IMPERIAL

Imaudible

Image-based music generator

Author
Phantakorn Prarusudamkerng
CID: 01925990

Supervised by
Dr Lala Chiraag

Second Marker
Dr Mark Wheelhouse

A Thesis submitted in fulfillment of requirements for the degree of
Master of Engineering in Computing

Department of Computing
Imperial College London
2024

Abstract

In recent years, generative art has become increasingly popular with models such as DALL-E and Suno generating art when given free text. In this report, we extend this further by building generative model which takes image as input specifically for music generation. We propose two approach: Pipeline and End-to-End. With Pipeline, we connect two existing models together specifically an image captioning model and a text-based music generator. With End-to-End approach we swap out a joint text-audio embedding space for an image-audio embedding space from an existing text-based music generator to condition the model on image directly. We used different strategies to evaluate the models with varying degree of success. To illustrate real life use case, we also build an interactive 3D gallery - ImAudible gallery available at https://imaudible.netlify.app.

Contents

0

List of Acronyms

CDCML Cross-modal Deep Continuous Metric Learning

AMT Amazon's Mechanical Turk

SAT Show-Attend-Tell

LSTM Long short-term memory

KL Kullback–Leibler

RNN Recurrent Neural Network

BPTT Backpropagation through time

VQ Vector Quantization

RVQ Residual Vector Quantization

FAD Fréchet Audio Distance

VA Valence-Arousal

IMEMNet Image-Music-Emotion-Matching-Net

ReLU Rectified Linear Unit

MSE Mean Squared Error

FMA Free Music Archive

PCA Principal Component Analysis

DTW Dynamic Time Warping

MFCC Mel-Frequency Cepstrum Coefficient

1 Introduction

1.1 Motivation

Over these past years, Artificial Intelligence (AI) has become deeply rooted in today's society and played a significant role whether it be from targeted marketing[1] to identifying early stages of cancer cells[2]. It has integrated into our everyday lives and arguably, has had a positive impact. Recently, ChatGPT[3], a large language model which are model capable of performing various general-purpose tasks from creating essays to debugging code has been introduced. Building on top of it is DALL-E[4] which has advanced generative digital art to produce realistic images with distinct character when given suitable prompt. This has opened up new corners in the creative industry for newer ideas and creation.

We believe art is about personifying individual(s) perception; capturing emotions they have felt, or their creative expression through a medium. A shared medium that raise questions and connect people together. To be able to understand or empathise through a person's artwork, an individual may need a great understanding of one self[5]. By getting into the correct headspace or having the right context, it can help someone see things from the lenses of someone else. Having music accompany a piece of art such as a painting, can be one way to provide the right context.

1.2 Objectives

This paper propose a new creative tool. Given an image, the model generates music with similar mood/emotion to the input. We will consider the model a black box for the time being.

Figure 1.1: Blackbox model of image-based music generator

2 Background

2.1 Preliminaries

2.1.1 Regressive model

A Regressive model estimates a function describing the relationship between two or more variables. It can be seen as trying to fit a function onto an observed dataset.

2.1.2 Neural Network

Neural network[6] are models that make decision in a similar manner to how a human brain would. They are a form of non-linear regressive model made up by stacking hidden layers. A hidden layer is composed of mainly a fully-connected layer and an activation layer (introduces the non-linearity) stack on top of one another.

2.1.3 Fully-connected layer

A fully-connected layer[7] or a linear layer connects every input neuron to an output neuron. Each output neuron is a weighted sum of every input neuron. This is done mathematically by multiplying the input neuron x(usually represented as vector or a matrix) with a weight matrix

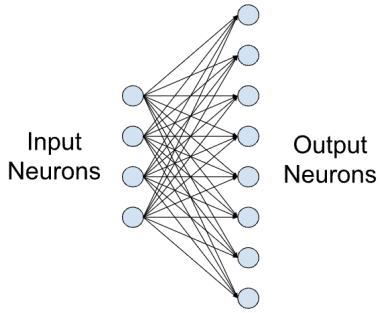


Figure 2.1: A linear layer

W. The output neuron y is then computed as:

$$y = Wx$$

2.1.4 Activation functions

Activation functions are functions which introduces non-linearity to neural network models. Without non-linearity, models are not able to learn complex features. Here, we introduce a few relevant activation function.

Softmax

Softmax takes in a vector and produce a probability distribution. It is defined as:

$$\sigma(\vec{z})_i = \frac{e^{z^i}}{\sum_{j=1}^{K} e^{z_j}}$$

where $\vec{z}$ is a vector of size K

Because of the output being a probability distribution summing to 1, this makes softmax an ideal function for probabilistic tasks such as emotion classification and word prediction task. With tasks such as image captioning, we may want the model to predict novel captions. To do so, we introduce randomness to our prediction so that our model do not predict generic word. We do this by introducing a temperature scaling constant τ. Our new function becomes:

$$\sigma(\vec{z})_i = \frac{e^{z^i/\tau}}{\sum_{j=1}^{K} e^{z_j/\tau}}$$

Tanh

Tanh is applied element-wise to a vector or matrix and defined as:

$$\sigma(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

It is used to determine whether a neuron is considered to activate or not.

Sigmoid

Similar to Tanh, we can also use Sigmoid which is defined as:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Rectified Linear Unit (ReLU)

Rectified Linear Unit (ReLU) allows only positive neuron to be considered as activated. It is defined as:

$$\sigma(z) = max(0, z)$$

Different activation function offers different benefits and trade-offs. We will omit this detail in this paper.

2.1.5 Gradient Descent

Gradient Descent is an optimization algorithm which tries to find an optimal weight that minimizes a cost (loss) function. Such optimization problem can be written as :

$$W^* = \min_W L(W)$$

where $L(\cdot)$ is the cost(loss) function we are optimising. Gradient Descent then updates the weights using the equation:

$$W_{new} = W_{old} - \alpha \frac{dL}{dW}$$

where $\frac{dL}{dW}$ is the derivative of the loss function w.r.t weight W and α is the step size/learning rate. Choosing an appropriate step size is essential to ensuring convergence.

2.1.6 Supervised learning

Supervised learning is a method in machine learning where each training data point is a pair consisting of an input and a ground truth. We use this training data to train models to predict outcomes and patterns by comparing their predictions against the ground truth values.

We can use loss function such as Mean Squared Error (MSE) to do the comparison and compute the loss. MSE computes the squared difference between each prediction and ground truth and average them over batch size.

2.1.7 Backpropagation

Backpropagation is a learning algorithm enabling neural network model to learn from previous mistakes. The model's loss is computed by comparing predictions to some truth value. Each model's parameter/layer is then updated using Gradient Descent with the derivative of the loss with respect to the parameter computed using the Chain Rule. The Chain Rule states that the derivative of a composite function can be expressed as the product of the derivatives of the composed functions. With models like RNNs, we have a recursive term; therefore, we also Backpropagate through time (BPTT).

2.1.8 ResNet

Residual Network[8] is a type of neural network model which stacks Residual blocks on top of one another. A residual block with input x outputs $f(x) + x$. By doing so, we only train our model to learn the residual of the network which prevents the model from straying away from learning the true function.

ResNet model comes in different sizes e.g ResNet-18, ResNet-34, ResNet-50 etc.

2.1.9 Recurrent Neural Network (RNN)

Recurrent Neural Network (RNN)[9] is another type of neural network characterized by the flow of information which has 2 inputs: our current input and one hidden state input h. This hidden state is passed back into the model. By unfolding through time we get that, at each time-step, we can compute our output y based on the current input and any information we have of the past (the hidden state).

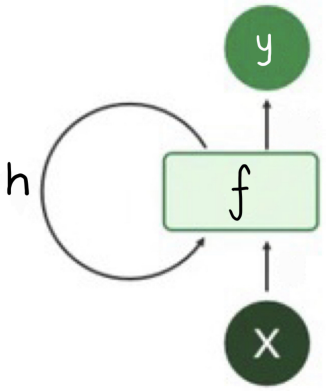


Figure 2.2: A single RNN unit

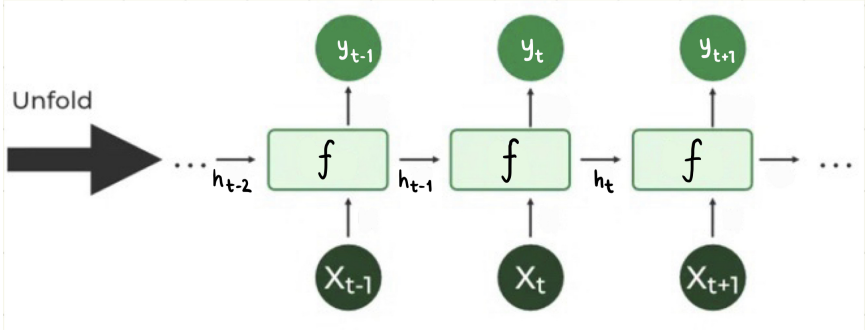


Figure 2.3: A single RNN unit unfolded through time

Mathematically, this can be written as

$$h_{t+1} = tanh(Wh_t + Ux_t)$$

where $W \in \mathbb{R}^{HxH}, h \in \mathbb{R}^{Hx1}, U \in \mathbb{R}^{HxV}, x \in \mathbb{R}^{Vx1}$

$$y_t = W_{hy}h_t + B_y$$

where $y \in \mathbb{R}^{Vx1}, W_{hy} \in \mathbb{R}^{VxH}, B \in \mathbb{R}^{Vx1}$

Here x is a one-hot encoding of the current input, U is the embedding layer weight, W is the hidden layer weight, W_{hy} is the output layer weight and B_y is the output layer bias, H is the hidden layer dimension and V is the vocabulary size of the model.

We can then use two of these units to form an encoder and decoder pair and train our model via supervised learning for task such as translating English to Spanish. During training, we first encode our English sentence through an encoder performing the logic we described above. Once we have our context vector (english sentence embeddings), we pass this as an input to our decoder along with a Start-of-Sequence token to predict the probability of the first spanish word. We can then compare this to our ground truth and compute our losses for Backpropagation through time (BPTT). During inference, once we predict the word of the current timestep, we pass this as input to the next timestep until an End-of-Sequence token is predicted. However when training, we have the ground truth therefore we can pass these instead of the predicted word since wrongly predicted word as input could lead to more wrongly predicted words and would slow down our model ability to learn.

2.1.10 Long short-term memory (LSTM)

A downside of RNN is that as you pass your hidden state through many time step, you start losing information from the earlier state. Long short-term memory (LSTM) mitigates this issue by introducing one additional component, a cell state representing the 'long-term memory' of the model. This allows the LSTM to dynamically focus on how much previous information should be retain. The long-term memory is calculated by the weighted sum of the current hidden state and the long-term memory of the previous timestep.

2.1.11 KL Divergence

Kullback–Leibler (KL) Divergence measures the difference between two probability distribution $p(x)$ and $q(x)$. It can be computed using the following formula:

$$D_{KL}(p(x)||q(x)) = \sum_{x \in X} p(x) \ln \frac{p(x)}{q(x)}$$

with $p(x) = q(x) \iff D_{KL}(p(x)||q(x)) = 0$ and $D_{KL}(p(x)||q(x)) \geq 0$

We will only consider the discrete form of KL Divergence.

2.1.12 Contrastive loss

Contrastive loss is a metric learning loss function used to train a neural network model. It is typically use to learn cross-modal(audio-text, image-text etc.) relationships by pulling similar embedding closer to one another and pushing opposites further apart.

Triplet margin loss

Triplet margin loss is a type of contrastive loss which pulls positive pairs closer to one another in the embedding space and pushes positive-negative pairs away.

This loss can only be done if we have discrete labelled datapoint (positive, negative).

2.1.13 Principal Component Analysis

Principal Component Analysis (PCA)[10] is a machine learning method used to reduce dimensionality of a dataset whilst preserving as much information as possible. We find directions (principal components) which the data varies the most with the first component having the highest variance following behind the second components, the third components and so on. Each components are orthogonal to one another. To find each component:

1. We find the center of our dataset and shift all our points such that the center is align to the origin

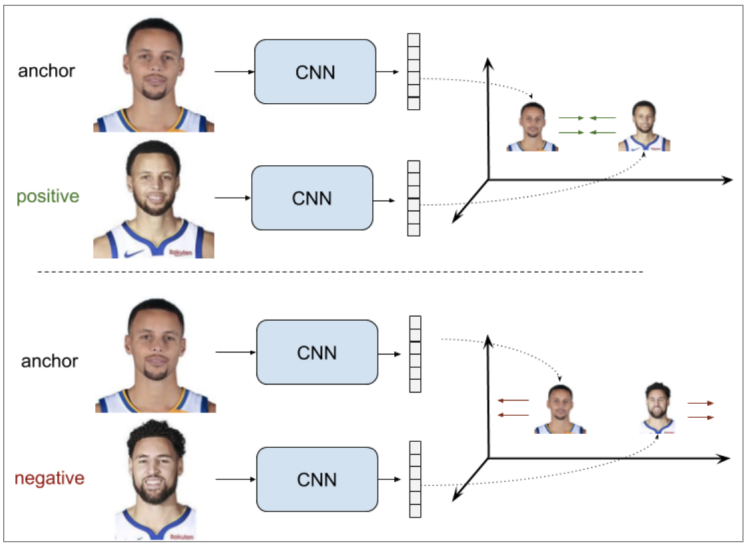


Figure 2.4: Triplet loss: pulls Stephen Curry-Stephen Curry image embeddings closer to one another and pushing Stephen Curry-Klay Thompson image embeddings away

2. We draw a line of best fit that maximizes the sum squared distance from the projected points (the orange dots in Figure 2.5) to the origin.

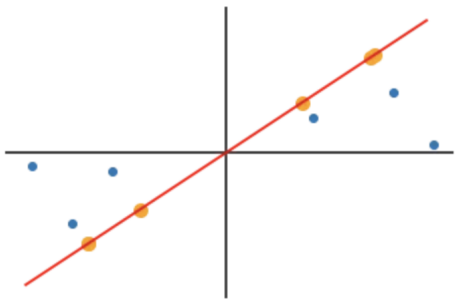


Figure 2.5: Each datapoint (blue) is projected onto the line (orange) and calculates the squared distance to the origin

Once we have enough components, we project our data onto these components treating them as axes. Simply put, the principal components are the axes which provides the best angle to visualise a dataset on. PCA is often used to visualise how a particular neural network behaves. This is usually done by extracting the hidden features from a particular layer of a model and perform PCA. For simple visualisation, we can reduce to 2 components and plot them on an XY plane.

2.1.14 Transformer

Transformer[11] is a type of neural network which consists of an encoder and a decoder similar to an RNN. Its main mechanism is Self-attention which computes a representation of each input based on the whole sequence. This representation allows the model to learn long-range dependencies that may not be as apparent to models like RNN because these model lose more information, the longer their sequence are.

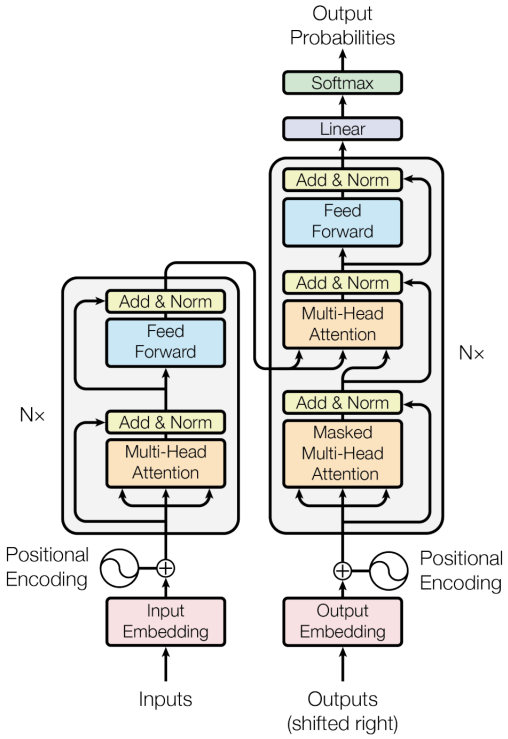


Figure 2.6: The Transformer - model architecture.

