ARTIFICIAL MUSIC DETECTION THROUGH MACHINE LEARNING CLASSIFICATORS

Besedova Olga, Grati Marcello, Perego Gabriele

Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB), Politecnico di Milano
Piazza Leonardo Da Vinci 32, 20122 Milano, Italy
[olga.besedova,marcello.grati,gabriele2.perego]@mail.polimi.it

ABSTRACT

This research explores the potential to distinguish between artificially generated music and human-composed music. Various AI music generators were examined when solving this problem. By utilizing classical machine learning classifiers, implemented in Python language, we successfully trained a model that can accurately determine with approximately 90% of accuracy whether an audio track is real or generated by AI. Furthermore, with the same level of accuracy, the model can identify the specific AI generator that was used.

Index Terms— Artificial music detection, Audio classification, Supervised machine learning, Audio features, MFCC, Random Forest, PCA

1. INTRODUCTION

The field of AI music generation is rapidly evolving, with ongoing research and advancements in various techniques.[1][2] Some notable approaches and trends in AI music generation are:

- Deep Learning Models: Deep learning, particularly recurrent neural networks (RNNs) and variants like long short-term memory (LSTM) and transformer models, has been widely used for music generation. These models can learn patterns, sequences, and long-term dependencies in music data, enabling the generation of coherent and expressive compositions.
- Symbolic Music Generation: Symbolic music generation focuses on generating music in symbolic formats such as MIDI or sheet music. It involves modeling music as discrete events, notes, and chords. Approaches like variational autoencoders (VAEs) and generative adversarial networks (GANs) have been employed for generating symbolic music.
- Audio Synthesis: AI techniques are employed for audio synthesis to create realistic and expressive musical sounds. Techniques like sample-based synthesis, concatenative synthesis, and physical modeling synthesis are combined with machine learning methods to synthesize various instruments, vocals, and sound effects.
- Style Transfer and Remixing: AI allows for generation of music in different genres or imitating specific artists' styles. By training models on large music databases, AI systems can learn the characteristics of different genres or artists and generate music that reflects those styles.

This rapid growth is also increasing the focus on ethical and legal considerations [3][4]. Researchers and practitioners are addressing issues like copyright, attribution, cultural implications, and fairness in music creation to ensure responsible and legal use of AI-generated music. In particular, the main issues are:

- Copyright infringement: AI-generated music can potentially create issues regarding copyright ownership because, if it resembles existing copyrighted works too closely, it may raise concerns of infringement. So, determining the extent of originality and authorship can be complex.
- Intellectual property rights: AI-generated music may challenge traditional notions of intellectual property rights. Determining how to attribute authorship and giving credit to the individuals or entities involved in creating or training the AI system can be a legal and ethical concern.
- Licensing and royalties: AI-generated music might be used for commercial purposes, but this raises questions on how the necessary licenses and royalties should be handled with respect to the original rights holders, especially if AI-generated music is based on copyrighted works from other artists.
- Fair competition and market distortion: AI-generated music has the potential to disrupt the music industry and raise concerns about fair competition, impacting the opportunities for human composers and performers, so a balance between innovation and the music industry must be kept.

This situation asks for new instruments capable of recognizing the human or machine authorship of a music track, but this type of research is still in its infancy, with respect to other media contents like images, videos or speeches[5]. Deepfake recognition research is an active and rapidly evolving field focused on developing techniques to detect and identify manipulated or synthetic media.[6][7] Researchers in deepfake recognition employ various approaches to tackle this problem. Traditional methods rely on visual artifacts, inconsistencies, or anomalies introduced during the deepfake generation process. They analyze specific cues such as facial expressions, eye movements, skin texture, and other irregularities that may indicate the presence of a deepfake. Machine learning approaches, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), are used to train models on real and deepfake video datasets. Image and video forensics techniques examine metadata, compression artifacts, and low-level features to uncover signs of manipulation.
Our research is focused on analyzing deepfakes recognition techniques in the field of music. In particular, with Python language and Scikit-Learn [8], we extracted useful features from the raw audio data, we processed them to exploit their statistics properties, and finally we developed machine learning models able to classify our audio data, and able to detect if they were from AI-generated music or not.

