallejo01@estudiant.upf.edu

Diana Tyman Rondiak
Universitat Pompeu Fabra
diana.tyman01@estudiant.upf.edu

ABSTRACT

This paper explores the use of deep neural networks (DNNs) in accurately estimating the rhythmic quality of music known as "groove." The report highlights the effectiveness of DNNs in capturing the groove of tropical music using the Tropical Genres Dataset, which consists of 1,500 audio tracks and 1,500 images with the spectrogram from five tropical music genres. A pre-trained recurrent neural network (RNN) model is utilized to estimate the groove value of an audio file, and a convolutional neural network (CNN) is trained to classify the groove value to the proposed Latin music genres. Multiple evaluation metrics are used, including Mean Squared Error (MSE), Root Mean Squared Error and Binary Cross Entropy (BCE). However, the experiment did not meet expectations, as the smaller dataset of 1,500 audio files resulted in suboptimal outcomes despite using the appropriate batch size and optimizer. The findings emphasize the importance of careful consideration of dataset size while working with deep neural networks, and larger datasets can significantly improve the accuracy and efficiency of these networks.

1. INTRODUCTION

The study presented in this paper is centered around the application of DNNs to evaluate the rhythmic character of the music, commonly known as groove. The assessment of groove is crucial in both music analysis and production since it provides valuable insights into the danceability of different tracks and genres, which can help musicians and producers create more engaging music.

The report delves into the challenges of measuring and quantifying grooves and highlights how DNNs can address these challenges. Specifically, the paper discusses the effectiveness of DNNs in capturing the groove of tropical music, which has a complex rhythmic structure that can be difficult to analyze using traditional methods.

To conduct the research, the authors rely on the Tropical Genres Dataset, which contains a diverse range of tropical music genres, including salsa, reggaeton, and bachata. The dataset is used to train and test the DNNs, and several evaluation metrics are employed to ensure the accuracy and reliability of the results. The paper concludes that DNNs are a powerful tool for analyzing and quantifying grooves, particularly in tropical music. The research presented in this study contributes to the expanding body of research on music analysis and production, providing a deeper understanding of the rhythmic complexities of different music genres. Overall, this research has the potential to inform and improve the creation and production of music, particularly in the rapidly evolving world of music technology.

2. DATASET: TROPICAL GENRES DATASET

The dataset comprises an impressive collection of audio features from five tropical music genres: bachata, merengue, vallenato, cumbia, and salsa. These features are extracted using Librosa and Essentia in Python and include mel-frequency cepstral coefficients (MFCCs), spectral contrast, and tonal centroid.

The dataset is quite substantial, consisting of 3,000 files, with 1,500 audio tracks and 1,500 images with the spectrogram. All tracks are conveniently stored in MP3 format and have a standardized duration of 30 seconds. Additionally, the dataset includes a CSV file that contains essential information about each track, including its unique ID, genre, and file path.

3. ALGORITHMS

To accurately estimate the groove value of an audio file, a pre-trained RNN model is utilized, which was trained using computed groove values for the whole dataset using beats information extracted with Essentia. By analyzing the positions of the beats, we can measure the micro-timing deviations from the ideal beat duration. The mean of all these deviations in seconds is then converted to a percentage by dividing it by the average beat duration, which makes the value independent of the bpm. The code provided in the next figure is used to compute the groove value for all 1500 audio files, which are then saved into a .csv file.

```python
# Compute beat positions using Essentia
bpm, beats, _, _, _ = es.RhythmExtractor2013(method="multifeature")(x)

# Swing calculations
beat_durations = essentia.array([beats[i+1] - beats[i] for i in range(len(beats)-1)])
avg_beat_duration = sum(beat_durations) / len(beat_durations)

swing = essentia.array([abs(beat_durations[i] - avg_beat_duration) for i in range(len(beat_durations))])
meanSwing = np.mean(swing)

# Swing as percentage
swingValue = (meanSwing / avg_beat_duration) * 100

print(f'The Groove/Swing value for the example track is: {swingValue:.2f}')
```

```
The Groove/Swing value for the example track is: 0.63
```

Figure 1. Groove computing

The groove value from the CSV file is then converted to a class label, by classifying together audio files with similar groove values. First, a linear approach to this classification was tried. This means that boundaries between classes are defined by the expression:

Where i is the class number, max is the maximum value of the groove value and n_classes are the total number of classes.