2.1.15 Cosine similarity

Cosine similarity[12] measures the similarity between two vectors by comparing their direction. Two vector going in the same direction will have a similarity score of 1. Two orthogonal vector will have a similarity of 0 and two vector going in the opposite direction have a similarity of -1. For vector A and B, the cosine similarity is computed as:

$$similarity(A, B) = cos(\theta) = \frac{A \cdot B}{||A||\ ||B||}$$

2.1.16 Pearson correlation coefficient

We can measure a linear correlation between two variable by using Pearson correlation coefficient. We do this by the mathematical formula:

$$r = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}}$$

where x,y are the two variable we wish to measure the correlation between. x_i,y_i is the ith value of the respective variable and $\bar{x}$,$\bar{y}$ is the mean value of the respective variable.

An r value of +1 signifies a positive correlation where if variable x increases so does y. An r value of -1 signifies a negative correlation where if variable x increases, y decreases. An r value of 0 signifies no correlation.

2.1.17 Min-max normalisation

In order to keep data in a sensible range while preserving their relationship, we can normalise the datapoint such that they lie in the range 0 and 1. Min-max normalisation is one of the normalisation technique which linearly transforms the data by using the minimum value and the range. The formula for min-max normalisation for a single datapoint x is:

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

where x_{max}, x_{min} is the maximum and minimum value from all datapoints respectively. This technique however is susceptible to outliers. We can perform feature clipping to clip whenever a value is below or above a certain threshold and then perform min-max normalisation.

2.1.18 One-hot encoding

A one-hot encoding refers to a vector in which only one entry of the vector is 1 while the rest are zeroes. This is typically use to encode a word allowing you to pass natural language (text, image, audio etc.) into mathematical models. An example is, given a vocabulary $V = \{Hello, Word, Go, Bob\}$, an encoding of the word Go would be $[0, 0, 1, 0]$ where the vector index refers to the position of each word in the vocabulary.

Figure 2.7: Examples from ArtEmis dataset; Each image can be emotionally ground truth which would generate different emotion-based caption

2.2 Related work

2.2.1 ArtEmis

ArtEmis[13] is a large-scale dataset consisting of visual artworks from WikiArt[14] accompanied by utterances and emotions from a discrete set of eight categorical emotion states. These discrete emotions are used following from previous studies [15] [16] [17] [18]. The dataset were constructed by asking at least 5 annotators per artwork the dominant emotions triggered within the emotion set. A ninth option being 'something else' is available if their answer did not belong in the set. These annotators were recruited via Amazon's Mechanical Turk (AMT) services. After this, they were asked to give a detailed explanation of their choice by free text that includes specific references to visual elements in the artwork. An analysis of the dataset - we refer to [13]. Examples of these artwork along with annotated emotion label and explanation are show in Figure 2.7.

Models trained with this dataset includes an image captioning speaker model capable of describing and reflecting emotions based on the visual elements. The speaker we will be focusing on is the Show-Attend-Tell (SAT)-based[19] approach speaker consisting of an image encoder specifically

a pretrained ResNet-34 model and a word/image attentive LSTM[20]. During inference, we can sample from the speaker by performing a greedy beam-search of size 5 and a soft-max temperature of 0.3. At each timestep, a probabilitiy distribution of words are predicted. To make a sentence, we could take the highest probable word at each timestep however this means giving up on any other sentences which may be a better translation as a whole. Beam search solves this by taking the top k most probable word and compute the probability of all word up to the current sequence then taking the top k highest probability from those at each timestep and repeating until the end.

Because we are able to describe and extract the emotional element of the images as text, this makes it an appropriate component for music generation when combined with text-based music generator publicly available such as Suno[21] and Udio[22]. ArtEmis also provides an image emotion classifier in case we opted for emotion grounding-based caption which is, you predict the emotion of the image and use that to generate a caption around. We do not opt for this option however it will become handy later in our **Analysis**.

2.2.2 Residual Vector Quantization (RVQ)

With a regular Vector Quantization (VQ), we are quantizing a vector into a single number by finding the closest entry of a predefined vector in a codebook of size K. Figure 2.8 illustrates this.

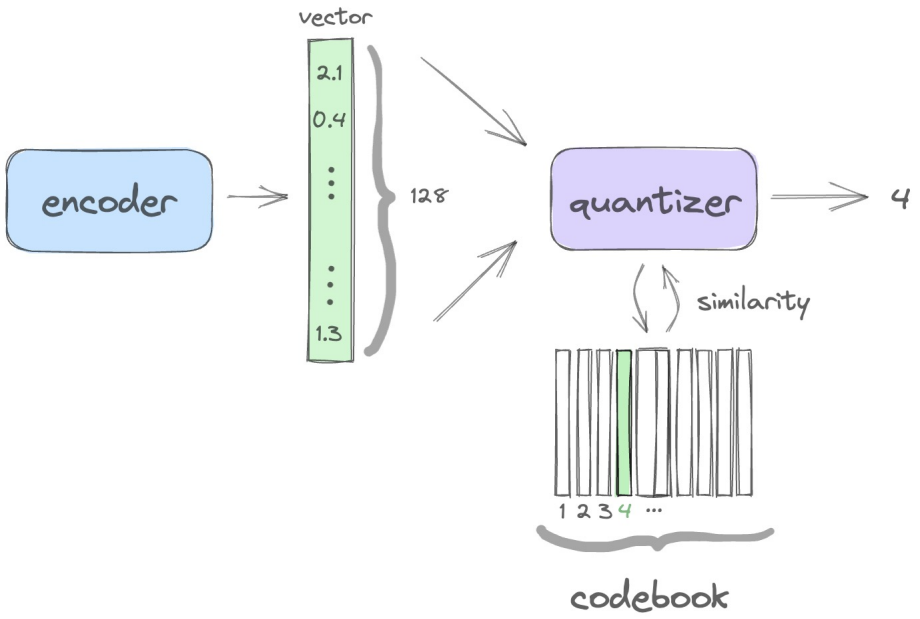


Figure 2.8: A regular VQ: the 128-dimension vector is most similar(closest) to the vector at index 4 in the codebook

However in order to reconstruct the encoding with minimal reconstruction error, it requires K to be infinitely large.

Residual Vector Quantization (RVQ)[23][24] addresses this issue by introducing additional codebooks (each of size K). By finding the closest entry from the first codebook, we can further quantize this by passing the residual(vector difference between the original vector and the vector approximant) into the second codebook therefore reducing reconstruction loss. We can repeat this process for as many additional codebook as we want. This also means the earlier codebooks capture broader audio and the latter capture the finer details. The concept is much better understood and illustrated in Figure 2.9

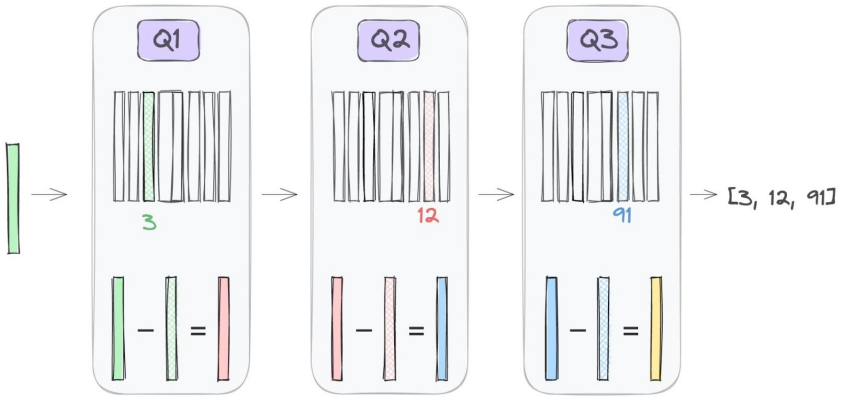


Figure 2.9: RVQ: we quantize our input vector by passing into 3 codebook

2.2.3 EnCodec

Neural compression utilises neural network to perform data size reduction of various data types (pixel(image), waveform(audio) etc.). By training the network to identify and learn pattern, the network is able to generate a compact representation of the data using the learnt pattern as basis unlike normal data compression which locates and remove statistical redundancies (Lossless compression[25]).

EnCodec[23] is a high fidelity neural audio compression capable of compressing audio waveforms and then reconstructing them with minimal losses. It consists of an encoder which encodes the audio waveforms into a fixed-size vector, an RVQ to quantized the audio encoding into tokens and a decoder which reconstructs the audio from the quantized token. The model is trained to reconstruct the original audio from the quantized audio and also learn the best way to represent vectors within each codebook.

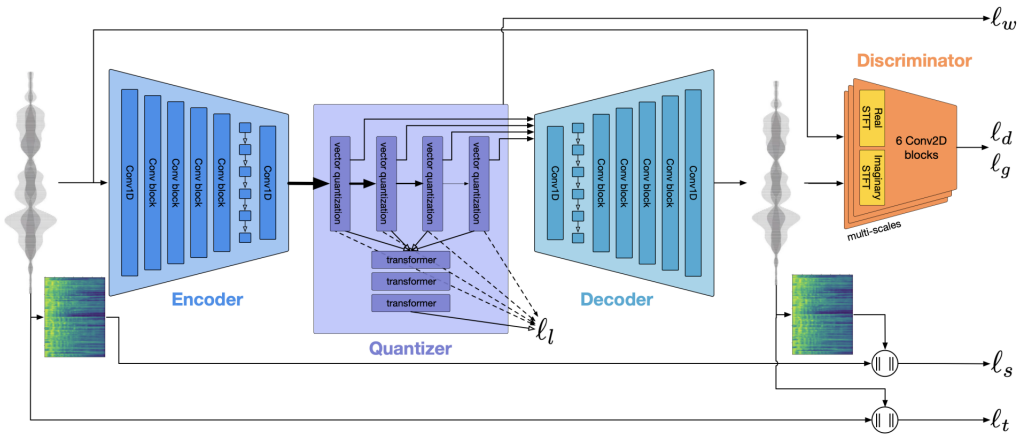


Figure 2.10: EnCodec architecture

This model is used in MusicGen[26] and the open-source version of MusicLM[27], Open-MusicLM[28] for audio tokenization by extracting the quantized audio from the RVQ.

2.2.4 MusicLM

AudioLM[29] is an audio generator capable of generating coherent, high-quality continuation of speech or audio with similar melody, harmony, tone and rhythm. MusicLM[27] builds on top of AudioLM by allowing the model to also condition on text. Figure 2.13 shows the architecture of MusicLM. We rely on tokens generated from 3 components namely MuLan[30], w2v-BERT[31] and SoundStream[32] (Figure 2.14).

Firstly, MusicLM relies on a joint text-audio embedding space MuLan. The model uses contrastive loss between the audio embedding and text embedding to align them in the same embedding space allowing the generated music to resemble the prompt as closely as possible. During inference, we make use of the joint embedding space to pass in our text prompt, quantized by an RVQ, to obtain MuLan text(audio) tokens.

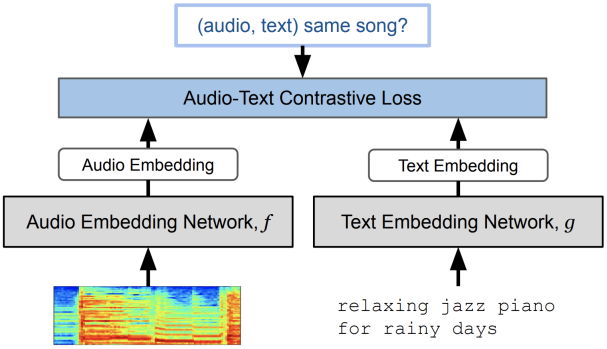


Figure 2.11: MuLan Learning Framework

Secondly, in order to generate long audio and ensuring coherency, MusicLM uses w2v-BERT which combines contrastive loss and masked prediction loss. The contrastive loss trains the model to predict the context of e.g. the context of the speech or the music timbre and the masked prediction loss trains the model to continue the audio with as much similarity as possible. Once quantized, we refer to these as semantic tokens.

Thirdly, MusicLM uses SoundStream, a neural audio compressor with a RVQ as a quantizer similar to EnCodec. We rely on the model to predict quantized audio which can be decoded into audio. We refer to these as acoustic tokens.

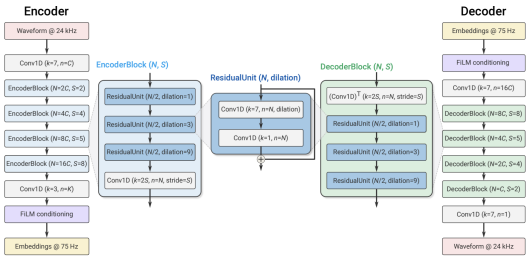


Figure 2.12: SoundStream encoder/decoder network architecture

The semantic and acoustic modelling both uses transformer-based decoder. The semantic modelling stage predicts the semantic token conditioned on MuLan text(audio) tokens meanwhile the acoustic modelling stage predicts 12 acoustic tokens for every 1 second of the music conditioned on both the MuLan audio tokens and the semantic tokens. Due to the residual nature of RVQ, each token cannot be predicted in parallel. To avoid the long sequence token, the acoustic stage is further split into a coarse and a fine modelling stage. The coarse stage predicts the first 4 acoustic

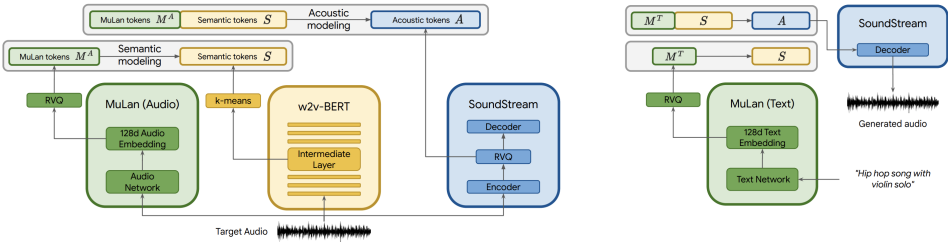


Figure 2.13: MusicLM architecture

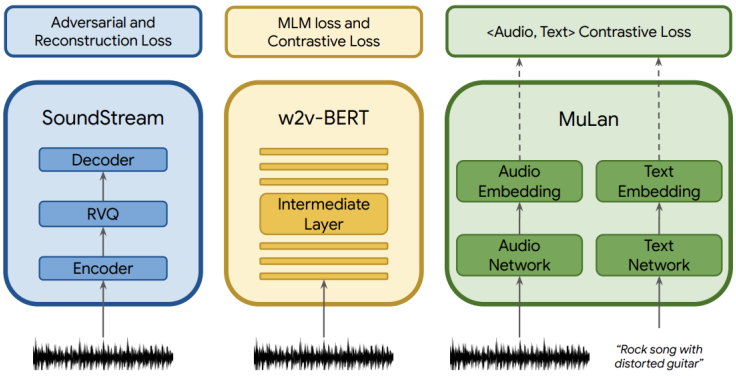


Figure 2.14: MusicLM components

tokens and the remaining 8 is predicted in the fine stage. Therefore there are 12 codebook, each acoustic token per book within SoundStream's RVQ.

Due to the time constraint, insufficient data and pretrained component being unavailable, we rely on an open-source implementation of MusicLM specifically we rely on the pretrained version of Zhang and Berry[28]. The difference between Open-MusicLM and the original MusicLM is that MuLan is replaced with CLAP[33] which also is a joint text-audio embedding space. SoundStream is replaced with EnCodec and w2v-BERT is replaced with MERT[34]. MERT also works similarly to w2v-BERT however it uses transformer-based encoder to compute a context representation from masked audio features.

