Self-Supervised Learning of Music Representations for Recommendation Systems

Andreas Cisi Ramos, Bruno Amaral Teixeira de Freitas
University of Campinas (UNICAMP), Institute of Computing

Abstract—This work explores self-supervised learning to derive music representations using the Audio Spectrogram Transformer (AST) model and SimCLR framework. It aims to enhance music recommendation systems by leveraging learned embeddings. The project employs the Free Music Archive (FMA) dataset and addresses challenges in audio augmentation, InfoNCE loss, and self-supervised fine-tuning.

Index Terms—Self-Supervised Learning, Music Recommendation, Audio Embeddings, SimCLR, AST Model.

1 Introduction

Music streaming platforms have become central to entertainment in the 21st century, representing one of the most widely used types of web-based applications and generating billions of dollars annually.

A critical factor in the success of these platforms is their ability to deliver personalized content that aligns with users' preferences. Traditional music recommendation systems often rely on collaborative filtering or metadata-based approaches, which can face scalability challenges when dealing with databases containing millions of tracks. Recent advancements in self-supervised learning offer a promising alternative, enabling the generation of robust and generalized embeddings without requiring labeled data.

This paper explores the application of the Audio Spectrogram Transformer (AST) [1], trained from scratch with a SimCLR [2] projection head, to derive music embeddings for recommendation systems. The primary objective is to evaluate how effectively these representations capture the concept of musical similarity as perceived by humans, without using labeled data during training, and to assess the quality of recommendations produced by a simple system leveraging these embeddings.

2 Materials and Methods

2.1 Dataset

We used the Free Music Archive (FMA) [3] small subset, which comprises 8,000 30-second audio tracks. Each track is also associated with metadata regarding information such as artist, genre and title.

To enable experimentation with limited computational resources, the tracks were truncated to the first 10 seconds. The training dataset consisted of the first 1000 tracks from the dataset, the validation dataset comprised the next 200 tracks, and finally, the test dataset consisted of another 200 tracks. Future experiments could explore training with more tracks and using their full extension.

2.2 Data Pre-Processing

The chosen dataset provides only .mp3 files, making preprocessing necessary. While representing the data as an array of sampled sound amplitudes, as in wav files, could be the simplest approach, we hypothesized that spectrograms might provide a more insightful representation of the nuances of musical structure.

A spectrogram is a graphical representation of audio, where the horizontal axis represents time and the vertical axis represents frequency. Each point on the plane is colored according to the amplitude of the given frequency at the given time, with more intense colors corresponding to higher amplitudes. This form of audio representation adds an extra second dimension to the data, essentially turning the task from audio processing into image processing. By converting the array of sampled amplitudes into a spectrogram, features of the track, such as the most prominent frequencies and how they vary over time, become more evident.

Despite providing a clearer picture of the data, simple spectrograms may not be the optimal way to portray music for a system designed to learn the meaning of song similarity or discrepancy from a human perspective. This is because humans are more sensitive to variations in lower frequencies. For example, it is much easier to distinguish a 500Hz sound from a 1kHz sound, while 10kHz may sound very similar to 10.5kHz.

For this reason, the amplitudes are normalized along the vertical axis according to mel scale, which attempts to better replicate the perception of human hearing. For comparison, figures 1, 2 and 3 represents the different ways the same song can be represented and used as an input for a model.

2.3 Model Architecture

The architecture combines the AST model available in Hugging Face with a SimCLR-inspired projection head implemented with Pytorch comprised of a 2-layer MLP network using ReLu activation, with each layer containing 512 units. The AST model processes mel-spectrogram inputs, with the

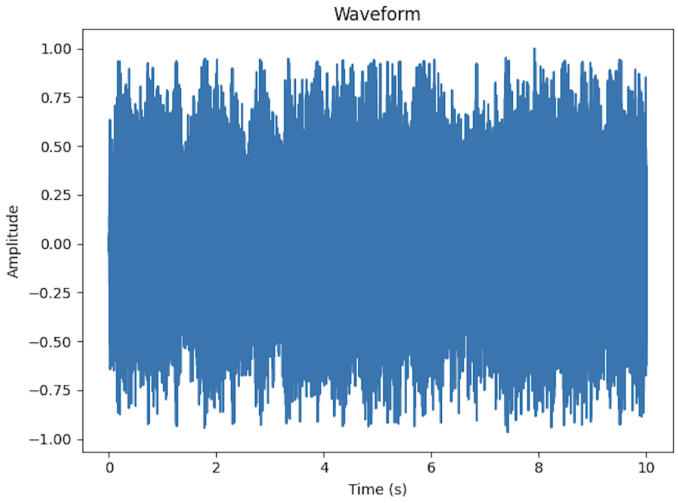

Fig. 1. Sampled amplitude representation for the first 10 seconds of the dataset's first song.