With this linear split method, the resulting classes are the ones shown in figure 2.

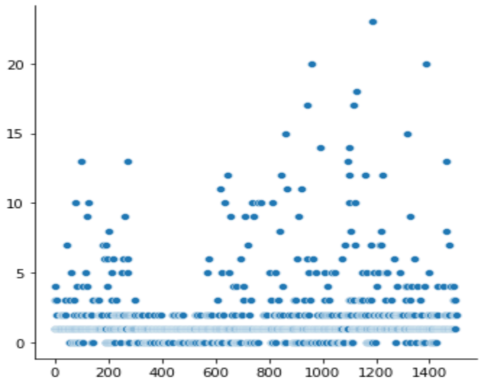

Figure 2. Linear split method.

As this classification tends to classify most of the groove values as 1, we tried a nonlinear approach. Specifically, we used a logarithmic function that yield the classification shown in figure 3.

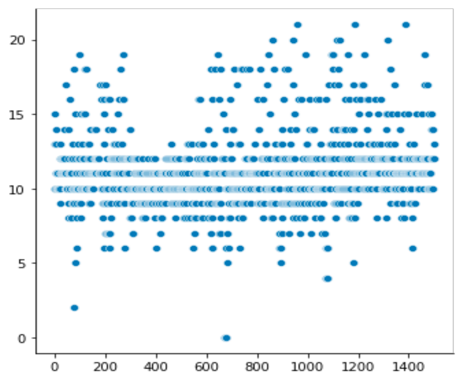

Figure 3. Logarithmic function.

```python
classes_groove = []
for i in range(len(df_processed['Groove_value'])):
  classes_groove.append(np.floor(float(df_processed['Groove_value'][i]) / 1))
```

```python
np.unique(classes_groove, return_counts=True)
```

```
(array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10., 11., 12.,
        13., 14., 15., 17., 18., 20., 23.]),
 array([216, 911, 220,  41,  22,  24,  13,  10,   5,   7,   9,   3,   5,
          4,   2,   2,   2,   1,   2,   1]))
```

Figure 4. Groove classes

For the next part of our approach, we decided to obtain the spectrograms of each song using Librosa and PIL libraries, in order to train the CNN. We resize the images in order to get the target size dataset of adequate dimensions for the CNN, which at the end is (1500, 224, 224, 3). We map each label computed previously to each song and feed the CNN.

Since we didn't get good results at first with the spectrogram approach, we experimented with using an onset detection function to train the model. For that, we used the Essentia OnsetDetection algorithm.

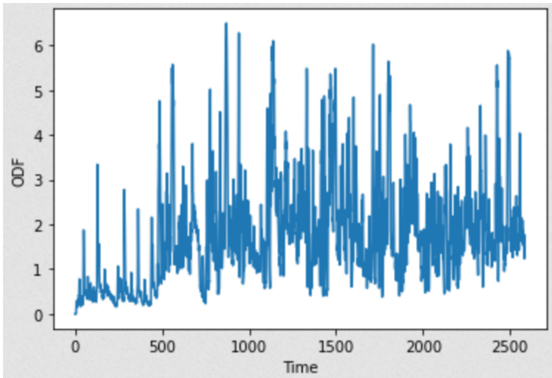

Figure 5. OnsetDetection

For the convolutional neural network, we use a classification model to classify each groove into its groove value class.

4. EVALUATION METRICS

To ensure the accuracy and validity of our results, we utilized multiple metrics, including Mean Squared Error and Binary Cross Entropy. These metrics were calculated using the appropriate functions provided by the reliable Scikit Learn library, providing a comprehensive and rigorous evaluation of our findings.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}|$$

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2$$

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1-y_i) \cdot log(1-p(y_i))$$

5. RESULTS

Our experiment with data and CNN did not meet our expectations as we faced several challenges that impeded our progress. One of the most significant issues was the difficulty in achieving satisfactory results with deep neural networks (DNNs) when dealing with small datasets.