2.2.5 MusicGen

Another text-based music generator we will be looking at is MusicGen[26]. MusicGen consists of an auto-regressive transformer-based decoder, an EnCodec for audio tokenization and a pretrained T5[35] text-based transformer to tokenize and encode our text.

The main difference between this and other text-based music generator like MusicLM is that,

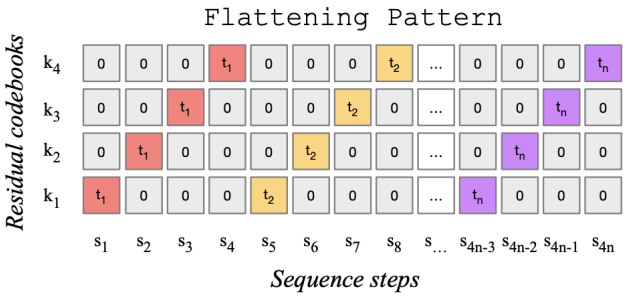


Figure 2.15: Flattening(Naive) interleaving pattern

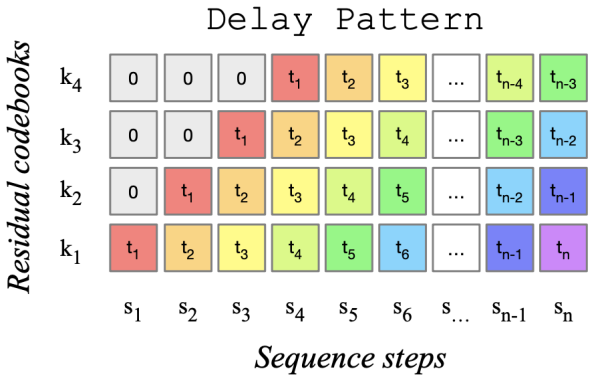


Figure 2.16: "Delay" interleaving pattern

with K codebook in our EnCodec to quantize each audio encoding. Due to the residual relationship between each codebook, we cannot predict these quantize token in parallel. A naive way to do this would be to pass the audio encoding sequentially as shown in Figure 2.15. However this increases the complexity of the model.

MusicGen improves this by adopting a "delay" interleaving pattern shown in Figure 2.16. At each timestep, the model predicts the quantized token for one codebook and interleaves them for the next codebook in the next timestep. By doing so, we are able to reduce the number of autoregressive timestep and speed up the model.

2.2.6 Valence-Arousal model

Emotional Psychology is the study of the mind and how emotion can trigger human reaction. Due to the complexity of emotion, trying to model them can be a difficult task. One of the way that has been done is through a Valence-Arousal (VA) Model. A VA model[36] can be seen as a 2D model where the x-axis represents the Valence and the y-axis represents the Arousal. Valence describes the extent to which emotion is positive-negative whereas Arousal describes the intensity

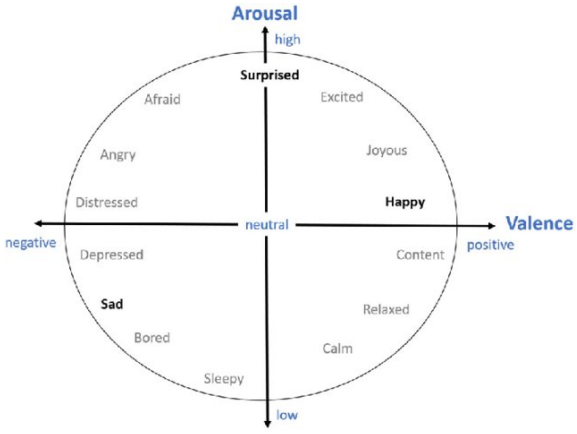


Figure 2.17: Valence-Arousal model

of the emotion. Valence and Arousal are subjective and is user-rated based on a scale of 0-9[37]. Although these ratings are subjective, there are studies which shows correlation between subjective ratings of Valence and activity in the sympathetic nervous system[38]. EEG[39] studies, a test measuring electrical activity in the brain, have also shown correlation between cerebral activation and subjective ratings of Arousal[38]. To correlate similarity between two points on the VA space, we can use euclidean distance to do so.

2.2.7 IMEMNet

Image-Music-Emotion-Matching-Net (IMEMNet) is a dataset introduced in Cross-modal Deep Continuous Metric Learning (CDCML) paper which consists of image-music pairs and their corresponding similarity score. The images are taken from 3 dataset being IAPS[40], NAPS[41] and EMOTIC[42]. The music is taken from DEAM[43] dataset. The similarity score is mathematically defined as:

$$S(I_i, M_i) = exp(-\frac{d(y^{I_i}, y^{M_i})}{\sigma_n^m}), i = 1...n, j = 1...m$$

where $d(\cdot,\cdot)$ is the euclidean distance function, y^{I_i}, y^{M_i} is the VA label of image i and music i. σ_n^m is the average euclidean distance between all image and music clips. To avoid the exponential blow up, we approximate σ_n^m by sampling 50 images per music clip. Within the 50 images, we choose 10 images with the highest similarity score to the music clip, 10 images with lowest similarity score and 30 images are randomly sampled. We then randomly sample 5(10%) of the 50 images and average their similarity score.

2.2.8 Cross-modal Deep Continuous Metric Learning (CDCML)

CDCML[44] is a joint image-music embedding space trained using IMEMNet dataset to project image and music into the same embedding space by preserving both the cross-modal and single-modal emotion relationships using metrics inspired by log-ratio loss [45].

The model aims to achieve three things:

1. Given a pair of image and music, predict the similarity score
2. Given an image or/and music, predict its VA label
3. An optimised embedding space where the cross-modal(image-music) and single-modal(image-image, music-music) relationship is preserved.

Figure 2.18 illustrates the CDCML framework

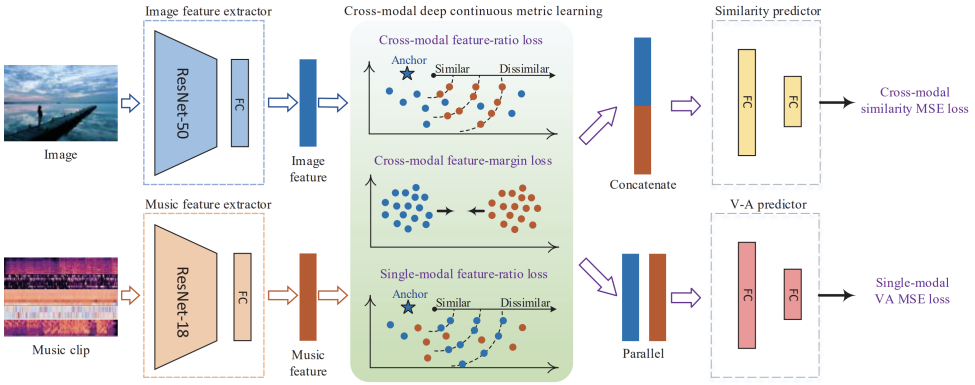


Figure 2.18: Top: Image branch, Bottom: Music branch

The features of the image are extracted by passing into ResNet-50, dropping the classification layer and adding an additional fully-connected layer to obtain a 512-dimensional embedding. Note that each image is resized to dimension 224×224×3 before passing into ResNet-50.

Similarly, the features of the music clip can be extracted by passing into ResNet-18, dropping the classification layer and adding an additional fully-connected layer to obtain 512-dimensional embedding. The music clips are pre-processed by extracting [193x87]-d music features. Because ResNet take in a 3-dimensional input $CxHxW$ where channel(C) = 3 and H,W is the width and height of the image, we tile the feature to obtain a feature matrix of dimension 3x193x87 before passing into the model.

$L_{CFR}, L_{CFM}, L_{SFR\_I}, L_{SFR\_M}$ are the CDCML metrics that [44] introduced to optimise the shared space. We first introduce any mathematical notations used in the metric down below

- $D(\cdot,\cdot)$ means the squared Euclidean distance
- $S(\cdot,\cdot)$ denotes the similarity function
- I_i denotes the ith image
- M_i denotes the ith music
- f^{I_i} are the feature extracted from I_i
- f^{M_i} are the feature extracted from M_i
- N is the batch size
- α is a threshold to manipulate the maximum tolerable distance
- $[\cdot]_+ = max(0,\cdot)$
- Cross-modal Feature-Ratio Loss

$$L_{CFR} = \sum_{i=1}^{N}\{log\frac{D(f^{I_i}, f^{M_i})}{D(f^{I_i}, f^{M_j})} - log\frac{S(I_i, M_i)}{S(I_i, M_j)}\}^2 + \sum_{i=1}^{N}\{log\frac{D(f^{M_i}, f^{I_i})}{D(f^{M_i}, f^{I_j})} - log\frac{S(M_i, I_i)}{S(M_i, I_j)}\}^2$$

  where $i \neq j$.

- Cross-modal Feature-Margin loss

$$L_{CFM} = \sum_{i=1}^{N}[\|f^{I_i} - f^{M_i}\|_2 - \alpha]_+$$

  We empirically chose $\alpha = 0.2$

- Single-modal Feature-Ratio loss

$$L_{SFR\_I} = \sum_{i=1}^{N}\{log\frac{D(f^{I_i}, f^{I_j})}{D(f^{I_i}, f^{I_k})} - log\frac{D(y^{I_i}, y^{I_j})}{D(y^{I_i}, y^{I_k})}\}^2$$

$$L_{SFR\_M} = \sum_{i=1}^{N}\{log\frac{D(f^{M_i}, f^{M_j})}{D(f^{M_i}, f^{M_k})} - log\frac{D(y^{M_i}, y^{M_j})}{D(y^{M_i}, y^{M_k})}\}^2$$

  where $i \neq j \neq k$.

Once we have our image and music features we passed them into two predictors :

1. Similarity predictor - predicts the similarities between a pair of image and music clip. The Cross-modal similarity MSE loss can be computed from the equation

$$L_{Sim} = \frac{1}{N}\sum_{i=1}^{N}(S(I_i, M_i) - \hat{S}(I_i, M_i))^2$$

2. VA predictor - predicts the VA values between a pair of image and music clip. The Single-modal VA MSE loss can be computed from the equation

$$L_{IVA} = \frac{1}{n}\sum_{i=1}^{N}(y^{I_i} - \hat{y}^{I_i})^2$$

$$L_{MVA} = \frac{1}{m}\sum_{j=1}^{N}(y^{M_j} - \hat{y}^{M_j})^2$$

Each predictor is composed of 3 fully-connected layer. Each layer contains a BatchNorm layer which re-centers and re-scale the hidden state to speed up training and a ReLU activation function except for the last layer which uses a Sigmoid activation function.

The joint embedding space can then be optimised by computing the sum of each metric.

$$L_{CDCML} = (L_{Sim} + L_{IVA} + L_{MVA}) + (L_{CFR} + L_{CFM} + L_{SFR\_I} + L_{SFR\_M})$$

2.2.9 Log-ratio loss

With continuous label like VA, triplet loss can no longer help us. [45] proposes a method to perform the optimisation by using log-ratio loss. The loss approximates the ratio of the euclidean distance of features embedding by the ratio between the data truth value. This is mathematically written as :

$$l_{lr}(a, i, j) = \{log\frac{D(f_a, f_i)}{D(f_a, f_j)} - log\frac{D(y_a, y_i)}{D(y_a, y_j)}\}^2$$

where (a,i,j) is the triplet of an anchor and two other datapoint, $D(\cdot, \cdot)$ is the squared euclidean distance, f_i is the embedding of datapoint i and y_i is the continuous label of datapoint i. The log-ratio loss allows the embedding space to reflect degrees of label similarity and as well as its rank (how far apart within the same vector direction they are).

2.2.10 Fréchet Audio Distance (FAD)

Fréchet Audio Distance (FAD)[46] is an audio quality metric used to evaluate how good a piece of audio is compared to reference audios. This reference audio is usually a clean, studio recording of music meaning FAD measures how "real" the music sound. Usually when training a music generation model, you would use the training audio as reference thus making it a reference-free metric. In case of inaccessible training data, we may use any kind of audio which we believe is of ideal quality to what we want our model to generate.

To calculate FAD, we pass our audio through a pretrained VGGish[47] model, an audio classifier trained on AudioSet[48]. We extract the activations prior to the classification layer to obtain the 128-dimension audio embedding. We compute the statistics from each source by computing the mean and covariance matrix and finally compare the two statistics by calculating the Fréchet Distance between the two curve also referred to as FAD score. Fréchet Distance between two multivariate Gaussians (statistics) $N_e(\mu_e, \Sigma_e)$ and $N_b(\mu_b, \Sigma_b)$ can be mathematically computed using the formula[49]:

$$\mathbf{F}(N_e, N_b) = ||\mu_e - \mu_b||^2 + tr(\Sigma_e + \Sigma_b + 2\sqrt{\Sigma_e \Sigma_b})$$

where μ_b,Σ_b is the multivariate Gaussian distribution mean and covariance matrix from our reference(background) audio and μ_e,Σ_e is the multivariate Gaussian distribution mean and covariance matrix from our generated audio. The lower our FAD score is, the more similar our audio quality is to the reference.

2.2.11 Fourier Transform

Fourier Transform[50] is a function which takes an audio(time-series) as input and decomposes it into constituent frequencies. Intuitively, we traverse the audio and draw based on its amplitude around a graph, wrapping around whenever we complete a full circle. We can adjust the frequency at which we wrap around the circle. At each frequency, we compute the center of mass of the graph, this mass will usually hover around 0 however when the frequency matches the frequency present in the audio, it causes a spike in our center of mass graph (Figure 2.20. Mathematically, the center of mass at frequency f can be mathematically written as:

$$\hat{g}(f) = \int_{t_1}^{t_2} g(t)e^{-2\pi ift}dt$$

where $g(t)e^{-2\pi ift}$ can be seen as the graph of the shape being drawn, wrapping around at frequency f and the integral computes the center of mass from the graph. Finally, we do this for all audible frequencies which would get us all the constituent frequencies from the audio.

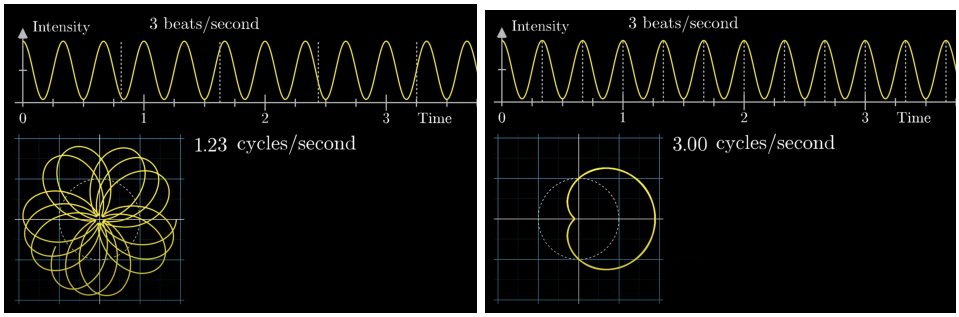


Figure 2.19: Changing the frequency at which the graph is wrapping around itself changes the shape that is drawn

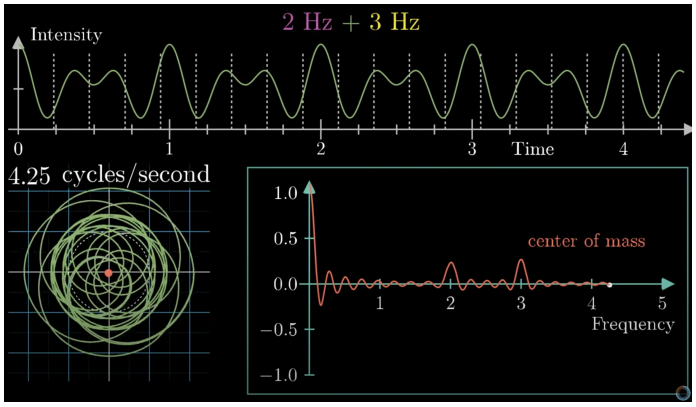


Figure 2.20: Top: Audio; Bottom left: The image being drawn by traversing the audio; Bottom right: Graph of center of mass as we increase the frequency at which we are wrapping the graph around itself

2.2.12 Mel-Frequency Cepstrum Coefficient

A Mel-Frequency Cepstrum Coefficient (MFCC)[51] is an audio feature which can be extracted from an audio by performing a Discrete Fourier Transform similar to a Fourier Transform but on a discrete set of frequencies. Applying log to our output, apply mel-scaling which scales the frequencies to be equidistant to perceivable sound and finally perform a Discrete Cosine Transform which decomposes and decorrelate different mel-bands into coefficients. Taking the first 12-13 coefficients gives us information such as the spectral envelope which are important cues for identification of an

instruments or voices[52]. Similarly, we can also extract the first 20 coefficients which in part, can be use to represent timbre[53]- the perceptual sound quality or the ”color” of sound of our audio. There are studies which shows correlation between musical timbre and emotion[54].