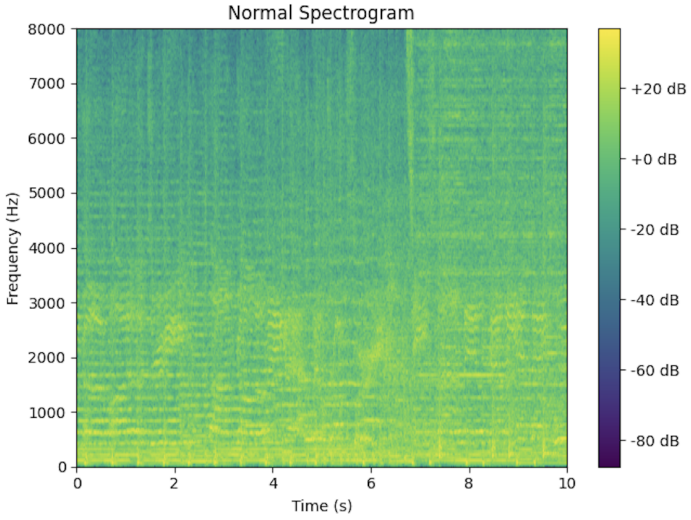

Fig. 2. Normal spectrogram representation for the first 10 seconds of the dataset's first song.

projection head mapping the 512-dimensional encoded embeddings to a 128-dimensional latent space for contrastive learning.

2.4 Training Pipeline

In order to better experiment with the pure capabilities of self-supervision, we chose to train the model initialized with random weights.

Training was conducted for 10 epochs using Adam optimizer with a learning rate of 3e-5. The training dataset was grouped in batches of 8 tracks, and Google Collaboratory's T4 GPUs were used.

2.4.1 Data Augmentation

Data augmentation was applied on the audio, before generating the spectrograms. It includes noise addition, pitch change and time stretch effects. The first was generated by adding random values to the amplitude array. These value

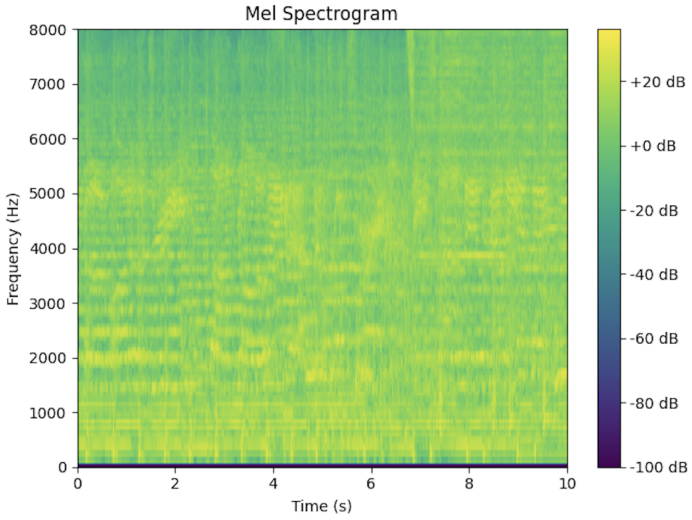

Fig. 3. Mel Spectrogram representation for the first 10 seconds of the dataset's first song.

were generated from a normal distribution with mean 0 and standard deviation of 1, scaled by 0.05. The other effects were implemented with the software Sound eXchange (SoX). For time stretch, the track's length could be stretched or compressed by a random amount ranging from 0% to 20%. For pitch change, the track's pitch is adjusted by a random value of -2, -1, 0, 1, or 2 semitones.

By applying data augmentation 2 times over a given track, the augmented audios were used for generating the pair of spectrograms over which SimCLR was applied.

2.4.2 Loss Minimization

The experiments described in this paper utilized the InfoNCE loss, a contrastive loss function central to the SimCLR framework. This metric aims to maximize the embedding similarity for augmented views of the same audio track (positive pairs) while minimizing the similarity between these views and all other tracks in the batch (negative pairs). By doing so, the encoder is encouraged to generate meaningful and discriminative representations of the tracks, as it learns to approximate similar samples and distance dissimilar ones within the projection head's latent space.

