Music Visualization Network
Towards Audio to Image Synthesis using cDCGAN

Natalia Koliou*
Artificial Intelligence Applications, M.Sc. in Artificial Intelligence
University of Piraeus & NCSR Demokritos

Abstract

This project aims to develop a Music Visualization Network[1] that generates visual representations of music from various art movements in History of Art. The main goal is to establish a coherent and meaningful mapping between the auditory and visual modalities, enabling the exploration of intricate semantic relationships that might potentially exist between music and paintings throughout different time periods. By employing advanced techniques in deep learning and cross-modal analysis, the MVNet enhances our understanding of how music and paintings relate to each other, offering new ways to perceive and appreciate their semantic correlation.

1 Introduction

The field of music visualization has gained significant attention as a means of exploring the connections between music and visual art. In this paper, we introduce Music Visualization Network (MVNet), a composite neural network that employs Deep Learning techniques to generate image-based representations of music data from five different art periods: renaissance, baroque, classical, romantic, and modern.

Our MVNet consists of two distinct deep learning models: a Convolutional Audio Encoder and a Conditional Deep-Convolutional GAN [2]. The Encoder model reduces the dimensionality of our audio data, turning them into small vectors called "conditional embeddings". These embeddings, along with random noise, are fed into our cDCGAN to generate visually appealing images that match the style of the input audio. While the specific class for each embedding is not precisely defined, as they do not follow some one-hot encoding scheme, we anticipate that the generated images will exhibit

*natalykoliou@yahoo.gr (mtn2205)

[1]Inspired by the paper "Crossing You in Style: Cross-modal Style Transfer from Music to Visual Arts" by Cheng-Che Lee, Wan-Yi Lin, Yen-Ting Shih, Pei-Yi Patricia Kuo, and Li Su. [1].

characteristics that align with the art-period classes of the corresponding audios - from which the embeddings were derived. This expectation arises from the fact that embeddings within each class are anticipated to share similarities with one another while being noticeably different from embeddings belonging to other classes.

The significance of our SVNet lies in its various applications. It can function independently as a standalone model for audio-to-image translation tasks, such as generating visual representations of soundscapes. Additionally, it can serve as the initial component of a style-transfer neural network model. In this approach, the SVNet extracts the artistic style from the input audio and transfers it to the visual domain, while the second component, some Style Transfer Network (STNet) refines the generated images to align them more closely with the desired style.

2 Dataset

For our experiments, we used a newly compiled dataset called WikiArt-IMSLP, which includes a collection of music recordings and paintings organized by decades. From this dataset, we extracted both images and audio files to construct our customized dataset: the Musart-Dataset. Unlike WikiArt-IMSLP, our dataset is organized according to art periods to fit our project needs. Finally, we split the dataset into training and testing subsets to ensure a robust evaluation of our model's performance.

Our training dataset consists of 1500 audio files (8.91 seconds each), with 300 files in each of the 5 art-period classes and 4500 images (64x64 RGB), with 900 images in each class. To augment the data, we mirror the images, resulting in a total of 9000 images. To train our SVNet we need to create pairs of audios and images, in order to establish a meaningful connection between the auditory and visual domains. Each audio is therefore paired with six images - three original images and their respective mirrored versions. So eventually, we end up having a training dataset of 9000 pairs.

To evaluate our SVNet, we need a testing dataset with a format similar to the training dataset. Although paired data is not required, we will create pairs for consistency. With that being said, our testing dataset consists of 30 audio files, with 6 files in each of the 5 art-period classes, and 90 images, with 18 images in each class. To augment the data in the same way as we did for the training dataset, we mirror the images, resulting in a total of 180 images. Pairs of audios and images are created, resulting in a total of 180 pairs.