2

2.2.13 Dynamic Time Warping

We can measure the similarity between two time-series by using Dynamic Time Warping. Dynamic Time Warping (DTW)[55] is a dynamic-programming algorithm which computes the optimal(shortest) path between two time-series by computing the distance between all possible points between them. This makes it good for unaligned audios or audios with varying speed. The optimal path from time-series n at time i time-series m at time j (n_i, m_j) can be mathematically written as:

$$D_{min}(n_i, m_j) = \min_{n_{i-1}, m_{j-1}} D(n_{i-1}, m_{j-1}) + d(n_i, m_j | n_{i-1}, m_{j-1})$$

where $d(\cdot)$ is the distance measure typically the euclidean distance. The average path cost would be:

$$\frac{\sum_{i,j} D(n_i, m_j)}{N}$$

where N is the total step to go from start to finish and i,j represents the optimal path.

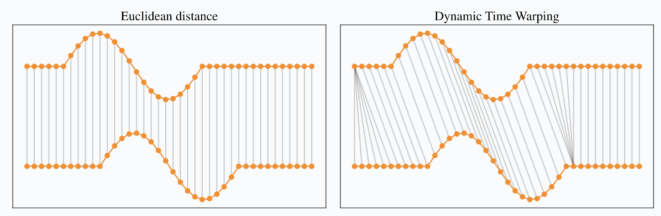


Figure 2.21: Measuring similarity between two time-series; Left: Euclidean distance; Right: Dynamic Time Warping

Since audio features (including MFCC) are represented in time-series, we can use DTW to measure the similarity between them which should give us the similarity in music timbre to an extent.

3 Models

3.1 Models

Here we propose two implementation details of the blackbox shown in 1.1.

3.1.1 Pipeline approach

Firstly, a pipeline approach which consists of two parts. An image captioning model which converts image into text and a text-based music generator which converts text into music. Note here that our image captioning model must also be able to describe emotional elements of the image as well in order to propagate these elements to the generated music.

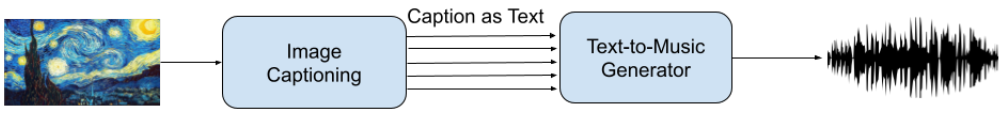


Figure 3.1: Pipeline approach of our music generator

Pipeline 1.0 (Open-MusicLM)

The image captioning model we will be using is the ArtEmis model. For the music generator, we use Open-MusicLM. Both of these are pretrained.

Pipeline 2.0 (MusicGen)

We switch out Open-MusicLM for MusicGen. The reason for this is MusicGen has been trained much more rigorously than Open-MusicLM which should give us more plausible audio and by having more than one pipeline model, we can also compare the two. We are going to be using the 'musicgen-medium' pretrained MusicGen with 1.5B parameters for our text-music generation which can be found on Hugging Face[56].

3.1.2 End-to-End approach

Open-MusicLM relies on a joint text-audio embedding space to generate music conditioned on text. We will be swapping this space out for a joint image-audio embedding space. Specifically we use CDCML to do so and will have to train these ourselves. Figure 2.13 shows the original architecture and Figure 3.2 shows our novel approach with the joint space replaced.

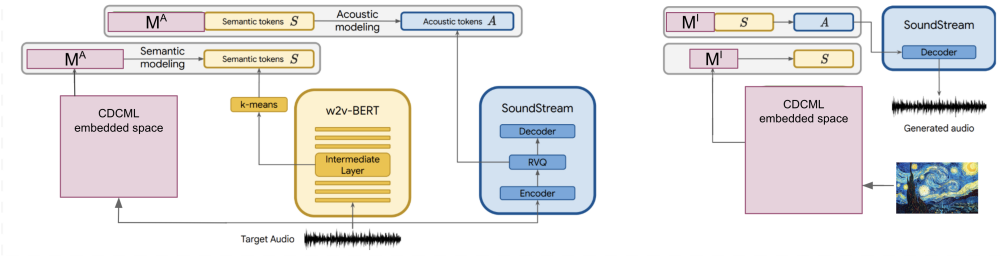


Figure 3.2: We swap joint text-music embedding space for a joint image-music

3.2 IMEMNet

In order to train CDCML, we first analyse IMEMNet dataset. We filter out any images belonging in the IAPS[40] since these dataset is no longer accessible. After filtering, we do a few preprocessing steps:

- An image may have multiple annotators thereby having more than one VA annotated label. We simplify these by averaging them.
- Certain datapoint do not have any VA annotation. We simply filter these out.
- We normalise both images and music clips VA annotation using min-max normalisation as also done in the original paper[44].

- We remove any unplayable audio clips. We run into these as we start processing the dataset.

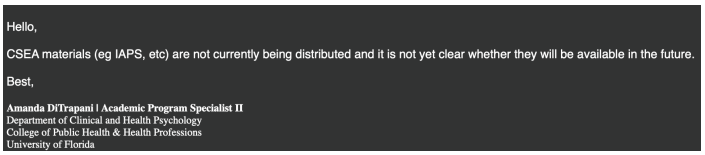


Figure 3.3: Response from the University of Florida department of Clinical and Health Psychology on the accessibility of IAPS dataset

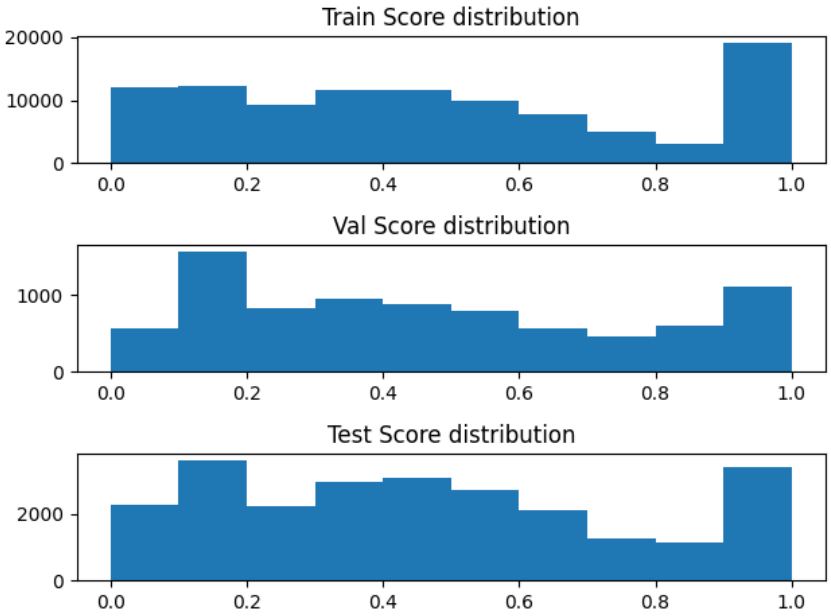


Figure 3.4: Similarity score distribution graph

Figure 3.4 shows the distribution of similarity scores across different training splits. We see that these patterns are similar across all splits therefore validation and evaluation should give us a good representation of when training with the training split.

Figure 3.5 shows the normalised VA annotation across all splits. We see that the values are quite spread out so should allow the model to learn different patterns for different extremities. However we also see that there are areas with sparse datapoint in both image and music(audio) VA space (low Valence-low Arousal, high Valence-high Arousal in image VA space and low Valence-high Arousal, high Valence-low Arousal in audio VA space) which means the model may not learn thoroughly enough to cover all spaces.

In the dataset, the ground truth similarity is calculated from the equation mentioned in **Background** section. In order to find reference music clips which we can use for our evaluation, we try to find a threshold in similarity score which we can deem as ”matching” image-music pairs. We

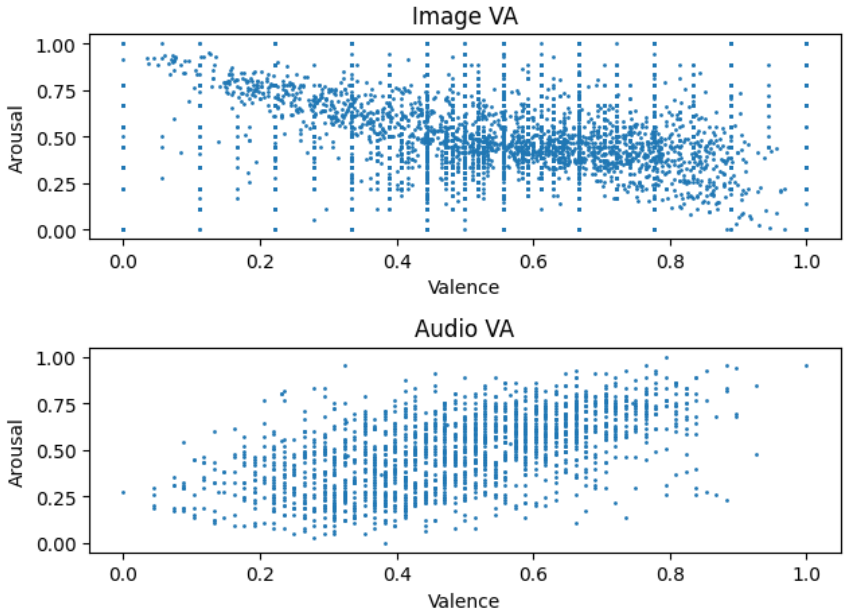


Figure 3.5: VA distribution graph

decided to plot similarity scores against the euclidean distance of their VA space. Figure 3.6 shows the result.

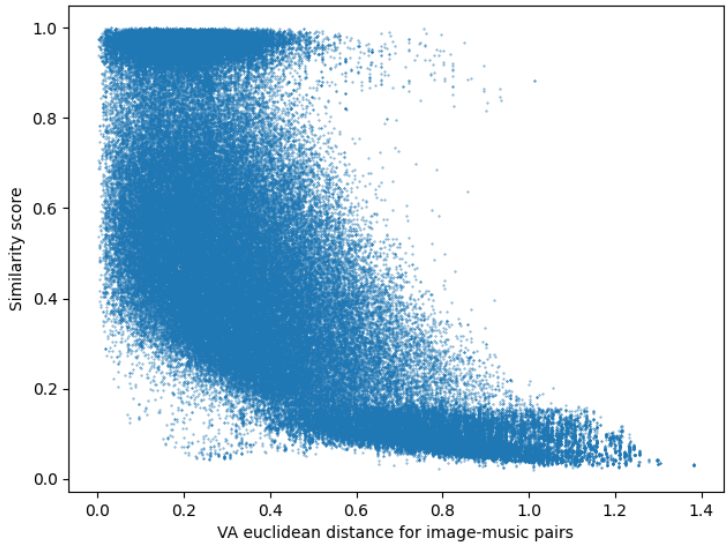


Figure 3.6: Similarity score-VA euclidean distance between image-music pair graph

We observe a cluster in the region as seen in Figure 3.7 and treat datapoint within this cluster as ”matching” image-music pairs. We decided on a threshold above 0.9 for the similarity score and and VA euclidean distance below 0.5. In this new subset, we have a total of 23405 image-music pairs. We will refer to these as Audible-Visual dataset. Due to time, we subsample 50 of these randomly to use in our evaluation. We will refer to these 50 pairs as Audible-Visual-50.

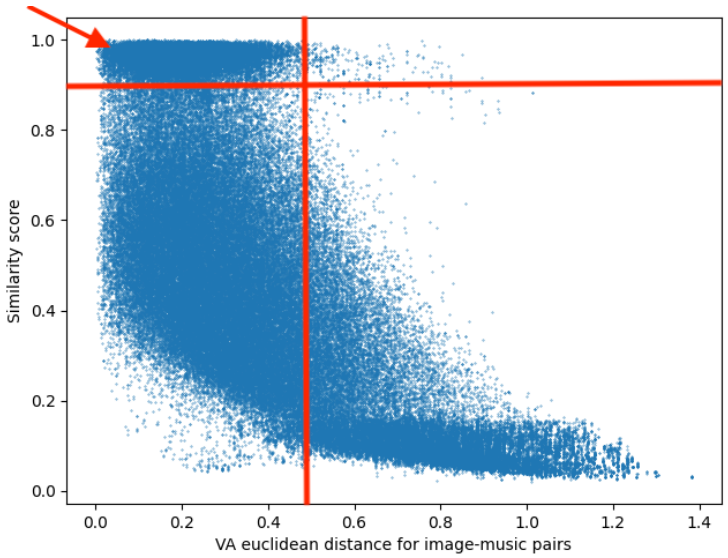


Figure 3.7: Thresholding datapoints with similarity score > 0.9 and VA euclidean distance < 0.5

Qualitatively, we also observe that the images are real life photos, most of which are snapshots. These images have a different style compared to ArtEmis dataset therefore passing in the images to the image captioning speaker may not output an accurate description.

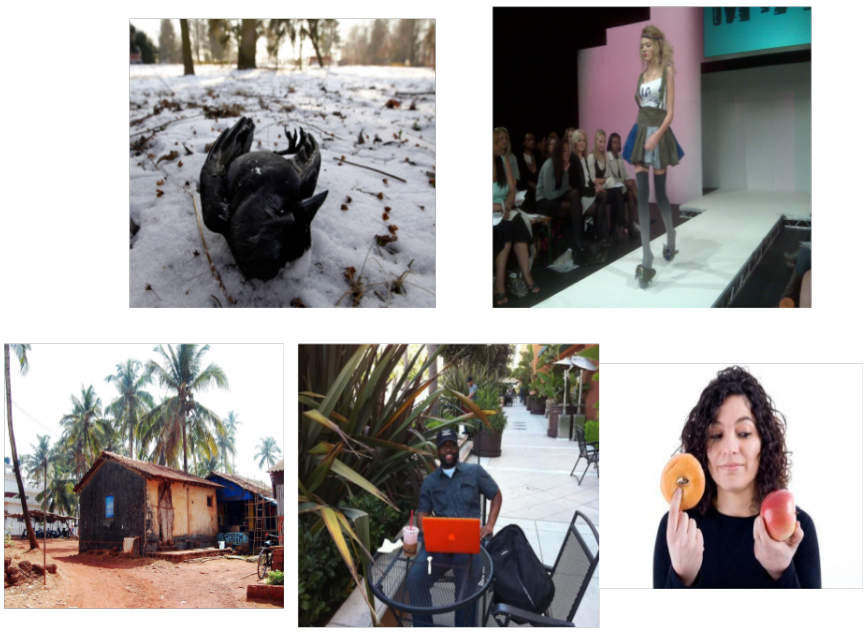


Figure 3.8: Image examples from Audio-Visual-50

4 Experiments

4.1 Experiments

4.1.1 Audio quality

Firstly, we evaluate the generated music on the quality itself specifically we use the FAD metric. This means the music may not necessarily follow the text it is conditioned on. Another reason we decided to settle for a pretrained music generator is because the amount of time and resources required to train the model such that it would generate plausible audio and the lack of high quality training audio. Both MusicLM (including Open-MusicLM) and MusicGen uses an internal dataset to do so.