2. PROBLEM FORMULATION

There are two problems that we considered in this paper:

- ***Real v.s. Fake problem*** **(RvF)** : is it possible, through machine learning techniques, to distinguish a "real" audio track (i.e. composition written by humans) from a "fake" one (i.e. composition created by artificial generator) ?
- ***Fake Generators problem*** **(FG)** : is it possible to associate correctly each audio track with its own generator, or in general class belonging, through machine learning techniques?

Both these questions can be reformulated from a mathematical point of view:

- Audio track : a vector of elements $x \in \Re$ defined as:

$$\underline{X} = [x_1, x_2, ...x_n] \tag{1}$$

- In the **RvF**, our goal is to implement a binary classification algorithm able to identify whether an audio track is "fake" or "real" with a function defined as:

$$f : \underline{X} \to \{0, 1\} \tag{2}$$

that associates to the audio track X one and only one element of the binary set $\{0, 1\}$, **"real" : 0** or **"fake": 1**.

- In **FG** instead, our goal is to implement a multiple classification algorithm able to associate to each audio track its class of generators using as a function:

$$f : \underline{X} \to \{G_1, G_2, G_3...G_m\} \tag{3}$$

that associates to each element of X one and only one class of generators defined symbolically with the name $G_1...G_m$ of the generator and with $m \in N^*$.

3. DATA PREPARATION

Only datasets containing classical piano music were selected for this study. Our dataset consists of five subsets of artificial audio tracks, generated by the following AI generators:

- **JSFakeChorale**: a MIDI dataset of 500 chorales generated by the KS_Chorus algorithm[9].
- **Riffusion**: an app for text-to-music generation with stable diffusion. The output audio files are directly in .wav format [10][11].
- **LA composer**: an AI tool which is a multi-instrumental music transformer for music generation. It allows the generation of new musical pieces based on a user's chosen composition. So it can imitate the given music style. The final output is in MIDI format[12].
- **RNN Music Generator**: an RNN model for music generation. The model is trained to learn patterns from raw sheet music and then use the model to generate new music[13][14][15].
- **Artificial Audio Multitracks Dataset (AAM)**: a dataset of artificial audio multi tracks. It consists of 3000 artificial music tracks based on real instrument samples and generated by algorithmic composition with respect to music theory[16].

For datasets of music composed by humans, we added two subsets: Maestro and MusicNet.

- **Maestro**: a dataset composed of about 200 hours of piano performances. The audio tracks are originally in .wav format[17].
- **MusicNet**: a collection of several hundred classical music recordings in .wav format[18].

To ensure uniformity and eliminate potential biases related to various file formats, stereo/mono configurations, and sampling rates, all audio tracks underwent preprocessing. For our specific problem, all MIDI files were converted to mono .wav files using the same procedure, with a sampling rate of 16 kHz. Similarly, all recorded .wav files were converted to mono, with the same sampling rate.
To construct a balanced dataset for training the model, we randomly selected audio tracks from each corpus of data such that the total number of artificially generated audio tracks matched the total number of tracks written by composers.

4. DATA PROCESSING PIPELINE

Before the computation of the classification, we manipulated the data to make them suitable for the model evaluation process. On data, we computed: **normalization**, **features extraction** and **dimensionality reduction**.

4.1. Normalization

Since the audio tracks were likely recorded in different environments or generated using different tools, they may exhibit significant variations in magnitudes. To bring all audio tracks to the same level of magnitude, we employed **Z-score normalization** formula :

$$z = \frac{x - \mu}{\sigma} \tag{4}$$

In this way, we resized the signals to adhere to the properties of a standard normal distribution [19] with zero mean, $\mu = 0$, and standard deviation equal to one, $\sigma = 1$.