2.1 Data Preprocessing

In order to feed the audio and image data into our network, it is necessary to preprocess them in a meaningful way. To ensure effective representation of the audio data in our model, we extract mel-spectrograms, which provide a more useful representation compared to simple spectrograms. Mel-spectrograms capture the perceptual properties of sound and are better suited for our encoding needs. Regarding the specific details:

- The music signals are sampled with a rate of 22.05 kHz. We compute the mel-spectrogram using a Hamming window of size 1024 and a hop size of 256. The mel-filterbank consists of 128 filters. The resulting mel-spectrogram is divided into three parts, each with a length of 2.97 seconds. These three parts are assigned to three input channels, resulting in a dimension of 128×256×3.

- The reason we divide the mel-spectrogram into three channels, as opposed to using a single channel, is primarily due to the limitations of single-channel features in accurately representing music information. Therefore dividing the mel-spectrogram into multiple channels allows for a more comprehensive and informative representation of the music data.

Having turned the audios in the right format, we proceed to create our dataloader. Each data entry consists of three components: the mel-spectrogram representation of an audio clip, the corresponding era label, and a pair-image.

- The mel-spectrogram captures the frequency content of the audio, providing a visual representation of its acoustic features.

- The era label signifies the art period to which the audio belongs - and therefore the pair (audio, image) as well.

- The pair-image is associated with the audio, to help the model learn the correlation between visual and audio representations. We mentioned earlier that the Musart-Dataset contains 6 times more images than audios. In other words, each audio sample is paired with six different images. This deliberate pairing process enables the dataset to exhibit greater diversity in semantic correlations between visual and audio representations.

Figure 1: Random Image Samples from the Musart-Dataset

3 Models

The Music Visualization Network (MVNet) incorporates two key models: the Convolutional Audio Encoder and the Conditional Deep-Convolutional Generative Adversarial Network (cDCGAN). The Audio Encoder is pretrained to establish a meaningful mapping between audio and image representations, while the cDCGAN is trained using an adversarial training procedure to generate visually coherent images that capture the style of the input audio.

3.1 Audio Encoder

The Convolutional Audio Encoder is responsible for reducing the dimensionality of the audio data and extracting meaningful features. It takes the mel-spectrogram representation of the audio as input and performs several convolutional and pooling operations to capture local and global features. The output of the encoder is a condensed representation called a "conditional embedding."

The input to our encoder is the mel-spectrograms, represented by a 3-channel feature map with dimensions of 128x256. This input undergoes a series of convolutional layers, each followed by batch normalization and activation functions (ReLU). The convolutional layers capture local patterns and relationships within the mel-spectrograms. By using filters of different sizes, the encoder can learn to recognize and extract relevant audio features at different scales. The batch normalization helps stabilize the training process, while ReLU introduces non-linearity for more expressive representations.

The Encoder produces conditional embeddings of size 10 that serve as the condition for the cDCGAN. These embeddings encode essential characteristics of the input audio and indicate the era class, similar to the way conditional GANs use one-hot encodings as conditions. In our case, the embeddings capture relevant information about the audio, enabling the cDCGAN to generate visually coherent images that align with its style overall. Later in the paper, we will explain how the outputs of our encoder serve as actual conditions for the cDCGAN - the process of mapping the audio features to meaningful embeddings that represent the era class.

3.2 Generative Adversarial Network

The Conditional Deep-Convolutional Generative Adversarial Network (cDCGAN) is the main component of the Music Visualization Network (MVNet). It consists of two main parts: the generator and the discriminator. The generator takes the conditional embeddings along with random noise as input, and aims to generate images that resemble the style of the input audio. The discriminator, on the other hand, acts as a binary classifier, distinguishing between real images from the Musart-Dataset (pair-images) and the generated images from the generator. It receives the real images and the generated images as distinct inputs and tries to correctly classify them. Through an adversarial training process, the generator and discriminator learn from each other, with the generator striving to produce images that fool the discriminator, and the discriminator improving its ability to discriminate between real and generated images.

3.2.1 Generator

The generator starts with a linear block, consisting of a fully connected layer followed by a leaky ReLU activation function. This block maps the input vector to a higher-dimensional feature representation. The deconvolutional blocks in the generator progressively upsample the feature map, refining it to resemble the target image. Each

deconvolutional block includes an upsampling operation, a convolutional layer, an instance normalization layer, and a leaky ReLU activation function - some blocks may also include self-attention modules to capture long-range dependencies. Finally, the output of the generator passes through a hyperbolic tangent (Tanh) activation function, ensuring that the pixel values of the generated image are within the range of [-1, 1].