DNNs tend to perform better with larger datasets because they can reduce overfitting by utilizing a higher number of parameters. Furthermore, larger datasets provide more examples for the network to learn from, enabling it to detect and understand intricate patterns better. On the other hand, a small dataset may not provide sufficient information for the network to create effective and robust models.

In conclusion, working with deep neural networks requires a sizable dataset to achieve optimal results. A larger dataset helps prevent overfitting and provides more opportunities for the network to learn, which enhances accuracy and performance. In our experiment, we trained the model with a batch size of 32 and 10 epochs using the optimizer Adam, with a focus on accuracy as the primary metric.

However, due to our dataset containing only 1500 audio files, the neural network approach did not yield the desired results.

- MSE: mean_accuracy = 0.05169, mean_loss = 0.0566

- Binary Cross Entropy: mean_accuracy = 0.1217, mean_loss = -23.47

6. CONCLUSIONS

Throughout our experiment, we ran into a handful of obstacles that are typically encountered when dealing with small datasets and deep neural networks, as well as errors on our part that caused less-than-ideal outcomes.

Primarily, our dataset was quite limited, with only 1500 audio files at our disposal. Despite implementing a suitable batch size and optimizer, we discovered that the spectrogram method required a more expansive dataset to function optimally. Regrettably, this made it impossible for us to attain the precision we had hoped for with our model.

We also realized that the window size (2048) of the generated spectrograms might have contributed to the poor performance of the model. For the neural network to be able to recognize the onsets from the generated spectrogram a small window size is needed and the chosen one was likely too large to yield useful results. Unfortunately, we did not have enough time to recompute all the spectrograms once we realized this.

Adding to this, and due to our lack of hands-on experience with CNNs we had some issues with the definition of the architecture that made the classification impossible.

We faced issues with our data split as well, as the chosen classes led to an uneven distribution of audio files based on their groove value. This may have resulted in a significantly biased model. Although we attempted to rectify the problem by modifying our splitting method, we were still unable to obtain satisfactory results or achieve perfectly balanced classes.

Looking ahead, there are several steps we can take to improve our results. One possibility is to expand our dataset or explore alternative machine learning models that can perform well on small datasets. We can also experiment with different parameters, optimizers, and network architectures to optimize our results further. Overall, our findings highlight the importance of careful consideration of dataset size while working with deep neural networks, and the significance of larger datasets in significantly improving the accuracy and efficiency of these networks.

7. REFERENCES AND BIBLIOGRAPHY

[1] Madison, G. (2006). “Experiencing Groove Induced by Music: Consistency and Phenomenology”. *Music Perception*, 24(2), 201–208.

[2] Janata, P., Tomic, S. S., & Haberman, J. (2012). “Sensorimotor coupling in music and the psychology of the groove”. *Journal of Experimental Psychology: General*, 141(1), 54–75.

[3] Butterfield, M. W. (2010). “Participatory Discrepancies and the Perception of Beats in Jazz”. *Music Perception*, 27(3), 157–176.

[4] Ohya, Y., Nakamura, K., & Tokunaga, T. (2012). “Extraction of groove feelings from drum data using non-negative matrix factorization”. *Soft Computing*

[5] Selfridge-Field, E., Balaban, M., Ebcioglu, K., Laske, O. E., Tanguiane, A. S., & Camilleri, L.

(1994). “Understanding Music with AI: Perspectives on Music Cognition”. *Notes*, 51(1), 168.

[6] Tidemann, A., & Demiris, Y. (2008). “A Drum Machine That Learns to Groove”. *Lecture Notes in Computer Science*, 144–151

[7] Eck, D., & Schmidhuber, J. (2002). A First Look at Music “Composition using LSTM Recurrent Neural Networks”. *IDSIA*, *IDSIA-07-02*

[8] Li, J. X., Han, L., Li, X., Zhu, J., Yuan, B., & Gou, Z. (2021). “An evaluation of deep neural network models for music classification using spectrograms.” *Multimedia Tools and Applications*, 81(4), 4621–4647.

[9] Choi, K., Fazekas, G., & Sandler, M. (2016). “Text-based LSTM networks for Automatic Music Composition”. *ArXiv* (Cornell University).