Imperial College London

IMPERIAL COLLEGE LONDON

DEPARTMENT OF COMPUTING

Decoding Music from the Brain using Generative Models

Author:
Jason Lee

Supervisor:
Yingzhen Li
Co-Supervisor:
Pedro Martinez Mediano

Submitted in partial fulfillment of the requirements for the MSc degree in MSc Computing of Imperial College London

September 2023

Abstract

This project aims to investigate the viability of using generative models to reconstruct musical stimuli from EEG recordings, which is a task that has not been conclusively been proven to be possible. The underlying premise of the project is that the task of translating brain signals into audio is reinterpreted as a fully convolutional image-to-image translation task, where the EEG and music are represented in visual formats in the form of segmented spectrograms and mel-spectrograms, which enables the use of various generative models such as GANs. The aim of this project is not to perfectly decode the musical stimulus from the EEG, nor is it an exhaustive and conclusive judgement on the feasibility of such an approach. It is rather a preliminary and experimental exploration of one possible methodology, which could provide some insight as to the potential of the approach as well as future variations that could be implemented.

The project was conducted by firstly developing a baseline model using a CNN, where potential convolutional architectures such as UNet and ResNet were explored. The results of the baseline were then compared to that of more enhanced generative models, such as Conditional GANs (cGAN). The expectation is that the cGAN should show an improvement on the results compared to the baseline.

The results show that the final and best performing version of the cGAN model outperforms the baseline, as expected. This is shown in terms of both quantitative metrics such as the Mel Cepstral Distortion (MCD) as well as subjective evaluation using the Mean Opinion Score (MOS). In addition, the results from the cGAN show that, while the generated outputs generally do not retain most musical aspects of the stimulus very well, some elements such as the tempo and rhythm can be retained to a certain degree, which is consistent with neuroscientific findings. These findings suggest that this is an approach that, with further improvements, could deliver promising results in decoding neural stimuli from EEG signals.

Acknowledgments

My most sincere gratitude goes to my two supervisors, Dr Yingzhen Li and Dr Pedro Martinez Mediano at the Department of Computing at Imperial College London. This thesis would not have been possible without their guidance.

I would also like to extend my gratitude to my friends and family for supporting me throughout this journey, especially during the most difficult of times.

Contents

Chapter 1

Introduction

Deciphering the intricate language of neural activity has long been an objective in the realm of computational neuroscience. The emergence of technologies such as Electroencephalography (EEG) and Functional Magnetic Resonance Imaging (fMRI) scans provide ways to retrieve information directly from the brain, but the precise nature of the information retrievable from these methods, as well as the reliability and accuracy of such extraction, remains a subject of fervent debate and exploration within both computational and cognitive neuroscience communities.

The goal of this project is to contribute to this discussion by attempting to reconstruct musical stimuli from EEG recordings using generative models. Drawing from the latest advancements in computational neuroscience and machine learning, the aim is to bridge the gap between the realm of electrical brain signals and of musical experiences. The degree of success of this project, naturally, will primarily be dependent on whether the reconstructed audio has in any way a resemblance with the original stimulus. However, it is also important to recognise that this is an ambitious project in an immature field, set within a very limited time frame. It must also be acknowledged that the task of decoding the brain is one that has remained largely unsolved, especially when it comes to reconstruction of stimuli. Therefore, it should be noted that the objective of this project is not of a definitive or conclusive nature, but rather an exploratory one. With this in mind, there are two main questions that this project aims to answer:

1. To what degree, if at all, is it possible to reconstruct musical stimuli from EEG signals?
2. To what extent is the methodology of this project a viable approach to solving this problem?

The central premise of this project's methodology revolves around the reinterpretation of the problem at hand - namely, converting electrical EEG signals to sound waves. Rather than wrestling with raw electrical signals and audio synthesis, the problem is recasted as an image-to-image translation task. In other words, both the EEG recordings and stimulus music are to be represented in a visual format in the form of spectrograms and mel-spectrograms, instead of dealing with them in their respective original domains. The advantage of this approach is that image generation models have been explored extensively in recent times, providing a plethora of tools to harness when compared to the task of converting raw electrical signals to

audio. These tools include, but is not limited to, models such as convolutional networks and generative adversarial networks. However, this methodology relies heavily on the assumption that EEG features can be captured and represented via visual means, and that this can be accurately translated into the target domain of mel-spectrograms.

The significance of this project transcends its immediate boundaries, resonating with broader implications for an array of fields. Beyond the decoding of musical stimuli, the findings can shape the development of brain-computer interfaces (BCI), which aim to facilitate direct and seamless communication between the brain and computer systems. Furthermore, the ramifications extend into domains as diverse as audio synthesis and cognitive processing, unveiling new opportunities for understanding and harnessing neural information.

The reconstruction of neural stimuli may well turn out to be an impossible task, but that does not imply that it is a topic not worthy of exploration. This project was therefore conducted with the outlook of becoming a foundation on which future projects can be built, rather than providing a comprehensive conclusion on the task of neural stimulus reconstruction.