The self-attention mechanism used in the last layers of the generator enhances the generator's ability to capture long-range dependencies and focus on important regions within the feature maps. It uses three convolutional operations to compute query, key, and value representations of the input feature map. The attention mechanism calculates attention scores by performing matrix multiplication and applies softmax normalization to obtain weights representing the importance of different spatial locations. These weights are then used to re-weight the value representation, producing an attention output that emphasizes relevant information. By combining the attention output with the original feature map through a scaling factor, controlled by the learnable parameter gamma, the generator can effectively refine the image generation task.

3.2.2 Discriminator

The discriminator takes as input an image and performs a series of convolutional operations to extract relevant features and classify the input as real or fake. Its architecture consists of several convolutional blocks, each responsible for capturing different levels of detail and patterns in the input image. Each convolutional block includes a convolutional layer, a batch normalization layer (except for the first block) for normalization and stabilization, a leaky ReLU activation function for introducing non-linearity, and an optional dropout layer for regularization. The convolutional blocks progressively decrease the spatial resolution of the input image while increasing the number of channels. The output feature map is flattened and passed through a linear layer, which maps the flattened features to a single output dimension. This output dimension represents the likelihood of the input image being real or fake. The sigmoid activation function is then applied to the output to obtain a probability score between 0 and 1.

4 Training

The training process of the Music Visualization Network (MVNet) consists of two main stages: training the Encoder and training the cDCGAN. Initially, the Encoder is trained separately to optimize its parameters and generate high-quality conditional embeddings that effectively represent the audio data. Once the encoder is trained, we proceed to train the cDCGAN through an adversarial training procedure.

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 32, 64, 128]	1,568
BatchNorm2d-2	[-1, 32, 64, 128]	64
ReLU-3	[-1, 32, 64, 128]	0
Conv2d-4	[-1, 64, 32, 64]	32,832
BatchNorm2d-5	[-1, 64, 32, 64]	128
ReLU-6	[-1, 64, 32, 64]	0
Conv2d-7	[-1, 128, 16, 32]	131,200
BatchNorm2d-8	[-1, 128, 16, 32]	256
ReLU-9	[-1, 128, 16, 32]	0
Conv2d-10	[-1, 256, 8, 16]	524,544
BatchNorm2d-11	[-1, 256, 8, 16]	512
ReLU-12	[-1, 256, 8, 16]	0
Conv2d-13	[-1, 512, 4, 8]	2,097,664
BatchNorm2d-14	[-1, 512, 4, 8]	1,024
ReLU-15	[-1, 512, 4, 8]	0
Conv2d-16	[-1, 1024, 2, 4]	8,389,632
BatchNorm2d-17	[-1, 1024, 2, 4]	2,048
ReLU-18	[-1, 1024, 2, 4]	0
Flatten-19	[-1, 8192]	0
Linear-20	[-1, 10]	81,930

