Album cover generation from audio

Team members: Txus Bach (Advisor), Jordi Martìnez, David Solano, Matilde Sartori
Universitat Politècnica de Catalunya
Framework: PyTorch

Abstract

This university project explores the intersection of art and technology by proposing an approach to generating visually captivating album covers from music audio inputs. Our main objective is to provide artists with an innovative avenue for immersive self-expression, redefining the boundaries of creative possibilities in the digital age.

The core of our project centers on the development of a stable diffusion model that takes music audio as input and produces album covers that are not only aesthetically beautiful but also deeply connected to the essence of the associated songs. Leveraging the power of deep learning, the model analyzes the unique audio characteristics and emotional nuances of each track to generate visually evocative covers that complement the music's theme.

To validate the effectiveness of our proposed stable diffusion model, we also built and compared results with Generative Adversarial Networks (GANs). This comparative analysis allows us to gain insights into the strengths and limitations of each method, enabling us to refine our model for optimal performance.

Introduction

The world of music thrives on captivating album covers, visually encapsulating the essence of an artist’s sonic creation. Inspired by the fusion of audio and visual arts, we embarked on a journey to generate album covers directly from song audio.
Leveraging the power of diffusion models, a state-of-the-art generative technique, we dive into unexplored artistic territories, making this project a captivating endeavor.

Diffusion models, with their ability to synthesize complex and high-fidelity images, have recently emerged as groundbreaking technology. By applying these models to the audio domain, we embrace the challenge of merging audio and visual modalities, pushing the boundaries of artistic expression in a unique and exciting manner.

Motivation

In this project, we present our methodology for translating audio signals into visually stunning representations using diffusion models. Through this fusion of artistry and

cutting-edge technology, we redefine the boundaries of creative possibilities in the digital edge, providing artists with a new avenue for immersive self-expression.

Objective

Our main goal is to create album covers that are beautiful and related to the song they are associated with.

Model

For the deployment of our project, we utilized three essential components: the Python library Musicnn, GANs and diffusion models. We have built two different models to compare them one with the other. We made a model using conditioned GANs and another one using a conditioned diffusion model. To ensure seamless integration and easy distribution, we constructed these components within Docker containers.

Musicnn

Musicnn is a Python library developed by Jordi Pons at the Music Technology Group, Universitat Pompeu Fabra in 2019. It provides a straightforward interface for extracting audio features from music files.
The library employs the concept of taggrams, which are compact representations of audio signal features associated with semantic tags. Taggrams bridge the gap between raw audio data and extracted information, enabling users to efficiently analyze and manipulate audio content.
With Musicnn, you can extract features such as rhythm, harmony, and melody from audio files. The library utilizes machine learning techniques and pre-trained models to achieve high accuracy in audio classification and tagging tasks. This makes Musicnn a valuable tool for music information retrieval, content organization, and recommendation systems.

GANs

GAN (Generative Adversarial Network) models are a class of generative models that consist of two main components: a generator and a discriminator. The generator aims to generate synthetic samples that resemble the real data, while the discriminator's objective is to distinguish between real and generated samples.

During training, the generator generates samples from random noise as input, attempting to deceive the discriminator into classifying them as real. On the other hand, the discriminator learns to correctly classify real and generated samples. This adversarial training process leads to an iterative competition between the generator and discriminator, resulting in the refinement of both models.

GAN models leverage a min-max game framework, where the generator and discriminator optimize their objectives simultaneously. As training progresses, the generator becomes more proficient at generating realistic samples that can deceive the discriminator. Ultimately, the goal is to achieve a point where the generator generates samples that are indistinguishable from the real data.

The advancements in GAN models have led to improved realism and creative generation, making them valuable tools in artificial intelligence and machine learning research.

Diffusion model using DreamBooth

Diffusion models, a class of generative models, have gained attention in generative artificial intelligence. They employ a step-by-step diffusion process, transforming data by adding calibrated noise levels. During training, they learn transformations to map an initial distribution to the target data. Diffusion models generate new samples by reversing the learned transformations from random noise.

They excel at capturing complex data patterns, generating high-quality samples, and performing tasks like image generation and denoising. With their versatility, diffusion models enhance creativity, data analysis, and synthetic data generation in various applications. For our project we are using a text-to-image diffusion model.