4.2. Feature extraction

In our investigation, we extracted for each audio track some audio features commonly used for audio analysis [20][19]:

- **Mel-frequency cepstral coefficients (MFCC)** : represent the shape of the power spectrum of a sound signal, capturing the essential characteristics of the audio.
- **Short-Time Fourier Transform (STFT)** : divides a long signal into shorter segments of equal length and computes the Fourier transform of each segment to analyze the frequency content over time.
- **Mel-scaled spectrogram (MEL)** : is a representation of audio signals where the frequency scale is adjusted to mimic human hearing perception because humans do not perceive frequencies on a linear scale.
- **Spectral contrast Band Energy Ratio (BER)** : divides spectrogram frames into sub-bands, and energy contrast between peak and valley quantiles is used to differentiate clear, narrow-band signals from broad-band noise.

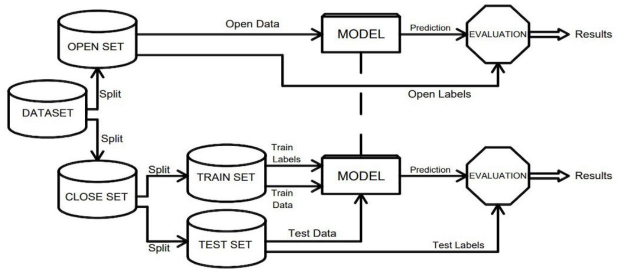


Figure 1: Model evaluation pipeline for RvF

4.3. Dimensionality reduction with PCA

Principal Component Analysis **(PCA)** is a dimensionality reduction technique which aims to identify the most informative features or patterns in a dataset by transforming the original variables into a new set of orthogonal variables called principal components [21]. In our case, we made use of PCA for feature reduction purposes, because it helps to reduce the dimensionality of high-dimensional datasets. To identify the optimal number of PCA components for our task, we did an analysis by identifying the relationship between the number of PCA components and F1 score of the model. As a result, we concluded that 100 PCA components is enough for better performance and accuracy. Thus, we decreased the number of features from more than 20000 to 100 components.

5. MODELS EVALUATION PIPELINE

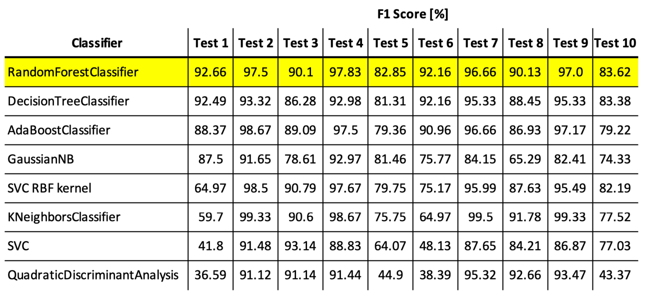
For our study, we relied on **machine learning** techniques. In this section, we described the procedure we adopted to solve RvF and FG, throughout well-known **supervised learning** algorithms [22] [19].

5.1. RvF models evaluation

Our approach involved identifying patterns within a training set and applying these patterns to predict or classify instances in the test set [23]. In our study, we introduced also an additional step to to obtain further proof of the good performance of our classifications. The initial dataset is divided into two balanced subsets: an **Open set**, formed by one class from ”fake” datasets and one class from ”real” datasets, and a **Close set** containing the remaining classes from both types. The Close set is used to train the models and evaluate their accuracy on known data. The Open set is used to test the accuracy of our models against completely new data. The complete scheme is shown in (fig. 1).

5.2. FG models evaluation

For the FG instead, the procedure involved standard multi-class classification: the input dataset is divided in two subsets: one for the training of the models (Train set), and the other one for prediction and evaluation of results (Test set). Also in this case, we considered a balanced input dataset: the same number of audio tracks for each dataset available. We considered the same models in both RvF and FG, to see the differences in their behaviors.