Layer (type)	Output Shape	Param #
Linear-1	[-1, 16384]	1,818,624
LeakyReLU-2	[-1, 16384]	0
Interpolate-3	[-1, 1024, 8, 8]	0
Conv2d-4	[-1, 512, 8, 8]	4,719,104
InstanceNorm2d-5	[-1, 512, 8, 8]	0
LeakyReLU-6	[-1, 512, 8, 8]	0
Interpolate-7	[-1, 512, 16, 16]	0
Conv2d-8	[-1, 256, 16, 16]	1,179,904
InstanceNorm2d-9	[-1, 256, 16, 16]	0
LeakyReLU-10	[-1, 256, 16, 16]	0
Interpolate-11	[-1, 256, 32, 32]	0
Conv2d-12	[-1, 128, 32, 32]	295,040
InstanceNorm2d-13	[-1, 128, 32, 32]	0
LeakyReLU-14	[-1, 128, 32, 32]	0
Conv2d-15	[-1, 16, 32, 32]	2,064
Conv2d-16	[-1, 16, 32, 32]	2,064
Conv2d-17	[-1, 128, 32, 32]	16,512
Softmax-18	[-1, 1024, 1024]	0
SelfAttention-19	[-1, 128, 32, 32]	0
Interpolate-20	[-1, 128, 64, 64]	0
Conv2d-21	[-1, 64, 64, 64]	73,792
InstanceNorm2d-22	[-1, 64, 64, 64]	0
LeakyReLU-23	[-1, 64, 64, 64]	0
Conv2d-24	[-1, 8, 64, 64]	520
Conv2d-25	[-1, 8, 64, 64]	520
Conv2d-26	[-1, 64, 64, 64]	4,160
Softmax-27	[-1, 4096, 4096]	0
SelfAttention-28	[-1, 64, 64, 64]	0
Conv2d-29	[-1, 3, 64, 64]	1,731
Tanh-30	[-1, 3, 64, 64]	0

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 64, 32, 32]	3,136
LeakyReLU-2	[-1, 64, 32, 32]	0
Conv2d-3	[-1, 128, 16, 16]	131,200
BatchNorm2d-4	[-1, 128, 16, 16]	256
LeakyReLU-5	[-1, 128, 16, 16]	0
Conv2d-6	[-1, 256, 8, 8]	524,544
BatchNorm2d-7	[-1, 256, 8, 8]	512
LeakyReLU-8	[-1, 256, 8, 8]	0
Conv2d-9	[-1, 512, 4, 4]	2,097,664
BatchNorm2d-10	[-1, 512, 4, 4]	1,024
LeakyReLU-11	[-1, 512, 4, 4]	0
Conv2d-12	[-1, 1024, 4, 4]	4,719,616
BatchNorm2d-13	[-1, 1024, 4, 4]	2,048
LeakyReLU-14	[-1, 1024, 4, 4]	0
Flatten-15	[-1, 16384]	0
Linear-16	[-1, 1]	16,385
Sigmoid-17	[-1, 1]	0

Figure 2: STNet's Architecture: Audio Encoder and cDCGAN

4.1 Training the Encoder

During the training of the Encoder, we adopt a triplet-based approach, which spans across 15 epochs. To begin, we initialize the Encoder model and configure the Adam optimizer with a learning rate of 0.00005 and betas of (0.5, 0.999). Throughout each epoch, we iterate over the data loader, which provides us with batches of audio data (batch size is set to 1). For each batch, we generate positive and negative samples by randomly selecting samples from other batches. Positive samples have the same era label as the current batch, while negative samples have different era labels. These samples are passed through the encoder. The triplet margin loss function is then applied to calculate the loss between these anchor, positive, and negative encoded samples. In our implementation, we set the margin to 3.0 and p to 5. The margin defines the minimum distance that should be maintained between the anchor and the negative sample, while p controls the norm type used to calculate the distance. By setting a higher margin, we encourage larger separations between the anchor and negative samples. After computing the loss, it is backpropagated through the network, and the optimizer is updated accordingly.

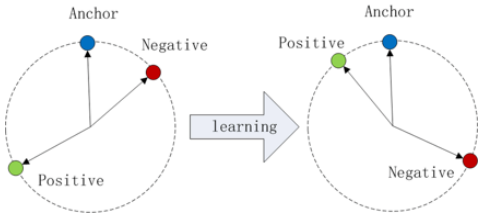

Figure 3: Triplet-margin Loss

$$\mathcal{L}_{\text{triplet}} = \sum_{i=1}^{B} \max\left(0, \|\mathbf{a}_i - \mathbf{p}_i\|_p^p - \|\mathbf{a}_i - \mathbf{n}_i\|_p^p + \text{margin}\right)$$

The triplet margin loss plays a crucial role in obtaining meaningful conditional embeddings during the training of the Encoder. Unlike conventional methods that rely on fixed one-hot encodings for each art-period class, our embedding space (10-dimensional) exhibits a more complex structure where classes form diverse clusters. These clusters

represent distinct groups of audio samples associated with a specific art-period. The triplet margin loss guides the Encoder to learn embeddings that clearly separate different art-period classes within the embedding space. It encourages embeddings from the same class to be grouped closely together, whereas embeddings from different classes to be farther apart.