In our work we decided to utilize DreamBooth: a fine-tuning approach of text-to-image models; for our case, diffusion models. DreamBooth enhances the capabilities of diffusion models by incorporating advanced techniques for training and generating high-quality samples. By utilizing DreamBooth, we further enhance the performance and potential of diffusion models in tasks such as image generation, denoising, and data analysis.
This combination of diffusion models and DreamBooth empowers our work to achieve superior results in capturing complex data patterns and generating realistic samples.

Dataset

Initially, we encountered challenges in finding a suitable dataset for our project. To overcome this, we decided to leverage the Spotify API provided by Spotify for Developers. The Spotify API enables us to extract a list of songs, which we iterated upon to download the corresponding audio files and album cover images. Additionally, the Spotify API allows us to retrieve various song features.
As a result, our dataset consists of 23,588 audio files and 20,693 album cover images. This comprehensive dataset serves as the foundation for our project, providing the necessary information for analysis, modeling, and generation tasks.

Methods

GANs

We have developed and trained three distinct GAN models using the entire dataset. Each model serves a different purpose and incorporates various conditioning factors.

The first GAN model we built is an unconditional GAN, which means it generates samples without any specific conditioning input. It learns to capture the underlying distribution of the dataset and produce novel outputs based solely on that learned representation.

The second GAN model we constructed is a conditional GAN that takes into account a variable provided by Spotify called "category." We pass this variable through an embedding, using nn.embedding. This is a 20-classes-variable that summarizes the general characteristics of a song and can be seen as analogous to the musical genre. By conditioning the GAN on the category variable, the model is able to generate samples that align with specific genres, capturing the essence and style associated with each category.

Lastly, the third GAN model we developed is conditioned on multiple factors, including the song's title, artist name, release day, and the first five output features extracted from Musicnn. These features, derived from the Musicnn library, provide additional information about the musical content. Same as we did with the second GAN, we pass these variables to the model by doing an embedding with the only difference that in this case we use an embeddingBag. This method takes the embedding of each word in the prompt and sum them up.

By incorporating these conditioning factors, the GAN model is able to generate samples that not only exhibit characteristics related to the genre but also align with specific songs, artists, and release dates.
Here is an example of the prompt: "cover album of Top Lists song, titled 2000, released in 2023-03-17, by Manuel Turizo Marshmello, with the following music tags techno, pop, fast, electronic, vocal"

Diffusion Model

We have implemented a fine tuned version of stable diffusion models by using Dreambooth. The fine-tuning process involved training the model with the entire dataset, enabling it to capture the intricate patterns and nuances present in the data. Similar to the previous GAN model, this diffusion model is conditioned on various factors, including the song's title, artist name, release day, and the first five output features obtained from Musicnn.

By conditioning the model on these specific inputs, it gains the ability to generate samples that not only align with the given song, artist, and release information but also exhibit the desired musical characteristics associated with them. This approach allows for highly tailored and personalized sample generation, ensuring that the generated outputs reflect the intended attributes of the input conditioning factors.

Dreambooth, plays a crucial role in enhancing the efficiency and effectiveness of the model. By utilizing this technique, we are able to reduce the computational complexity and training time required for fine-tuning, making the process more efficient and scalable.

Production environment

In our project's production environment, we have set up a Google Cloud instance running on Ubuntu. This instance serves as the foundation for our web application. We have installed essential components such as Python, Docker, Apache, etc.

Within this instance, we have integrated a web server along with a Docker container housing the Musicnn library. Placing the Musicnn library in the Docker container allowed us to resolve dependency conflicts between various libraries, ensuring seamless processing.

Our web application is built using Flask and provides users with an interface to input an audio song file in .mp3 format, along with the song's title, artist's name, and release date. Once the user submits the information, the web application forwards it back to the instance.

Subsequently, the input data is passed to the Docker container containing the Musicnn library. The Musicnn library processes the input and returns the requested information. Combining this output with the user-provided data, a prompt is created.

This prompt is then displayed on the web application and sent outside the instance to a GoogleColab file. The decision to utilize GoogleColab was driven by its ability to leverage GPU processing.