	F1 Score [%]									
Classifier	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10
RandomForestClassifier	92.66	97.5	90.1	97.83	82.85	92.16	96.66	90.13	97.0	83.62
DecisionTreeClassifier	92.49	93.32	86.28	92.98	81.31	92.16	95.33	88.45	95.33	83.38
AdaBoostClassifier	88.37	98.67	89.09	97.5	79.36	90.96	96.66	86.93	97.17	79.22
GaussianNB	87.5	91.65	78.61	92.97	81.46	75.77	84.15	65.29	82.41	74.33
SVC RBF kernel	64.97	98.5	90.79	97.67	79.75	75.17	95.99	87.63	95.49	82.19
KNeighborsClassifier	59.7	99.33	90.6	98.67	75.75	64.97	99.5	91.78	99.33	77.52
SVC	41.8	91.48	93.14	88.83	64.07	48.13	87.65	84.21	86.87	77.03
QuadraticDiscriminantAnalysis	36.59	91.12	91.14	91.44	44.9	38.39	95.32	92.66	93.47	43.37

Figure 2: Open test results (RvF)

6. IMPLEMENTATION

Our project is implemented using python language, in particular in the form of a notebook. This was the easiest choice since python already provides some of the most used machine learning libraries, such as Scikit-Learn[8], which contain already implemented versions of the classification models we used. We developed our code using Google Colab in order to take advantage of its computing power and ease of sharing work, and the dataset can be accessed through a Google drive folder. Everything can be replicated by using both the code and the dataset provided in our GitHub repository[24].

7. RESULTS EVALUATION

7.1. Feature Evaluation

After trying different approaches, we can say that extracting **MFCCs** seems the best way to process the raw audio data in terms of final accuracy in the results. The reasons behind this are various:

- MFCCs are designed to mimic the human auditory system's sensitivity to different frequency bands, so they're able to capture important timbral characteristics of the audio signal [19]
- They accomplish dimensionality reduction, extracting a relatively small number of coefficients from the audio signal that represent the essential information about the frequency content. This reduces the computational complexity and potential overfitting.
- Their invariance to noise and distortions, due to the logarithmic compression and the use of frequency bands in their computation.

7.2. Models evaluation

Our research is particularly focused on the choice of classification model. As shown by the results (fig. 2), the best model for our task is Random Forest [25], since it has an F1 score that never falls under 80% in any test on the Open Set pairs and it's always in the top ranking positions.
This particular model seems to be the best candidate for our task because of its ability to estimate feature importance, that allows it to identify which aspects of the MFCCs contribute most significantly to distinguishing between the two classes. It helps us understand the characteristics that differentiate the two