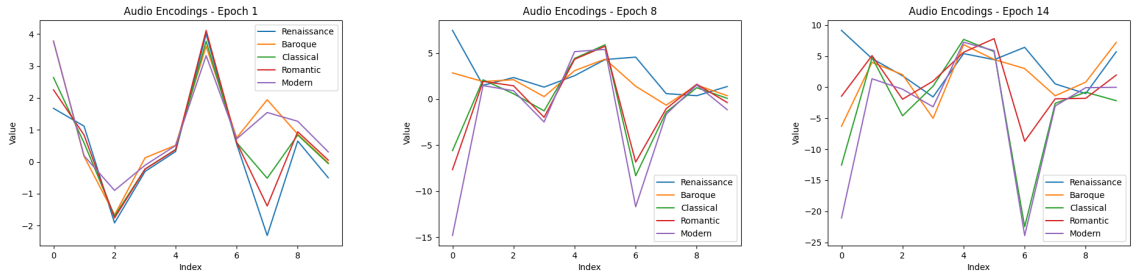

Figure 4: Random Conditional Embeddings from different training epochs

The loss curve shows a gradual reduction in the loss value as the training progresses, resembling the shape of the cotangent inverse function. Starting from an initial loss of 3.0, the curve gradually approaches a value of around 0.20.

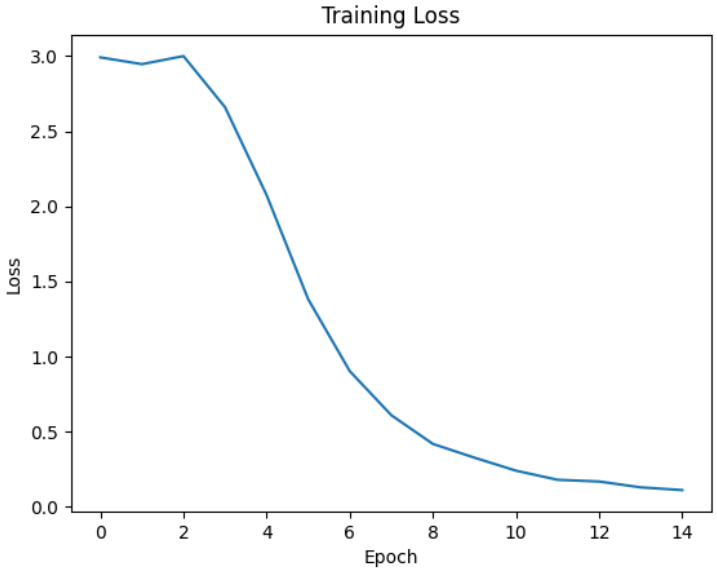

Figure 5: Encoder Training Loss

4.2 Training the cDCGAN

The training of the cDCGAN involves an iterative process where the generator and discriminator models are trained alternately. The training process spans over 100 epochs

- we observed that the model often converges much earlier. We begin by initializing the encoder, generator, and discriminator models. The encoder is loaded with the pre-trained parameters obtained from the previous training phase. The generator and discriminator models are initialized, and the respective optimizers (Adam) are set up with specific configurations. For the generator, we use a learning rate of 0.002 and betas values of (0.5, 0.999). The discriminator, on the other hand, has a learning rate of 0.00005 and the same betas values.

For each epoch, we iterate over the data loader, which provides batches of audio data (batch size is set to 15). We encode the mel-spectrograms using the encoder (conditional embeddings) and concatenate them with random noise vectors, to create the input for the generator. The generator generates synthetic images based on the input, which are then fed into the discriminator along with their corresponding pair-images from the dataset. Finally, the discriminator evaluates them and provides feedback on their authenticity.