A key advantage of this approach is its self-supervised nature: no labels are used during the optimization process. It is worth noting that similar training strategies could be applied using a supervised genre classification pretext task by leveraging a similar projection head.

The InfoNCE loss can be formally defined as:

$$\mathcal{L}_{\text{InfoNCE}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{2N}\mathbb{1}_{[k\neq i]}\exp(\text{sim}(z_i, z_k)/\tau)}$$

where z_i and z_j represent the latent vectors of the positive pair, $\text{sim}(\cdot,\cdot)$ is the similarity function (e.g., cosine similarity), τ is the temperature parameter and N is the number of tracks in the batch.

Figure 4 shows the training and validation loss over the epochs for the described approach. The simultaneous minimization of both the train and validation losses indicate

that the model did not suffer from overfitting, even though no improvements were noticed after the 4th epoch.

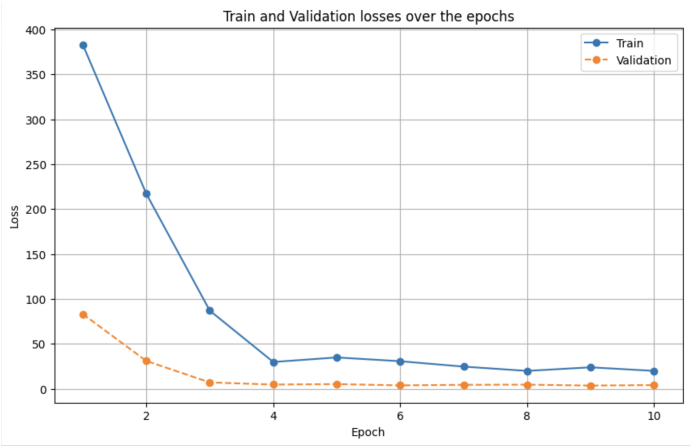

Fig. 4. Validation and training losses over the epochs for the SimCLR training.

2.5 Dimensionality Reduction

To better understand how music is represented in the latent space, 2D dimensionality reduction techniques were employed. The primary method used was t-SNE, applied with different perplexity values to visualize various perspectives of the data. PCA was also considered but was discarded due to its inability to produce results as clear as t-SNE, likely because it does not effectively capture the non-linearities inherent in musical structure.

To compare the trained and untrained embeddings more effectively, dimensionality reduction was applied to both scenarios, even though they were plotted separately.

Although this paper does not aim to learn representations that explicitly group tracks by genre—at least not directly—the existing genre labels from the FMA dataset were used to color the points in the t-SNE plots. This approach enables a discussion of how the model handled different genres, despite never being trained or exposed to such a feature.

2.6 Building a simple recommendation system

A simple recommendation system was also built to test the practicality of the learned embeddings for music recommendations. In this system, a subset of the test dataset was embedded using the trained model, and one of the remaining tracks was selected as a reference. In practical applications, the embedded subset could represent the available songs on a streaming platform, while the reference would be a specific song the user is interested in, potentially serving as the basis for recommendations.

The recommended track is determined by identifying the one with the embedding most similar to the reference embedding. Both Euclidean distance and cosine similarity were evaluated for measuring similarity; however, cosine similarity demonstrated better performance and was ultimately used in the experiments.

3 Results and Discussion

3.1 Embeddings distribution

Qualitative analysis on t-SNE plots using the validation dataset shows that the trained encoder was able to organize the data into a web-like pattern comprised of multiple filaments of grouped tracks, considerably different from the randonly distributed untrained embeddings. These results can be visualized in figures 5, 6 and 7.