To evaluate FAD, we require reference music clips. We initially used MusicCaps[57], a text-music pairs dataset as our reference music however whilst going through the dataset, we observe that certain music clips are not ”clean” i.e. they are not a clean audio recording (example: https://www.youtube.com/watch?v=_1woPC5HWSg). This is not ideal as reference music, so we switched to Free Music Archive (FMA)[58] dataset but also soon found out they contain lyrics which is not ideal since our generated music are instrumentals only. Finally we settled on randomly sampling 100 instrumentals from FMA dataset instrumental-genre section. We then randomly crop 16 seconds per music clip and follow [46] by sampling 1 second clip for every 500 millisecond. In total, we have 3100 1-second music clips to be use for reference music. Fortunately, there is a package[59] which computes FAD for us. All we need is to specify the path for both generated and reference music clips.

We evaluated FAD score of our generated music by passing images from Audible-Visual-50 dataset to Pipeline 1.0 and Pipeline 2.0 as input. Additionally, we also compare generative music quality to Suno[21] and Udio[22]. Since we rely on ArtEmis to output text prompt to feed into our text-based music generator, we can also feed this prompt to Suno and Udio to get their generative music. We obtain 50 music clips from Udio(Beta) and 50 from Suno(v3). Note that without any fine-tuning to Open-MusicLM, we can expect the audio quality from Pipeline 1.0 and End-to-End model to be similar in quality. This is because both model use the same pretrained decoder to generate the music.

4.1.2 Emotional suitability

Timbre similarity

To measure musical timbre similarity between two audio, we can extract the MFCC of each audio and compute the shortest path between them using DTW.

First, we randomly sample 50 IMEMNet datapoints, each point consisting of an image-music pair and their similarity score. Second, we pass the image through our model which gives us generated music. We now have a generated music and a "ground truth" music that we can compare. We extract the MFCC from both music clips and then compute the DTW distance(shortest path).

A normal DTW algorithm has time complexity of $O(n^2)$. This is only ideal for smaller time-series hence we approximate DTW using FastDTW[60]. First, we shrink the time series into smaller resolutions of different scale. Secondly, starting from the smallest time series, we find the DTW distance and use the optimal path found as basis by only considering points along the upscale path to refine the path on bigger resolution. This approximation gives near-optimal result in $O(n)$ time. Figure 4.1 illustrates FastDTW algorithm.

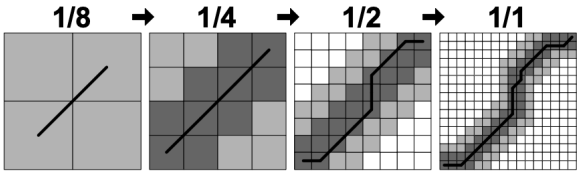


Figure 4.1: DTW table at different scale

Finally, we plot a similarity score against DTW distance graph to observe a relationship between the two metric and whether our model managed to generate music similar to the ground truth for higher similarity score.

Audio Genre as emotion

Another way to measure similarity between audio is by adopting a proxy method similar to [27], [61] and [62]. These paper uses a audio classifier trained on AudioSet[48] which classifies audio events e.g whether the audio contains sound of musical instrument, spray sound, gargling sound, vehicle sound etc. Therefore by feeding text prompt such as *The low quality recording features a renaissance music that consists of soft wooden percussion*, which is an annotation of a ground truth music clip, you could measure how suitable the generated music is by computing the KL Divergence between their classification output. In our case of using image as input, we focus more on how much emotion from the image is translated to the music. This type of classifier may not be appropriate for the experiment.

Instead we use SeyedAli's music genre classifier[63] which classifies what type of music it is based on music genre. Due to the subjectivity of emotions, it poses a challenge to test how suitable our generated music is to our image input without making certain assumptions. Here, we make an assumption that different music genre invoke different emotion to an extent. We use Audible-Visual-50 dataset to generate music from image and then pass both generated music and ground truth music to the classifier to obtain a genre probability distribution. We can compare these two distribution using KL Divergence. A small KL value should suggest the generated music invoke similar emotions to our ground truth therefore able to capture the emotional elements of the image well. We average KL value across all 50 pairs.

Caption semantics

We also experimented with stopwords removal on Pipeline 2.0. Stopwords are words which are deemed insignificant and do not carry any useful information. We can remove these from our caption to improve succintness and in turn generate more succinct music. We also remove the words *look, looks, like* which seems to be a common pattern found from generating caption using ArtEmis. We do this and perform the same experiment as 4.1.2 to see whether this had any affect on our performance. Again, we average KL value across all pairs from Audible-Visual-50.

4.1.3 Qualitative result

As emotions are subjective, we also ask people whether the generated music suited the image. We randomly chose 15 images some of which are paintings from the internet and some which are

photos taken by our friends and families. We generated their caption, applied stopword removal and generated the corresponding music. 5 images from the internet were use to conduct our survey. Firstly, we asked listeners to listen to the generated audio. Secondly, we ask them to rate how fitting the caption is from a scale of 1-10. Finally, we ask how fitting the image is from a scale of 1-10. Figure 4.2 shows the 5 images.

Figure 4.2: From top left to bottom right: Spiderman, Distracted Boyfriend, Starry Night, Woman in Blue, The Kiss

5

Evaluation & Analysis

5.1 Result analysis

5.1.1 CDCML Convergence

We train CDCML following [44] however due to the time constrain we only use 10k out of 101,900 training datapoints randomly selected. The loss curve is shown in Figure A.1.

From Figure A.1 (Appendix A), we see that the loss curve does not follow a negative(decreasing) trend suggesting that nothing was learnt during training. To investigate whether this was the case, we randomly took 640 image-music training data pairs from our 10k training datapoints and perform the following:

1. We pass the image through ArtEmis' image emotion classifier to predict the emotion.
2. We pass the paired music through SeyedAli's music genre classifier to predict the music genre.
3. Within each modal(image/audio), each datapoint is passed through CDCML to obtain a 512-dimension embedding .
4. We perform PCA to reduce a 512-dimensional vector into a 2-dimension vector.
5. We plot image and music on a separate graph along with its predicted labels.

The results are shown in Figure 5.1.

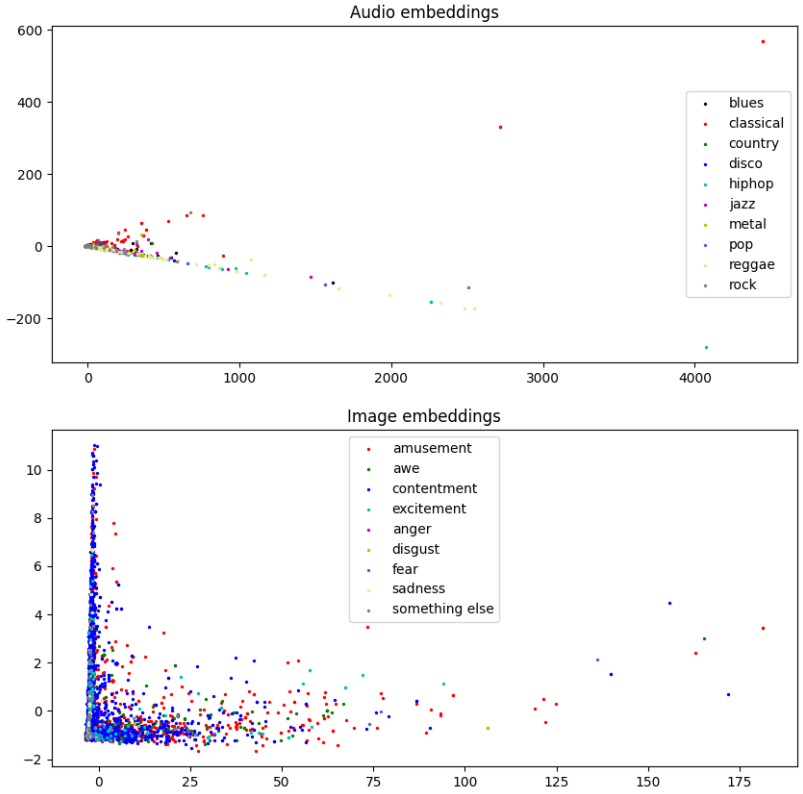


Figure 5.1: Music and Image embedding space along with their predicted labels

We observe from Figure5.1 that the spaces were not random however these pattern was not apparent enough to infer any information from. We also see from the image embedding space that many datapoints were classified as emotion label: *contentment* and plotted the image emotion distribution by passing each image into ArtEmis' image emotion classifier (Figure 5.2). The emotions were imbalance with *contentment* having the highest count. Because of this image emotion bias and the smaller subset we used, there was not enough information for the model to learn which attributed to why the loss curve was not converging. We assembled our end-to-end model using this and tried generating audio however the audio were generally noise hence further evaluation of this generated audio is trivial and we omit this from any of our further evaluation.

5.1.2 Fréchet Audio Distance (Audio Quality)

Table 5.1 shows FAD score for all reference music clips based on experiment 4.1.1.

Based on our result from Table 5.1, we see that similar FAD score pattern appear across all three different reference music. Pipeline 2.0 (MusicGen) performs the best in all three background sources suggesting a consistent audio quality throughout the different sources. When listening to

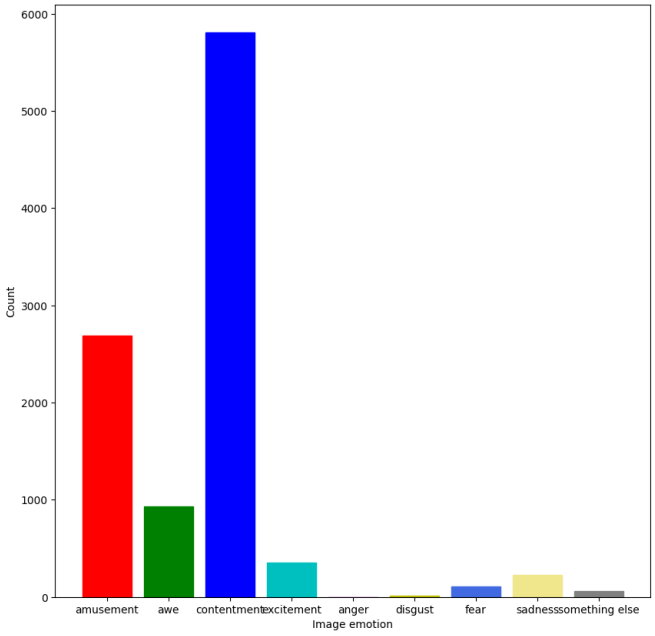


Figure 5.2: Image emotion distribution

-	**FAD score**		
Models	MusicCaps	FMA	FMA Instrumentals
Pipeline 1.0 (Open-MusicLM≈End-to-End)	13.278	11.863	11.416
Pipeline 2.0 (MusicGen)	**7.703**	**4.981**	**2.397**
Udio	7.915	5.234	3.209
Suno	12.643	5.8091	3.966

Table 5.1: FAD score evaluated on different models

the audios, we perceived Pipeline 2.0 to contain raspy noise which explained why it achieved the lowest FAD score on MusicCaps which contained noisy audio. These noises were not apparent in our reference audios, Suno's and Udio's which is why they may have scored higher here. Suno's score on MusicCaps was the largest suggesting that it does not have noisy audio quality in them and adheres the same pattern similar to other models across FMA and FMA Instrumentals signifying it has more good quality than bad which is what we hear.

We also pass all 1 second music clip from FMA Instrumentals into SeyedAli's music genre classifier and we plotted their genre distribution. From Figure 5.3, we see that the distribution are quite imbalance and due to this, the statistics of the reference audio may have been skewed so it does not represent "clean" music accurately. This could have explained the bigger differences in FAD score of Pipeline 2.0, Udio and Suno on FMA Instrumentals.

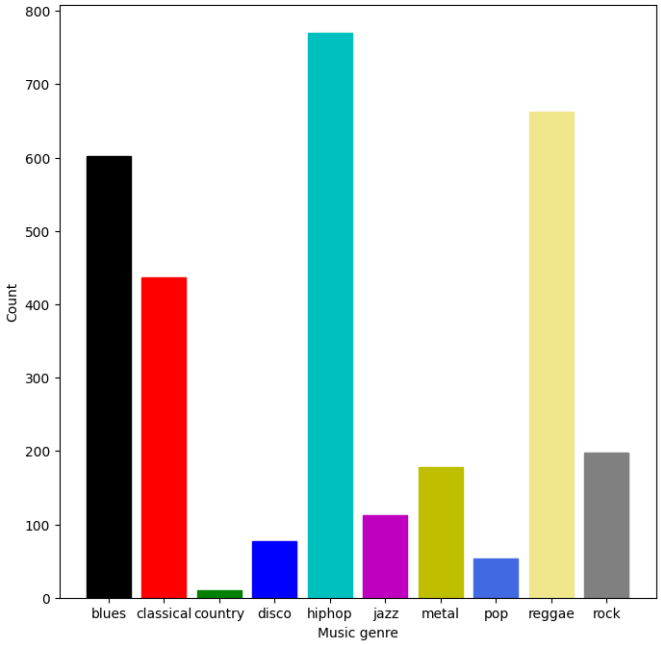


Figure 5.3: FMA Instrumentals music genre distribution bar chart

5.1.3 Timbre similarity

Based on experiment 4.1.2, we obtain the following graph shown in Figure 5.4.

Initially, we would expect the similarity to follow an inverse relationship to the DTW distance however after we performed a linear regression on the graph we observed a counter-intuitive result of a positive slope (positive relationship). We believe the reason why this is may be due to points in the first quadrant may have been outlier. These outlier might have been caused by the randomness introduced when computing the similarity score when building the IMEMNet dataset. We tried filtering out 4 points lying in the first quadrant which we believe are outliers and did observe a slight negative slope when performing linear regression (Figure 5.5).

Secondly, it is still an ongoing research as to whether timbre does have a direct impact to how we perceive emotions. It is indeed true that timbre are perceptive sound quality which explain why we are able to characterise trumpet sounds from pianos and so invoke different emotions based on subjective experiences. These are indirect relationship therefore trying to find direct correlation between the two metric may require a more complex experimental setup. In addition, extracting MFCC audio feature cannot fully express the timbre of an audio as said in **Background** and so we can only compare so much similarity between the music.