During cDCGAN training, the generator is optimized using the Binary Cross Entropy (BCE) loss function. This loss function compares the predicted labels of the generated images (fake logits) with the real labels and guides the generator to generate more realistic images. The discriminator is also optimized using the BCE loss function. It calculates separate losses for fake images and real images. For fake images, the predicted labels (fake logits) are compared with fake labels (zeros with a smoothing factor). For real images, the predicted labels (real logits) are compared with real labels (ones minus the smoothing factor). The loss for the discriminator is the average of these two losses, encouraging it to accurately classify both real and fake images.

$$\mathcal{L}_{\text{gen}} = -\frac{1}{B}\sum_{i=1}^{B}\log(D(G(\mathbf{z}_i))) \quad \text{and} \quad \mathcal{L}_{\text{dis}} = -\frac{1}{B}\sum_{i=1}^{B}\left(\log(D(\mathbf{x}_i)) + \log(1 - D(G(\mathbf{z}_i)))\right)$$

The loss curves of the Generator and Discriminator in the cDCGAN training reveal interesting patterns. The Generator's loss starts around 2.25 and experiences an immediate drop to 1.65. It remains relatively stable throughout the training process, indicating a consistent performance in generating images. On the other hand, the Discriminator's loss begins at 0.65, slightly decreases to 0.50, and remains stable over the epochs as well.

We observe a phenomenon of significant variations between epochs in the generated images, while the variations within each epoch are relatively small. This behavior suggests a potential issue of model collapse. Several factors could contribute to this problem. Firstly, the size of the dataset, consisting of 9000 samples across five classes, may not provide enough diversity and complexity for the model to capture the intricate characteristics of each art period. Additionally, the limited training time due to restricted GPU availability might hinder the model's ability to converge to a more diverse and accurate representation. Furthermore, achieving a perfect balance between the generator and discriminator networks, which is crucial for stable and diverse image generation, is a challenging task in itself.

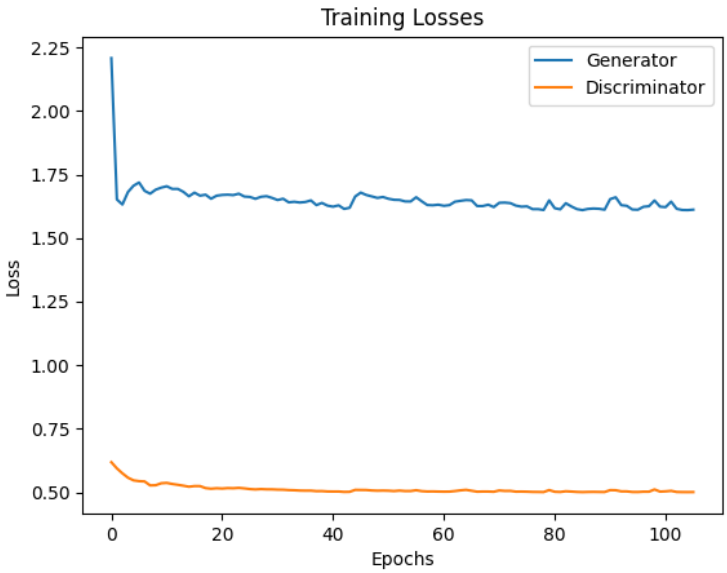

Figure 6: cDCGAN Training Loss

Below we present some images generated in different epochs along with their corresponding pair-images. We can see that there is some sort of similarity in terms of style between the generated images and their pair-images. This similarity indicates that the MVNet successfully captures and reproduces the desired artistic characteristics and style from the corresponding audio in the pair.