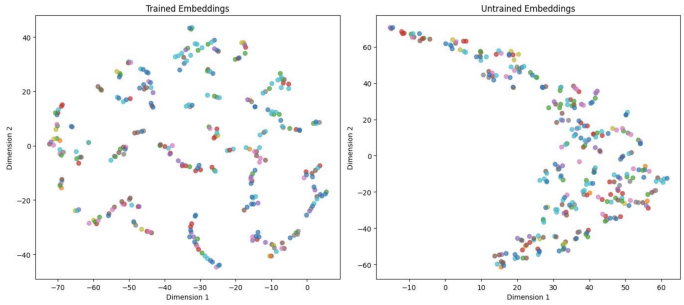

Fig. 5. t-SNE plot of the validation dataset representing trained and untrained embeddings in the same space. Perplexity parameter was set to 3

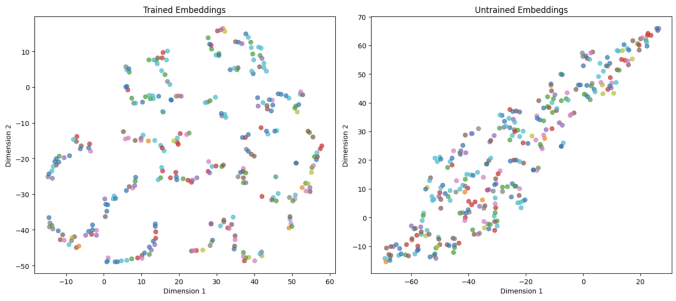

Fig. 6. t-SNE plot of the validation dataset representing trained and untrained embeddings in the same space. Perplexity parameter was set to 7

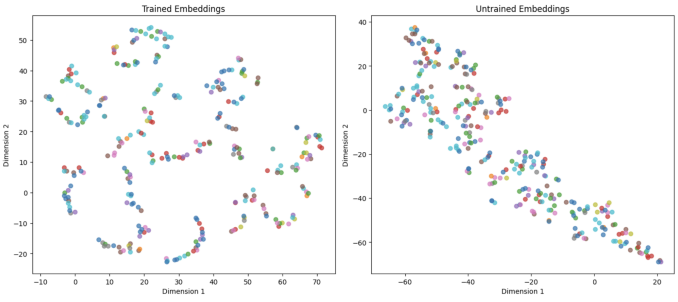

Fig. 7. t-SNE plot of the validation dataset representing trained and untrained embeddings in the same space. Perplexity parameter was set to 10

Despite the fact that the model was never exposed to the concept of genre due to the self-supervised nature of the pretext task, it was expected that the tracks would be grouped by this label - considering that this might be one of the most common means by which humans measure music similarity.

A curious aspect of the plotted distributions for the trained embeddings is that data was grouped in a way that is not dependent of genre. While this could indicate that the model was not able to properly learn the musical relations between tracks, it could also be argued that the AST is not trying to group music by genre, but by other more subtle elements of musical structure. Considering the model was not optimized by taking genre into consideration due to the self-supervised nature of the pretext task, the results point that the genre labels may not be the best information for understanding music similarity. Further studies would be needed to address this question and clarify the type of information in which the encoder's latent space focused.

3.2 Qualitative Tests

For a system like the one we are working on, it is difficult to conduct traditional quantitative tests and produce objective metrics, as the recommendations rely heavily on subjective human judgment. Therefore, we conducted a qualitative analysis to evaluate the performance of the recommendation system.

We selected 50 songs from the test dataset, based on the trained model, and used them as queries to recommend new songs. The goal was to compare the queried music with the recommended tracks and assess the quality of the recommendations.

To categorize the results and create a form of analyzable metric, we established three groups based on the analysis of the recommendations:

- Satisfactory: The similarity between the recommended song and the queried music was clear and evident.
- Partially Satisfactory: Some similarities could be identified, though the match was not perfect.
- Unsatisfactory: The recommended songs were not similar at all to the queried music.

This qualitative categorization allowed us to separate the recommendations and obtain the following results for the trained embeddings:

- 24 songs were categorized as Satisfactory (48% of the total).
- 13 songs were categorized as Partially Satisfactory (26% of the total).
- 13 songs were categorized as Unsatisfactory (26% of the total).

For comparison, we evaluated the same experiment using the random recommendations generated by the untrained encoder:

- 5 songs were categorized as Satisfactory (46% of the total).
- 10 songs were categorized as Partially Satisfactory (30% of the total).
- 35 songs were categorized as Unsatisfactory (34% of the total).

The results can be better visualized in the pizza plots at figures 8 and 9

The tests revealed that the trained model significantly outperformed random recommendations. The percentage of satisfactory recommendations increased from 10% to 48%, while unsatisfactory recommendations dropped from 70% to 20%.