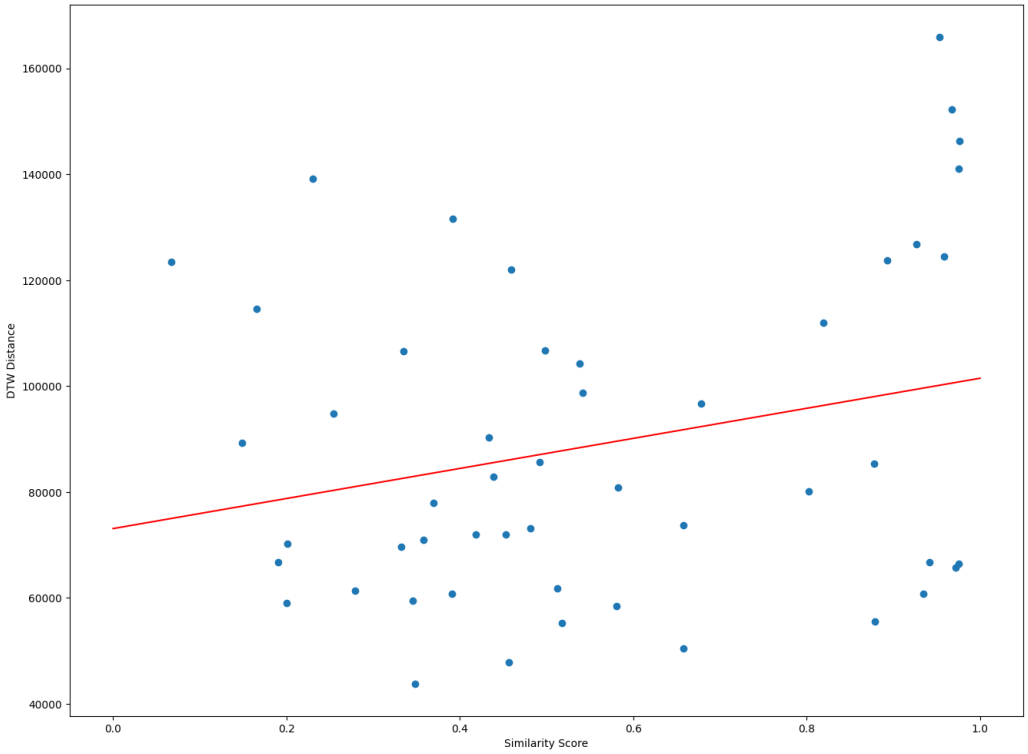


Figure 5.4: Similarity score - DTW distance graph with a line of best fit

5.1.4 KL Divergence (Audio Genre as emotion)

Models	KL Divergence
Pipeline 1.0 (Open-MusicLM)	**0.6668**
Pipeline 2.0 (MusicGen)	0.7889
Pipeline 2.0 (MusicGen) w/ SWR	0.8066

Table 5.2: KL Divergence evaluated on different models

Table 5.2 shows our result from experiment 4.1.2

Despite having the largest FAD score, Pipeline 1.0 (Open-MusicLM) has the lowest KL Divergence value. To investigate this further, we plotted the graph of Similarity score and KL Divergence across Audible-Visual-50 pairs shown in Figure 5.2 and 5.7.

We observe irregular behaviour suggesting a music genre classifier may not be the best tool for our proxy method. To clearly see this, we also randomly sample 50 IMEMNet dataset with varying similarity to see the result. What we expect is that the smaller the KL Divergence, the higher the similarity score should be (inverse relation).

From Figure 5.8 and 5.9, we do not observe the inverse relationship between similarity score and KL Divergence therefore we can conclude music genre may not be the best proxy metric to

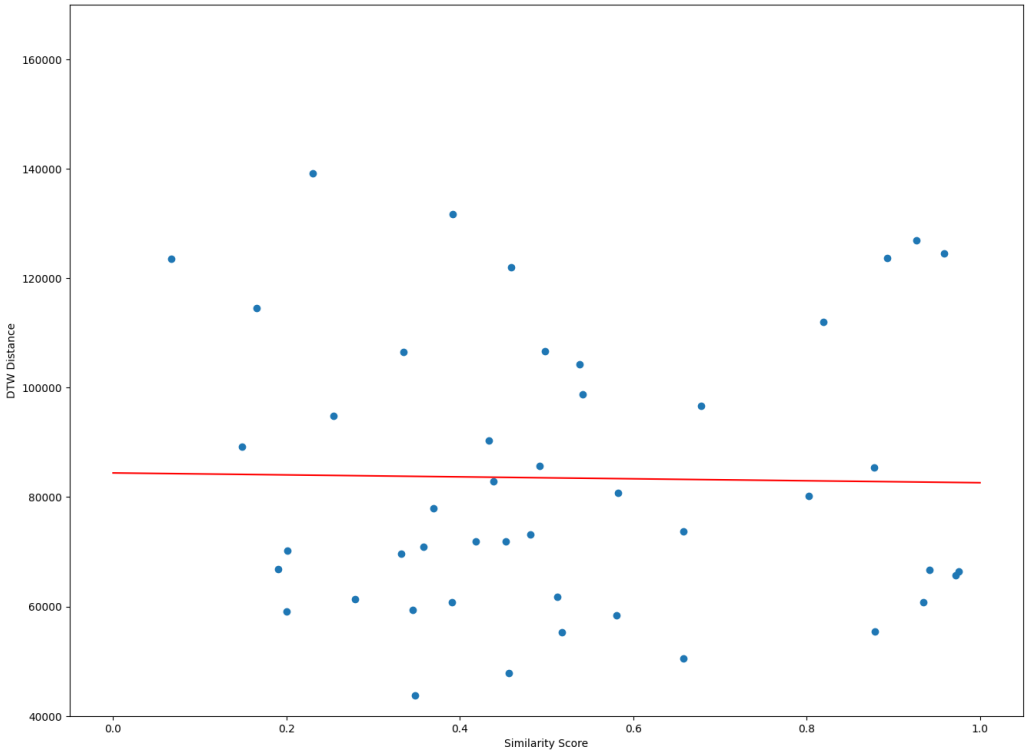


Figure 5.5: Similarity score - DTW distance graph with 4 points from the first quadrant removed and a line of best fit

measure different emotions in music.

Caption semantics

We also observe from Table 5.2 that using stopwords removal worsen the performance on our KL Divergence contrary to our belief. To ensure that this was the case, we measured how accurate our model was at predicting the same top music genre between our generated music and reference music. If both are the same genre, we increment the number of correct prediction. Table 5.3 shows the result for Pipeline 2.0 without stopwords removal and Pipeline 2.0 with stopwords removal. Based on the result, we see an improvement in accuracy contradicting our result in 5.2 but also emphasizing that music genre is not an ideal proxy metric.

Table 5.3: Accuracy of model at predicting the same music genre between generated music and reference music

Models	Correct	Incorrect	Total	Accuracy
Pipeline 2.0 (MusicGen)	6	44	50	12%
Pipeline 2.0 (MusicGen) w/ SWR	8	42	50	**16%**

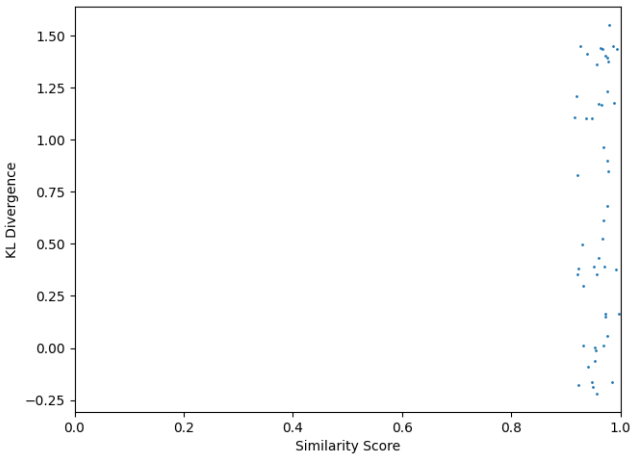


Figure 5.6: KL Divergence - Similarity score graph of Pipeline 1.0 using Audible-Visual-50 pairs

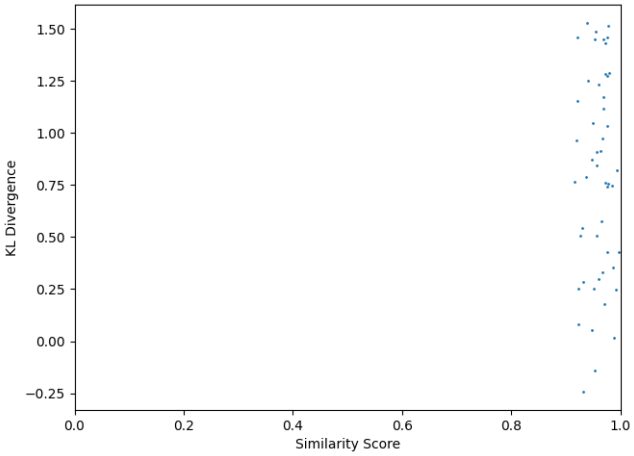


Figure 5.7: KL Divergence - Similarity score graph of Pipeline 2.0 using Audible-Visual-50 pairs

5.1.5 Qualitative Feedback

From Figure 5.10, we observe that certain caption generated using Audible-Visual-50 dataset only describe what the image is showing carrying with them little to no emotional elements. We also see that these sentence are quite generic and some are inaccurate which we believe may have skewed our evaluation above reiterating that the captioning speaker may not do well on non-painting-like images.

Based on experiment 4.1.3, we obtained 30 responses. We average their ratings and present the result in Table 5.4. We see that more than half of the generated music clip surveyed suited the corresponding image input more than its generated caption suggesting that the model has somewhat successfully generated music that emotionally align with the image input.

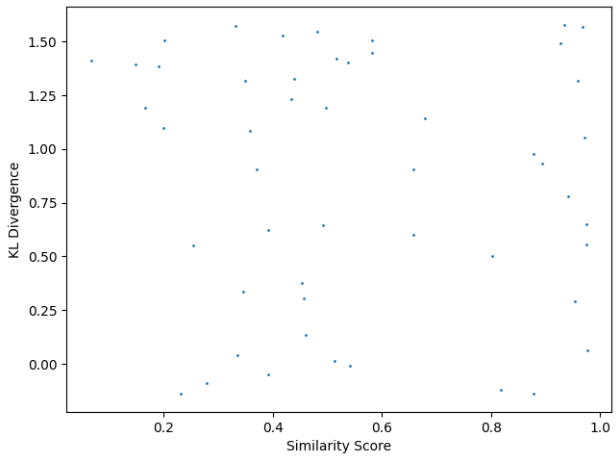


Figure 5.8: KL Divergence - Similarity score graph of Pipeline 1.0 using 50 randomly sampled IMEMNet pairs with varying similarity score

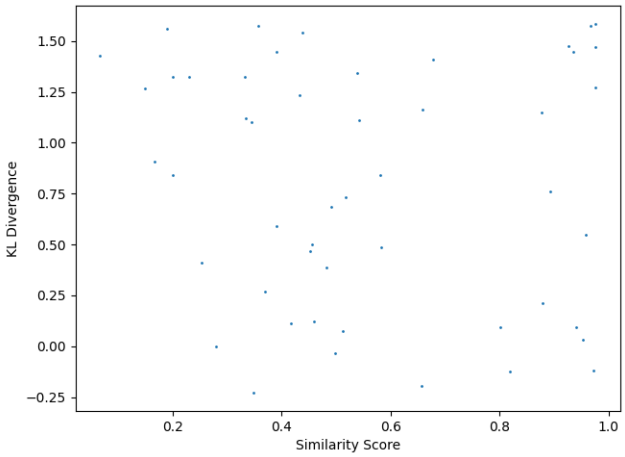


Figure 5.9: KL Divergence - Similarity score graph of Pipeline 2.0 using 50 randomly sampled IMEMNet pairs with varying similarity score

Furthermore, we listened to a few generated music clips and found that with image such as one in Figure 5.11. Despite the difference in color and both being an image containing the sea, we subjectively observe similar style in the generated music. However when comparing to Figure 5.12 which also an image containing the sea, the music were different.

To try and understand this behaviour, we pass our images to ArtEmis' encoder obtaining a 1000-d vector. We then reduce the dimension down to 2 components using PCA and then plot these points in Figure 5.13. Table 5.5 also show their generated caption (with stopwords removed). We observe that the same caption were generated for *igor* and *rock* but their generated music was different.

Additionally, we also observe in Figure 5.13 that image *kk* and *lr* are quite far from each other

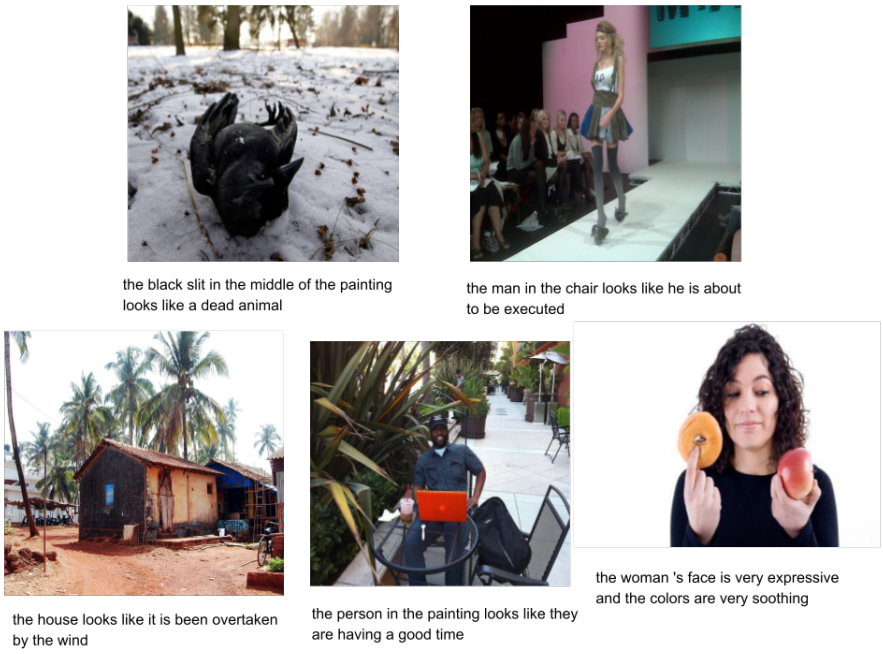


Figure 5.10: Image examples from Audio-Visual-50 with generated caption

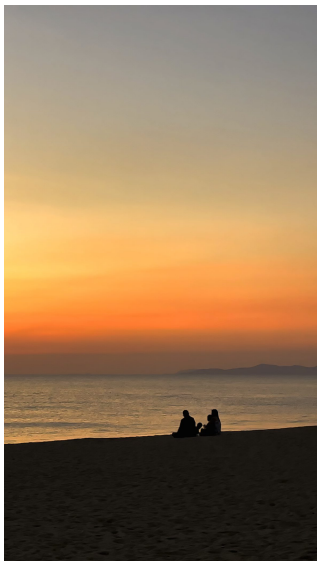
Table 5.4: Captions and images ratings suitability to generated music

-	**Ratings**	
Image	Caption	Image
Starry Night	7.2	**7.27**
Spiderman	5.17	**6.77**
Woman in Blue	**7.57**	7.47
Distracted Boyfriend	**4.43**	4.3
The Kiss	6.87	**7.1**

but have the word *calming* in common. We also observe that image *kk* and *p* are nearer to each other and both generated caption had the word *colors, warm, inviting* in common. This seems to be suggesting that the nearer two points are to each other, the more words they have in common. However, image *igor* and *rock* have the same generated caption however were not near each other suggesting that a 2D space may not be able to fully inform us well about both the semantics and the emotion of the image.

We investigated further by looking at where the music was generated. We pass the generated caption through MusicGen's text encoder, obtain a [10,1536]-d embedding from the encoder, squeeze them into 15360-d embedding and compute the cosine similarity table between all possible pairs. The result is shown in Figure 5.14. Furthermore, we also reduced 15360-d embedding to 2 dimension using PCA and visualise them in Figure 5.15.