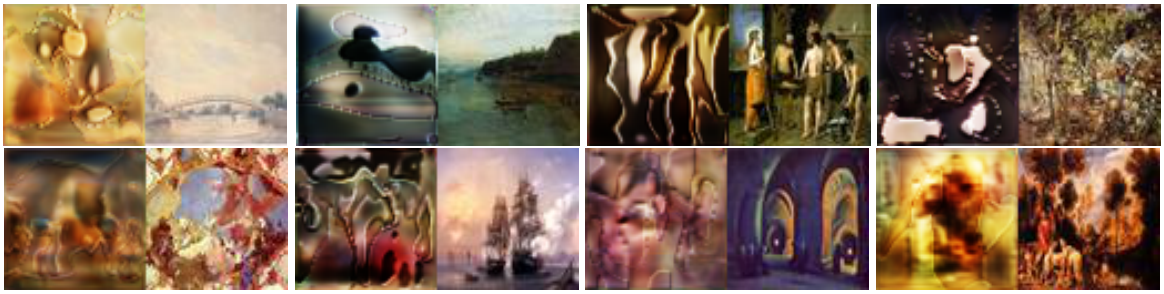

Figure 7: Generated-Images and their Pair-Images during different Training Epochs

5 Testing

To test our MVNet, we selected one musical piece for each of the five classes representing different art periods. From each musical piece, we extracted six segments, resulting in a total of six audio clips for each class. These audio clips serve as the input for generating visual representations. In order to be compatible with the training dataset, where each

audio was paired with six images, we followed the same pairing scheme for the testing dataset. Therefore, for each class, we created a total of 36 pairs.

Below, we present a collage of the generated images for each class. The generated images exhibit a high degree of similarity within each class, with only subtle variations. This phenomenon (model collapse) indicates that the generator is not generating diverse outputs and is unable to capture the full range of style variations within each art period. Despite this limitation, we present our results as they provide an initial glimpse into the potential of our Music Visualization Network.

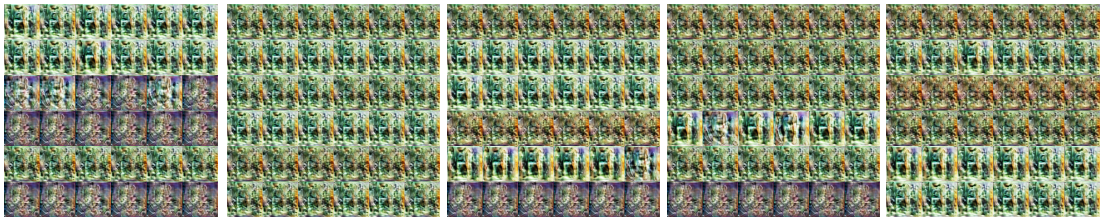

Figure 8: Generated-Images for each art-period in Testing

Despite the limited variation among the generated images, these initial results provide a basis for further exploration and improvement of the Music Visualization Network. Future work will focus on addressing the model collapse issue and enhancing the diversity and quality of the generated visual representations.

6 Conclusion

Overall, we have developed a Music Visualization Network (MVNet) that aims to establish a meaningful mapping between auditory and visual modalities. By integrating a Convolutional Audio Encoder and a Conditional Deep-Convolutional GAN, the MVNet enables the generation of visually coherent images that capture the style and characteristics of the input audio from various art periods. Furthermore, it offers a novel approach to exploring the semantic correlations between music and paintings throughout different time periods.

However, it is worth noting that the MVNet is not without its challenges and limitations. The size of the dataset, consisting of 9000 audio-image pairs, may not provide sufficient diversity and complexity to capture the full range of art-period styles. Additionally, the training time was restricted due to limited GPU availability, which may have affected the model's ability to learn more intricate representations. Last and most importantly, achieving a perfect balance between the generator and discriminator in the GAN framework is also a challenging task.

Future work could involve expanding the dataset with a more extensive collection of audio and image pairs (higher quality as well), allowing for a more comprehensive exploration of art-period styles. Furthermore, longer training times and additional fine-tuning techniques could potentially improve the model's performance and handle issues

related to model collapse.

References

[1] Cheng-Che Lee, Wan-Yi Lin, Yen-Ting Shih, Pei-Yi Patricia Kuo, and Li Su. Crossing you in style: Cross-modal style transfer from music to visual arts. *arXiv preprint*, 2021.

[2] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.