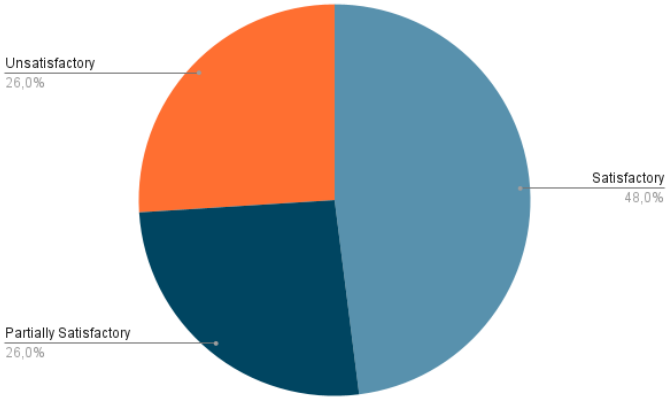

Fig. 8. Results for the qualitative tests with the trained model.

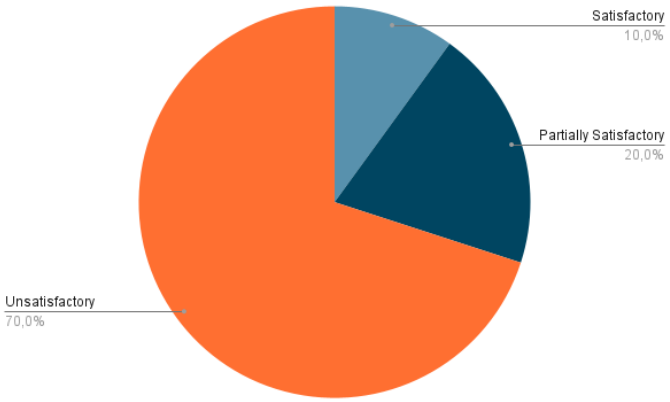

Fig. 9. Results for the qualitative tests with the untrained model.

Although these results indicate that the system is effective in achieving its goal, there remains considerable room for improvement. Employing more robust encoders trained using a combination of self-supervised and supervised methods, or leveraging recommendation algorithms that integrate track metadata (e.g., genre) with embedding similarity, could yield even better outcomes.

While testing with the trained embeddings, some interesting insights emerged. Although 7 of the recommended songs shared the same genre, they lacked sufficient similarity to one another, which led us to classify them as unsatisfactory or partially satisfactory. This reinforces the idea that recommendation systems could learn more than just obvious labels like genre or artist.

Another relevant point is that the vast majority of satisfactory songs belonged to the pop and hip-hop genres, which raises the hypothesis that the model may have learned more intrinsic characteristics of these styles. This could be due because beats and basslines are more effectively captured by the Mel spectrogram. These findings provide valuable insight into the system's performance, showing that while there is room for improvement, the approach is promising for generating music recommendations

based on features such as composition, timbre, and flow, rather than relying solely on genre or artist.

4 Conclusion

This work presents a self-supervised framework for music recommendation, using the AST model fine-tuned with SimCLR. Qualitative testing based on human preferences showed that the system learned meaningful music representations, providing recommendations that significantly outperformed those from an untrained model.

The analysis of trained embeddings and test results also suggests that genre alone is not always the best criterion for grouping similar songs. Subtler features often play an important role in what people consider similar music.

Although the framework achieved its goal of generating embeddings suitable for recommendation systems, there is still room for improvement. Enhancements could involve combining supervised methods with the self-supervised approach or refining the recommendation algorithms.

All the code used in this work is publicly available at https://github.com/Brunoatf/MusicSSRL.

Acesse o repositório aqui.

Acknowledgments

This project was supported by UNICAMP and developed during the discipline Introduction to Self-Supervised Learning. The authors thank prof. Dr. Marcelo da Silva Reis for teaching the fundamentals that made this work possible.

References

[1] Y. Gong, Y.-A. Chung, and J. Glass, "Ast: Audio spectrogram transformer," *arXiv preprint arXiv:2104.01778*, 2021, accepted at Interspeech 2021. [Online]. Available: https://doi.org/10.48550/arXiv.2104.01778

[2] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International Conference on Machine Learning (ICML)*, vol. 119. PMLR, 2020, pp. 1597–1607.

[3] M. Defferrard, K. Benzi, P. Vandergheynst, and X. Bresson, "Fma: A dataset for music analysis," *arXiv preprint arXiv:1612.01840*, 2017, iSMIR 2017 camera-ready. [Online]. Available: https://doi.org/10.48550/arXiv.1612.01840