Based on the PCA visualisation alone, we see that *kk* and *lr* are closer to each other and both are



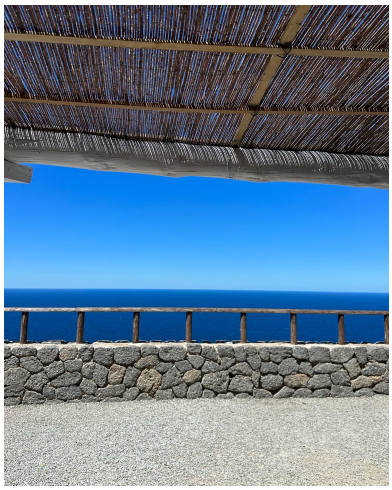


Figure 5.11: Images containing view of the sea (left: image *kk*, right: image *lr*)

Figure 5.12: Image containing view of the sea (image *m*)

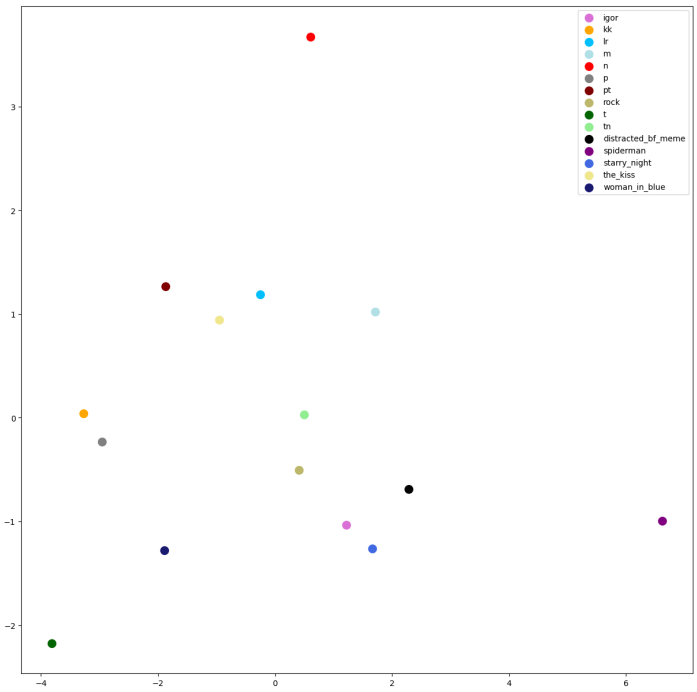


Figure 5.13: Image embedding after Principal Component Analysis

further away from *m*, explaining the similarity and dissimilarity of their generated music. However, when viewed against the cosine similarity table, the cosine similarity between *kk* and *lr* is only 0.44, while *m* and *woman in blue*, being further apart in the graph, have a higher cosine similarity of 0.5. Music generated from *m* and *woman in blue* also convey similar emotion (subjectively).

We see from Figure 5.5 that *rock* and *igor* had the same caption. This was also observed in Figure 5.14 that the cosine similarity between their text embedding were the same (cosine similarity = 1). When viewed in Figure 5.15, *rock* and *igor* are mapped onto the same point. Their generated music however were emotionally different. This is due to the sampling process the model performs for every quantize token it predicts. We believe that cosine similarity between different text prompt embeddings is a good metric for measuring emotional similarity in generated music however this wouldn't be the case for the same input caption described earlier.

We also tried measuring the timbre similarity of these audio. We compute the MFCC representation and compute the DTW distance between all pairs. We then clip any value above/below the mean $\pm$ 3 * standard deviation and perform a min-max normalization to scale the data between 0 and 1. A lower DTW distance suggests more similarity between music clips. The result is shown in Figure 5.16.

Although we observe a relatively low DTW distance between *kk* and *lr*, suggesting similarity

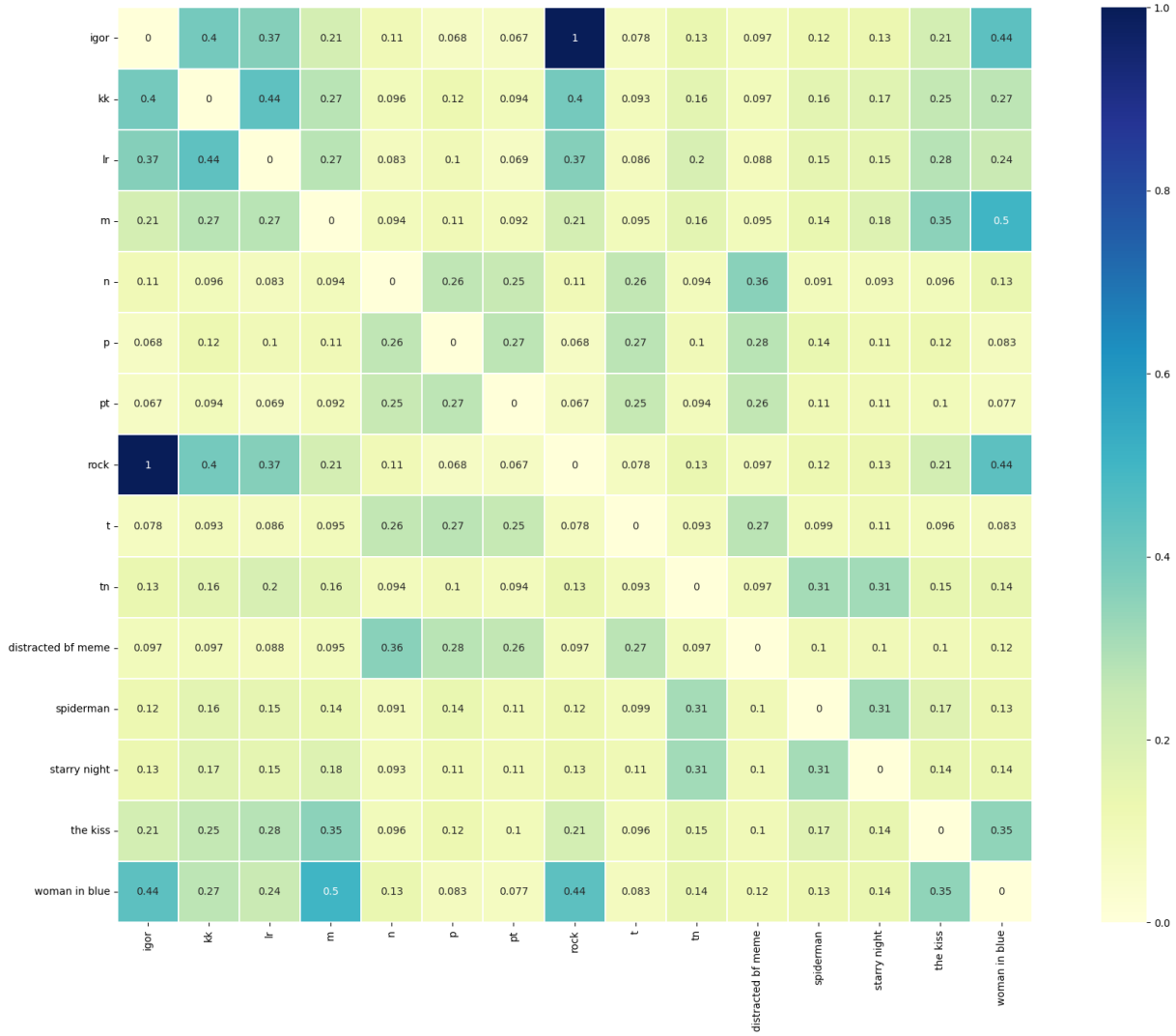


Figure 5.14: Cosine similarity between all pairs of text embeddings

Table 5.5: Image name and Generated caption with stopword removal

Image	Caption w/ SWR
igor	man 's face looking
kk	sunset calming colors warm inviting
lr	blue white colors calming soothing
m	grey sky dark blue sky make feel sad
n	woman good time
p	colors warm inviting
pt	figures detailed realistic
rock	man 's face looking
t	trees glowing wind
tn	painting abstract colors pleasing
distracted bf meme	girls good time
spiderman	shapes colors pleasing eye
starry night	stars sky calming
the kiss	bright colors woman 's dress calming
woman in blue	woman 's face sad colors dark

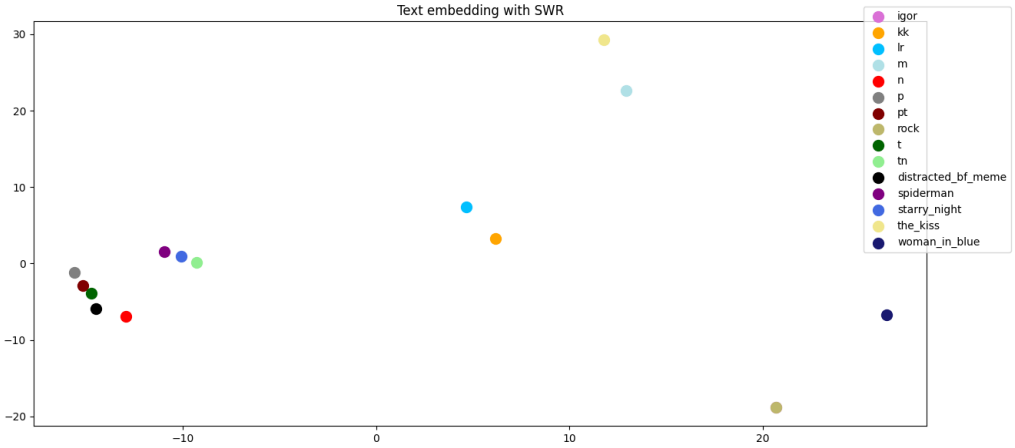


Figure 5.15: Text embedding w/ SWR after PCA

in their generated music, we do not observe a similar pattern between them and *m* like what we saw in the cosine similarity of text embeddings. We measure the correlation between the two tables (Figure 5.14 and Figure 5.16) and obtain a correlation score of 0.028, which suggests little to no linear correlation between the cosine similarity of caption embeddings and the optimal DTW path between music MFCCs. We also understand that due to the non-linear nature of neural networks, the relationship between the generated music and caption is non-linear. Therefore, measuring a linear correlation between them may not be effective, reiterating the need for a complex experimental setup to observe any correlation between the two metrics.

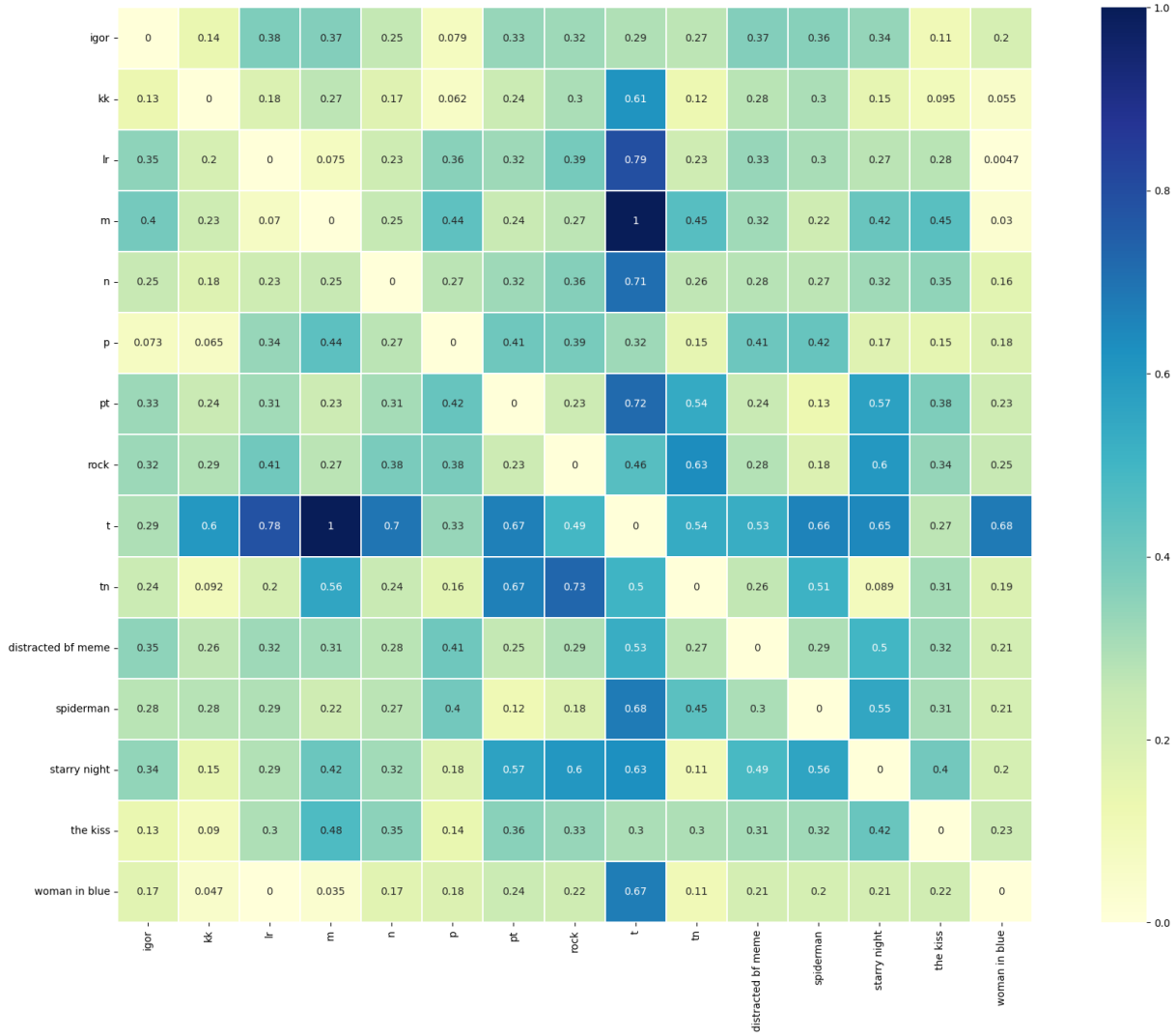


Figure 5.16: DTW distance between all pairs of MFCC audio features

6

ImAudible & Conclusion

6.1 ImAudible Gallery

To illustrate the use case and the motivation of the model. We created a 3D interactive gallery - ImAudible (Image Audible) gallery. The gallery is built using a 3D library called Three.JS[64] which uses WebGL - a Javascript API to render 2D/3D graphics. Also by using Three.JS, we can also host this on a website. Within the gallery, the user can navigate around the gallery and interact with paintings by mouse click. The small squared at the center of the screen represents the location the click is triggered within the environment. Clicking on a painting, plays the pre-generated music using the image clicked as input. A snapshot of the gallery is shown in Figure 6.1. The gallery is open to the public at https://imaudible.netlify.app. The images in the gallery were the 15 images we used in experiment 4.1.3.

6.2 Conclusion

We have successfully implement a working image-based music generator. Although we did not manage to get our End-to-End model working, our other model did seemed to generate plausible music of various musicality. Due to the subjectively of emotion, it poses a challenge to evaluate the model objectively. We have come up with different ways to do so with some degree of success however it is far from perfect. Our main setbacks are generating better captions with emotional elements and methods to quantify timbre and emotions - which are both ongoing research areas.

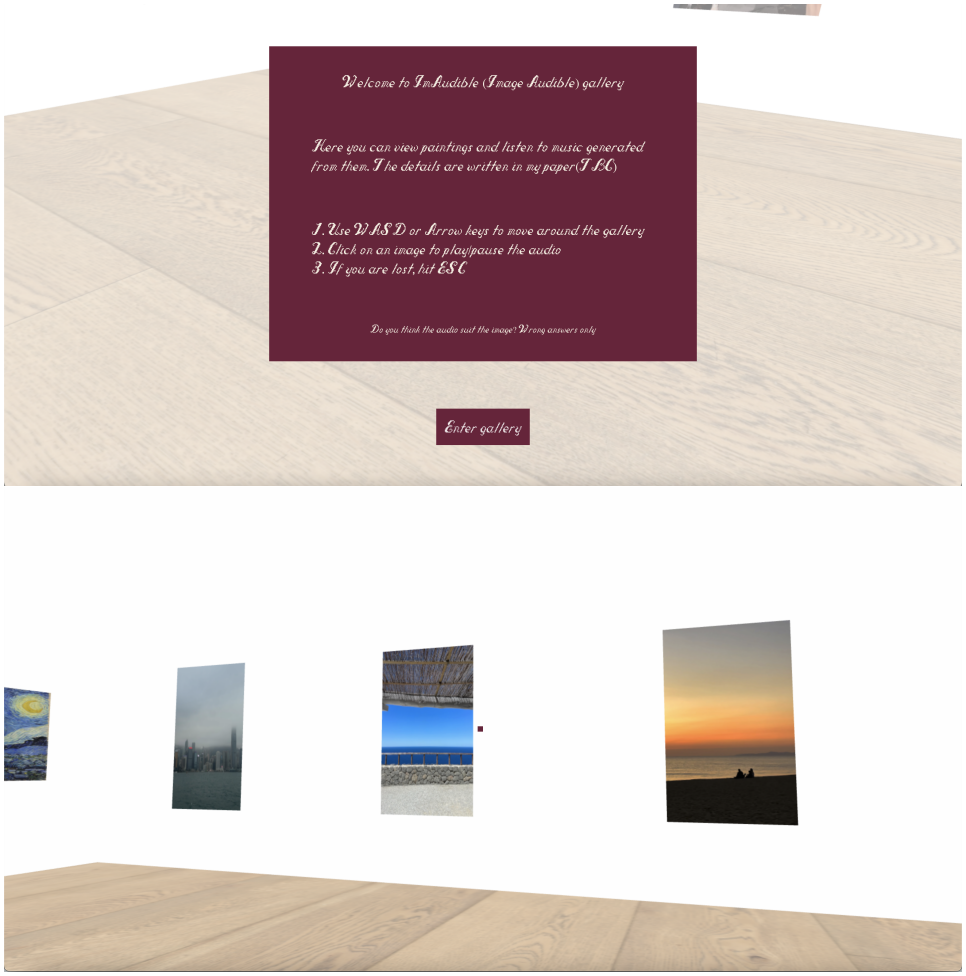


Figure 6.1: Snapshots from ImAudible Gallery

In order to get our end-to-end model working, we will need better hardware in order to utilise all of our IMEMNet training data. Once we have the joint image-music space, we would also need to fine tune the pretrained Open-MusicLM model to get better music generation.