Within the GoogleColab environment, we have deployed a stable diffusion model fine-tuned with Dreambooth. This model takes the prompt as input and generates an image corresponding to the cover album for the given song. The resulting image is then passed back to the instance and displayed on the web application, providing users with a visually captivating album cover closely related to the input audio.

Result and discussion

Result

In this study, we explored the performance of different models for our project, including GANs with varying conditions and the Stable Diffusion Model. Here are compared the different results and different models:

1. **GAN Unconditioned:** the GAN without any conditioning was trained for 200 epochs over the course of one day, but it was not utilizing GPU resources. This resulted in prolonged

training times and limited convergence. The lack of conditioning might have contributed to poor results, as the model failed to capture specific patterns and structures related to our dataset.
2. **GAN Conditioned to Variable "Category":** for the GAN conditioned to the variable category, we trained it for 440 epochs using the GPU of ColabPro+. The training time was significantly reduced to approximately 5 minutes per epoch, thanks to the GPU acceleration. While the results improved compared to the unconditioned GAN, they were not yet satisfactory. The model showed some ability to generate content related to the category but lacked fine-grained details and coherence.
3. **GAN Conditioned to Song Detail and Musicnn Features:** we trained this third GAN for 170 epochs, with each epoch lasting 5 minutes using the powerful GPU of ColabPro+. The computational resources required for this approach were considerably higher, but the results did not justify the cost.
4. **Stable Diffusion Model:** finally, we experimented with the Stable Diffusion Model fine-tuned with Dreambooth. We trained it for a single epoch over 40 minutes using the GPU of the normal Google Colab. Although the training time per epoch was more extensive than the GANs, the results were markedly superior. The Stable Diffusion Model demonstrated its ability to generate high-quality content with impressive coherence and realistic features.

Discussion

The results indicate that, without easy access to GPU resources, training deep learning models like GANs can be challenging and time-consuming. The lack of GPU utilization during the unconditioned GAN training led to suboptimal results, limiting its ability to learn meaningful representations from the data.

Furthermore, we observed that conditioning the GANs to specific variables or features did lead to some improvement in the generated content. However, the cost of training with more complex conditions, such as the song details and Musicnn features, did not provide commensurate improvements in the final results, making them computationally inefficient. On the other hand, the Stable Diffusion Model, despite requiring a longer time per epoch compared to GANs, outperformed them in terms of both training stability and generated content quality.

In conclusion, the availability of GPU resources significantly impacts the training time and performance of deep learning models, particularly in complex tasks like image generation. The Stable Diffusion Model emerges as a promising alternative, offering better results and stability compared to traditional GANs, which tended to be slower and less effective in our experiments.

Conclusion and future work

Conclusion

In conclusion, we are highly satisfied with the outcomes of our study, particularly with the successful implementation of the diffusion model.Our objective to create album covers that are visually appealing and thematically relevant to the associated songs has been achieved.

We have observed the significant advantage of utilizing existing algorithms over developing proprietary algorithms from scratch. This approach allows us to leverage established techniques and optimizations, resulting in superior performance.

Furthermore, we acknowledge the time required for GANs to learn and generate outputs. Longer conditioning prompts can negatively impact learning and output quality. Striking the right balance between prompt length and desired results is crucial.

Through the successful utilization of the diffusion model and understanding the strengths and limitations of GANs, we have made substantial progress towards our goal. The implications of this research extend to various domains where image generation and relevance are paramount. Continued exploration and optimization in this field hold promising prospects for even more exceptional outcomes in the future.

Future work

To achieve better final results we propose some different future works that can be applied to this project:

1. **Enhancing GANs Architecture:** Further research can be conducted on the development of the GANs architecture. One possible approach is to extend the training duration by adding more epochs.

2. **Change the aggregation Method in the Third GAN:** In the third GAN, which incorporates the use of embeddingBag, we aim to investigate the impact of different aggregation methods on the model's output. Currently, the model utilizes the sum of the vectors. As part of future work, we propose exploring an alternative approach by taking the mean of the vectors instead.

3. **Comparative Analysis of Diffusion Models:** The next area of exploration involves comparing the performance of our existing diffusion model with different diffusion models, as we did for GANs. We plan to experiment with two additional variations: an unconditioned diffusion model and a conditioned diffusion model utilizing only the "category" variable.