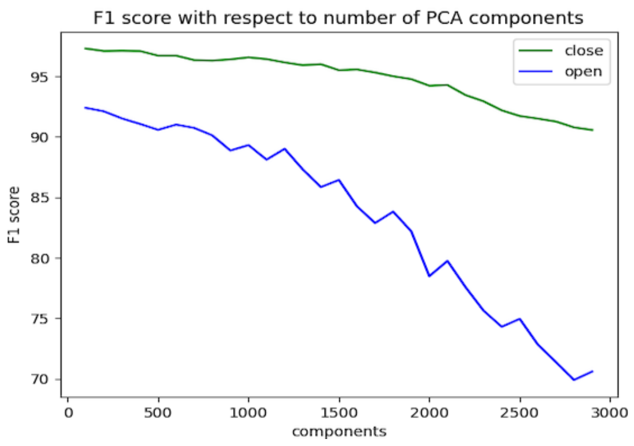


Figure 3: Number of PCA components evaluation

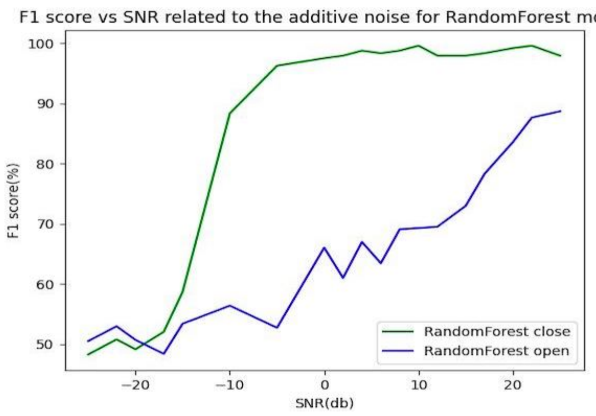
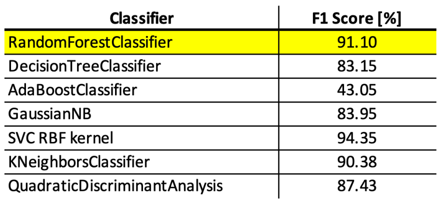


Figure 4: Effect of noise on F1 score



Classifier	F1 Score [%]
RandomForestClassifier	91.10
DecisionTreeClassifier	83.15
AdaBoostClassifier	43.05
GaussianNB	83.95
SVC RBF kernel	94.35
KNeighborsClassifier	90.38
QuadraticDiscriminantAnalysis	87.43

Figure 5: Fake Generators test results

and guides further analysis. Additionally, the model is robust in handling new, noisy, or ambiguous data. It can learn various patterns from training data and make reliable predictions on new instances, accommodating differences in audio quality, musical styles, and potential manipulations in ”fake” music.

7.3. PCA Evaluation

Using Random Forest performance as a reference, we are able to see that another determinant factor is the number of components we take as output after the PCA processing stage. As shown in the plot (fig. 3), the F1 score is inversely proportional to the number of components. This fact points out that reducing the dimensionality is useful in this task because it helps to remove noisy or irrelevant features that might introduce confusion to the classification model, improves generalization leading to less overfitting and better accuracy on unseen data (especially in our case with a small dataset), and in general reduces the computational complexity. Anyway this behavior tends to change as the model varies, so every model might require a different PCA tuning.

7.4. Noise Effect Evaluation

In order to test the model robustness, we add white noise with increasing intensity with respect to the original signals. It is possible to see from the plot (fig. 4) that the model is still able to classify correctly with a ***Signal-to-Noise-Ratio*** **(SNR)** around 10 - 15 dB. Once reached that point, the performance decreases reaching 50%. In addition to the robustness of the model, these results are very important to prove that the accuracy obtained does not come from any peculiar bias of the dataset, as it could have been the microphone noise in the case of real music signals or some particular missing frequency bands regarding the ”fake” generated music.

7.5. FG Results Evaluation

Regarding the generator identification problem, it is a simpler task with respect to the RvF, since the model is tested only on datasets seen during training (no Open test). For this reason, the majority of the classification models and types of feature gives good results without any significant difference (fig. 5). Another reason might be that the various generators produce very different music, even though we restricted to only instrumental classical piano.

8. CONCLUSION

What we learned from our observations is that in general this type of processing of the data exploits and highlights the differences between the dataset sources. This is especially useful in the case of FG, but it makes RvF task harder because, instead of generalizing the problem, the model tends to “memorize” which generators are ”fake” and which are ”real”, and then it decides accordingly. This problem happened every time we introduced a new dataset in our analysis, forcing us to review all processing chain parameters and make some changes. In the future it would be useful to expand the dataset in terms of number of different generators and number of samples from each one of them, in order to find more general patterns that might help this classification.
Another relevant factor to our results is the use of ”fake” MIDI music sources as well. This type of audio signal is virtually the same as a “real” music track that uses the same synthesizer, so a future improvement might be to try to extract features more related to composition style analysis, in order to be able to guess if a song has been written by a human or a machine just by looking at the music sheet.

9. REFERENCES

[1] C. Hernandez-Olivan and J. R. Beltrán, *Music Composition with Deep Learning: A Review*. Cham: Springer International Publishing, 2023, pp. 25–50. [Online]. Available: https://doi.org/10.1007/978-3-031-18444-4_2

[2] "Magenta: Make music and art using machine learning," https://magenta.tensorflow.org/, accessed: 13/07/23.