6.3 Ethical consideration

The appropriation of using AI to generate music is debatable since generative music are not necessarily new but combinations of music seen during training. Needless to say the same thing can be argued about musicians, they do not necessarily make ”new” music but rather access a collection of past experiences of songs they’ve memorised, played or heard and turning it into their ”own”. At the present time, AI has no cognitive ability to understand originality and so biases and discrimination are more likely involved in generative music therefore a commercialised use may need further consideration.

Musical production or composition can be seen as a creative practice, having created a model which automatically does this may be seen as uncreative, unoriginal and against by some. Musical artists are less credited and rewarded for their creative act, devaluing the possibility of a musical career. A shadow looms over the music industry.

We believe strongly that music are bottled emotions in audible form and when done properly, represent an inner workings or stories felt by real people. Turning this into something made artificially removes a frequency which people can share and relate to.

6.4 Future Work

ArtEmis V2.0[65] is a new dataset which improves on the current ArtEmis that we have been discussing. Alongside it is also an improved image captioning speaker. Due to the time constraint, we did not have the chance to try out this model. For future work, we can do so by switching to this new speaker as the first component for our pipeline approach and see whether this would improve the performance.

Currently, ImAudible gallery uses pre-generated music, which is loaded and played whenever we click on a painting. For future work, we can extend this by allowing users to upload their own images, which would generate the corresponding music in real-time. This would provide a much more enhanced user experience, and we can also switch the displays in the gallery to the ones uploaded by users, creating an ever-shifting display of work.

We can also utilise neural networks to train and extract audio features which resembles more closer to music timbre than MFCC. This would mean creating/finding a suitable dataset and looking at more research papers on the topic.

A

Appendix A

Figure A.1: CDCML loss curve

Bibliography

[1] *15 amazing real-world applications of ai everyone should know about.* [Online]. Available: https://www.forbes.com/sites/bernardmarr/2023/05/10/15-amazing-real-world-applications-of-ai-everyone-should-know-about/, (accessed: 12.06.2024).

[2] B. Zhang, H. Shi, and H. Wang, *Machine learning and ai in cancer prognosis, prediction, and treatment selection: A critical approach.* [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10312208/#:~:text=Machine%20learning%20(ML)%2C%20a,in%20predicting%20cancer%20than%20clinicians, (accessed: 10.11.2023).

[3] OpenAI, *Chatgpt.* [Online]. Available: https://chat.openai.com/auth/login, (accessed: 10.11.2023).

[4] OpenAI, *Dall · e 2.* [Online]. Available: https://openai.com/dall-e-2, (accessed: 10.11.2023).

[5] M. Cezza, *Learning more about yourself could help you better understand others.* [Online]. Available: https://www.bps.org.uk/research-digest/learning-more-about-yourself-could-help-you-better-understand-others, (accessed: 19.11.2023).

[6] IBM. [Online]. Available: https://www.ibm.com/topics/neural-networks, (accessed: 27.05.2024).

[7] NVIDIA. [Online]. Available: https://docs.nvidia.com/deeplearning/performance/dl-performance-fully-connected/index.html, (accessed: 27.05.2024).

[8] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, 2015. arXiv: 1512.03385 [cs.CV].

[9] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," 1986. [Online]. Available: https://api.semanticscholar.org/CorpusID:62245742.

[10] H. Swati. [Online]. Available: https://medium.com/@noorulhudaajmal12/understanding-principal-component-analysis-pca-f831f0ce08c5.

[11] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, *Attention is all you need*, 2023. arXiv: 1706.03762 [cs.CL].

[12] [Online]. Available: https://www.learndatasci.com/glossary/cosine-similarity/, (accessed: 05.06.2024).

[13] P. Achlioptas, M. Ovsjanikov, K. Haydarov, M. Elhoseiny, and L. Guibas, "Artemis: Affective language for visual art," *CoRR*, vol. abs/2101.07396, 2021.

[14] [Online]. Available: https://www.wikiart.org/, (accessed: 25.05.2024).

[15] J. Machajdik and A. Hanbury, "Affective image classification using features inspired by psychology and art theory," in *Proceedings of the 18th ACM International Conference on Multimedia*, ser. MM '10, Firenze, Italy: Association for Computing Machinery, 2010, pp. 83–92, ISBN: 9781605589336. DOI: 10.1145/1873951.1873965. [Online]. Available: https://doi.org/10.1145/1873951.1873965.

[16] V. Yanulevskaya, J. C. van Gemert, K. Roth, A.-K. Herbold, N. Sebe, and J.-M. Geusebroek, "Emotional valence categorization using holistic image features," *2008 15th IEEE International Conference on Image Processing*, pp. 101–104, 2008. [Online]. Available: https://api.semanticscholar.org/CorpusID:14629609.

[17] S. Zhao, Y. Gao, X. Jiang, H. Yao, T.-S. Chua, and X. Sun, "Exploring principles-of-art features for image emotion recognition," in *Proceedings of the 22nd ACM International Conference on Multimedia*, ser. MM '14, Orlando, Florida, USA: Association for Computing Machinery, 2014, pp. 47–56, ISBN: 9781450330633. DOI: 10.1145/2647868.2654930. [Online]. Available: https://doi.org/10.1145/2647868.2654930.

[18] Q. You, J. Luo, H. Jin, and J. Yang, *Building a large scale dataset for image emotion recognition: The fine print and the benchmark*, 2016. arXiv: 1605.02677 [cs.AI].

[19] K. Xu, J. Ba, R. Kiros, *et al.*, "Show, attend and tell: Neural image caption generation with visual attention," in *Proceedings of the 32nd International Conference on Machine Learning*, F. Bach and D. Blei, Eds., ser. Proceedings of Machine Learning Research, vol. 37, Lille, France: PMLR, Jul. 2015, pp. 2048–2057. [Online]. Available: https://proceedings.mlr.press/v37/xuc15.html.

[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, Dec. 1997. DOI: 10.1162/neco.1997.9.8.1735.

[21] Suno. [Online]. Available: https://suno.com/, (accessed: 25.05.2024).

[22] Udio. [Online]. Available: https://www.udio.com/, (accessed: 25.05.2024).

[23] A. Défossez, J. Copet, G. Synnaeve, and Y. Adi, *High fidelity neural audio compression*, 2022. arXiv: 2210.13438 [eess.AS].

[24] [Online]. Available: https://www.assemblyai.com/blog/what-is-residual-vector-quantization/, (accessed: 27.05.2024).

[25] Barracuda. [Online]. Available: https://www.barracuda.com/support/glossary/data-compression, (accessed: 27.05.2024).

[26] J. Copet, F. Kreuk, I. Gat, *et al.*, *Simple and controllable music generation*, 2024. arXiv: 2306.05284 [cs.SD].

[27] A. Agostinelli, T. I. Denk, Z. Borsos, *et al.*, *Musiclm*. DOI: https://doi.org/10.48550/arXiv.2301.11325. [Online]. Available: https://arxiv.org/pdf/2301.11325.pdf, (accessed: 23.12.2023).

[28] A. Zhang and J. Berry. [Online]. Available: https://github.com/zhvng/open-musiclm, (accessed: 26.05.2024).

[29] Z. Borsos, R. Marinier, D. Vincent, *et al.*, *Audiolm: A language modeling approach to audio generation*, 2023. arXiv: 2209.03143 [cs.SD].

[30] Q. Huang, A. Jansen, J. Lee, R. Ganti, J. Y. Li, and D. P. W. Ellis, *Mulan: A joint embedding of music audio and natural language*, 2022. arXiv: 2208.12415 [eess.AS].

[31] Y.-A. Chung, Y. Zhang, W. Han, *et al.*, *W2v-bert: Combining contrastive learning and masked language modeling for self-supervised speech pre-training*, 2021. arXiv: 2108.06209 [cs.LG].

[32] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi, *Soundstream: An end-to-end neural audio codec*, 2021. arXiv: 2107.03312 [cs.SD].

[33] Y. Wu, K. Chen, T. Zhang, *et al.*, *Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation*, 2024. arXiv: 2211.06687 [cs.SD].

[34] Y. Li, R. Yuan, G. Zhang, *et al.*, *Mert: Acoustic music understanding model with large-scale self-supervised training*, 2024. arXiv: 2306.00107 [cs.SD].

[35] C. Raffel, N. Shazeer, A. Roberts, *et al.*, *Exploring the limits of transfer learning with a unified text-to-text transformer*, 2023. arXiv: 1910.10683 [cs.LG].

[36] P. Presti, D. Ruzzon, P. Avanzini, F. Caruana, G. Rizzolatti, and G. Vecchiato, *Measuring arousal and valence generated by the dynamic experience of architectural forms in virtual environments*, 2022. DOI: 10.1038/s41598-022-17689-9.

[37] T. M. Sutton, A. M. Herbert, and D. Q. Clark, "Valence, arousal, and dominance ratings for facial stimuli," *Quarterly Journal of Experimental Psychology*, vol. 72, no. 8, pp. 2046–2055, 2019, PMID: 30760113. DOI: 10.1177/1747021819829012. eprint: https://doi.org/10.1177/1747021819829012. [Online]. Available: https://doi.org/10.1177/1747021819829012.

[38] P. B. Posner J Russell JA, "The circumplex model of affect: An integrative approach to affective neuroscience, cognitive development, and psychopathology," vol. 17(3):715-34, 2005. DOI: 10.1017/S0954579405050340.

[39] M. Clinic. [Online]. Available: https://www.mayoclinic.org/tests-procedures/eeg/about/pac-20393875, (accessed: 28.05.2024).

[40] B. S. Branco D Gonçalves ÓF, "A systematic review of international affective picture system (iaps) around the world," vol. 23(8):3866, 2023. DOI: 10.3390/s23083866.

[41] M. A, Z. Ł, J. K, and G. A, "The nencki affective picture system (naps): Introduction to a novel, standardized, wide-range, high-quality, realistic picture database. behav res methods," vol. 46(2):596-610, 2014. DOI: 10.3758/s13428-013-0379-1.

[42] R. Kosti, J. Alvarez, A. Recasens, and A. Lapedriza, "Context based emotion recognition using emotic dataset," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[43] A. Alajanki, Y.-H. Yang, and M. Soleymani, "Benchmarking music emotion recognition systems," *PLOS ONE*, 2016, under review.

[44] S. Zhao, Y. Li, X. Yao, *et al.*, *Emotion-based end-to-end matching between image and music in valence-arousal space*, 2020. arXiv: 2009.05103 [cs.CV].

[45] S. Kim, M. Seo, I. Laptev, M. Cho, and S. Kwak, *Deep metric learning beyond binary supervision*, 2019. arXiv: 1904.09626 [cs.CV].

[46] K. Kilgour, M. Zuluaga, D. Roblek, and M. Sharifi, *Fréchet audio distance: A metric for evaluating music enhancement algorithms*, 2019. arXiv: 1812.08466 [eess.AS].

[47] S. Hershey, S. Chaudhuri, D. P. W. Ellis, *et al.*, "Cnn architectures for large-scale audio classification," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017. [Online]. Available: https://arxiv.org/abs/1609.09430.

[48] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, *et al.*, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE ICASSP 2017*, New Orleans, LA, 2017.

[49] D. Dowson and B. Landau, "The fréchet distance between multivariate normal distributions," *Journal of Multivariate Analysis*, vol. 12, no. 3, pp. 450–455, 1982, ISSN: 0047-259X. DOI: https://doi.org/10.1016/0047-259X(82)90077-X. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0047259X8290077X.

[50] 3b1b, *But what is the fourier transform? a visual introduction.* [Online]. Available: https://www.youtube.com/watch?v=spUNpyF58BY, (accessed: 06.06.2024).

[51] V. V. .-. T. S. of AI, *Https://www.youtube.com/watch?v=4sh2nfbqz8*. [Online]. Available: https://www.youtube.com/watch?v=4_SH2nfbQZ8, (accessed: 06.06.2024).

[52] J. D. Warren, A. R. Jennings, and T. D. Griffiths, "Analysis of the spectral envelope of sounds by the human brain," en, *Neuroimage*, vol. 24, no. 4, pp. 1052–1057, Feb. 2005.

[53] M. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney, "Content-based music information retrieval: Current directions and future challenges," *Proceedings of the IEEE*, vol. 96, pp. 668–696, May 2008. DOI: 10.1109/JPROC.2008.916370.

[54] X. Liu, Y. Xu, K. Alter, and J. Tuomainen, "Emotional connotations of musical instrument timbre in comparison with emotional speech prosody: Evidence from acoustics and event-related potentials," *Frontiers in Psychology*, vol. 9, 2018, ISSN: 1664-1078. DOI: 10.3389/fpsyg.2018.00737. [Online]. Available: https://www.frontiersin.org/journals/psychology/articles/10.3389/fpsyg.2018.00737.

[55] [Online]. Available: https://www.theaidream.com/post/dynamic-time-warping-dtw-algorithm-in-time-series, (accessed: 06.06.2024).

[56] Meta. [Online]. Available: https://huggingface.co/spaces/facebook/MusicGen, (accessed: 27.05.2024).

[57] GoogleAI. [Online]. Available: https://www.kaggle.com/datasets/googleai/musiccaps, (accessed: 29.05.2024).

[58] F. M. Archive. [Online]. Available: https://freemusicarchive.org/, (accessed 29.05.2024).

[59] [Online]. Available: https://github.com/gudgud96/frechet-audio-distance/tree/main, (accessed: 25.05.2024).

[60] S. Salvador and P. K. Chan, "Fastdtw: Toward accurate dynamic time warping in linear time and space," 2004. [Online]. Available: https://api.semanticscholar.org/CorpusID:6226669.

[61] D. Yang, J. Yu, H. Wang, *et al.*, *Diffsound: Discrete diffusion model for text-to-sound generation*, 2023. arXiv: 2207.09983 [cs.SD].

[62] F. Kreuk, G. Synnaeve, A. Polyak, *et al.*, *Audiogen: Textually guided audio generation*, 2023. arXiv: 2209.15352 [cs.SD].

[63] [Online]. Available: https://huggingface.co/SeyedAli/Musical-genres-Classification-Hubert-V1, (accessed: 30.05.2024).

[64] [Online]. Available: https://threejs.org/, (accessed: 09.06.2024).

[65] [Online]. Available: https://www.artemisdataset-v2.org/, (accessed: 17.06.2024).