[3] B. L. T. Sturm, M. Iglesias, O. Ben-Tal, M. Miron, and E. Gómez, "Artificial intelligence and music: Open questions of copyright law and engineering praxis," *Arts*, vol. 8, no. 3, 2019. [Online]. Available: https://www.mdpi.com/2076-0752/8/3/115

[4] "Ai created a song mimicking the work of drake and the weeknd. what does that mean for copyright law?" https://tinyurl.com/hlsharvardedu, accessed: 13/07/23.

[5] "Chatgpt and academic integrity concerns: Detecting artificial intelligence generated content," http://www.langedutech.com/letjournal/index.php/let/article/view/49, accessed: 13/07/23.

[6] B. Hosler, D. Salvi, A. Murray, F. Antonacci, P. Bestagini, S. Tubaro, and M. C. Stamm, "Do deepfakes feel emotions? a semantic approach to detecting deepfakes via emotional inconsistencies," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2021, pp. 1013–1022.

[7] R. Durall, M. Keuper, F.-J. Pfreundt, and J. Keuper, "Unmasking deepfakes with simple features," 2020.

[8] S. Raschka, Y. H. Liu, V. Mirjalili, and D. Dzhulgakov, *Machine Learning with PyTorch and Scikit-Learn: Develop machine learning and deep learning models with Python*. Packt Publishing Ltd, 2022.

[9] "Js fake chorales," https://github.com/omarperacha/js-fakes, accessed: 13/07/23.

[10] S. Forsgren and H. Martiros, "Riffusion - Stable diffusion for real-time music generation," 2022. [Online]. Available: https://riffusion.com/about

[11] "Riffusion app," https://github.com/riffusion/riffusion-app, accessed: 13/07/23.

[12] "Los angeles music composer," https://github.com/asigalov61/Los-Angeles-Music-Composer, accessed: 13/07/23.

[13] "Generating Music using an LSTM Neural Network," https://david-exiga.medium.com/music-generation-using-lstm-neural-networks-44f6780a4c5, accessed: 13/07/23.

[14] I. Simon and S. Oore, "Performance rnn: Generating music with expressive timing and dynamics," https://magenta.tensorflow.org/performance-rnn, 2017.

[15] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, N. Shazeer, C. Hawthorne, A. M. Dai, M. D. Hoffman, and D. Eck, "Music transformer: Generating music with long-term structure," *arXiv preprint arXiv:1809.04281*, 2018.

[16] "AAM: Artificial Audio Multitracks Dataset," https://zenodo.org/record/5794629, accessed: 13/07/23.

[17] "The MAESTRO Dataset," https://magenta.tensorflow.org/datasets/maestro, accessed: 13/07/23.

[18] "MusicNet," https://paperswithcode.com/dataset/musicnet, accessed: 13/07/23.

[19] "Introduction to audio classification with deep neural networks," https://github.com/IliaZenkov/sklearn-audio-classification/blob/master/sklearn_audio_classification.ipynb, accessed: 13/07/23.

[20] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*, vol. 8, 2015, pp. 18–25.

[21] T. Kurita, "Principal component analysis (pca)," *Computer Vision: A Reference Guide*, pp. 1–4, 2019.

[22] B. Mahesh, "Machine learning algorithms-a review," *International Journal of Science and Research (IJSR).[Internet]*, vol. 9, pp. 381–386, 2020.

[23] J. Tan, J. Yang, S. Wu, G. Chen, and J. Zhao, "A critical look at the current train/test split in machine learning," *arXiv preprint arXiv:2106.04525*, 2021.

[24] "Capstone L01 : artificial music detection," https://github.com/marcello-grati/capstone_L01_artificial_music_detection, accessed: 13/07/23.

[25] A. Cutler, D. Cutler, and J. Stevens, *Random Forests*, 01 2011, vol. 45, pp. 157–176.