Chapter 2

Background

2.1 Key Concepts

2.1.1 Music and the Brain

Music has long been known to have the power to evoke a profound emotional response within the human psyche, and experiments confirm that music affects various aspects of cognitive and emotional processing in the brain [1]. Different types of music are also known to elicit specific patterns of brain activity, which is demonstrated in the form of brain waves. For instance, when exposed to fast-paced, rhythmic compositions, our brains tend to exhibit an increase in beta waves, which are associated with heightened alertness, focus, and a sense of engagement. In contrast, the slow, calming melodies lead to an uptick in alpha and theta waves, signifying a calm, meditative state of the brain.

Analysing EEG signals is a way to find correlations between the stimulus and the brain waves, which can provide valuable insights into the neural correlates of music perception and cognitive responses. For instance, the effects of different kinds of melody and rhythms on the brain has been a topic of interest for neuroscientists. This idea has been further applied in the past in areas such as emotion recognition [2] and music classification [3, 4] based on EEG signals. These studies have been largely quite successful and provide evidence that the effect of music is reflected in brain signals, which can be monitored via EEG.

In essence, the intricate relation between music and the human brain, illuminated by the analysis of EEG signals, not only enriches our understanding of music's impact on our emotions and cognition but also opens doors to innovative applications that span the realms of psychology, neuroscience, and technology.

Electroencephalography (EEG)

Electroencephalography (EEG) is a non-invasive method for measuring the electrical activity in the brain. It involves placing a set of electrodes, ranging anywhere from 20 to more than 100, on specific locations on the scalp to detect and record the electrical signals produced by the brain's neurons. These electrodes are strategically positioned to capture the neural activity from different regions of the brain, allowing for a comprehensive assessment of brainwave

patterns.

These signals are categorized into different frequency bands, such as alpha, beta, theta, and delta waves. Each of these bands is associated with specific brain states and functions. For example, alpha waves (8-12 Hz) indicate a relaxed and wakeful state, while beta (12-40 Hz) and gamma (40-100 Hz) waves reflect a more active and focused state of alertness and cognitive engagement. Theta (4-8 Hz) and delta (0-4 Hz) waves, on the other hand, are typically observed during transitions between wakefulness and sleep or deep, restorative sleep [5].

EEG is a versatile tool with a wide range of applications. In clinical settings, it is commonly used for medical purposes, such as diagnosing and monitoring neurological conditions like epilepsy [6], where it helps identify abnormal electrical activity in the brain. In cognitive neuroscience and neuropsychology, EEG plays a pivotal role in understanding brain activity during various cognitive processes. For instance, researchers use EEG to investigate sleep stages, tracking changes in brainwave patterns throughout the sleep cycle [7]. Additionally, EEG is employed to study how the brain processes sensory information during tasks like visual and auditory perception, shedding light on the neural mechanisms underlying the processes of perception and cognition. In essence, EEG provides a detailed and dynamic window into the brain's state and function, enabling researchers to explore the intricacies of human cognition and brain function.

In the context of this project, the EEG data used was collected while participants were listening to a selection of music pieces. A detailed discussion of the dataset will be provided in the next chapter.

The Spectrogram and Mel-Spectrogram

As mentioned in the introduction, the core premise of this project is that the electrical and sound waves of the EEG and audio, respectively, are represented via a visual medium. This is because there is a difficulty in processing raw electrical and audio signals to be used in neural networks. This difficulty will be circumvented in both domains using the *EEG spectrogram* and *mel-spectrogram*.

The concept of the spectrogram is quite straightforward. A spectrogram is a visual representation of a signal's frequency content over time, computed by breaking the signal into short, overlapping segments, applying a window function to each segment, and then performing a Fast Fourier Transform (FFT) to analyze the frequency components within each window. The result is a matrix where time is on the x-axis, frequency on the y-axis, and color or intensity represents the power or energy of each frequency component at a specific time, unveiling how the signal's frequencies change over time. Spectrograms are crucial in tasks involving audio analysis, speech processing, and scientific research.

For EEG signals, this is done by first computing the power spectral density (PSD) of the signal. The PSD describes how the power of a signal is distributed across different frequencies. In simpler terms, it is a measure of how much energy or power is contained within each frequency component of a signal. In the context of EEG, this will be the electrical signals from the brain that is grouped in the different frequency bands. The spectrogram of the EEG will therefore visualize how the power spectral density of a signal varies with time.

In the audio domain, a similar transition is applied, with a few extra steps to account for the perceptive differences in audio. The raw audio will be converted to what is known as the mel-spectrogram. The mel-spectrogram is a variation of the spectrogram that takes into account the fact that humans do not perceive frequencies on a linear scale – instead, the frequencies are converted into the mel scale, which is a logarithmic function of the actual frequency, to reflect the actual perception of audio by humans (see Figure 3). In short, the mel-spectrogram maps the Fourier-transformed frequency spectrum onto the mel scale, which approximates the non-linear relationship between frequency and perceived pitch [8]. The mel-spectrogram provides a visual representation of the intensity of different frequency components over time, making it a valuable tool for analysing and processing audio signals.

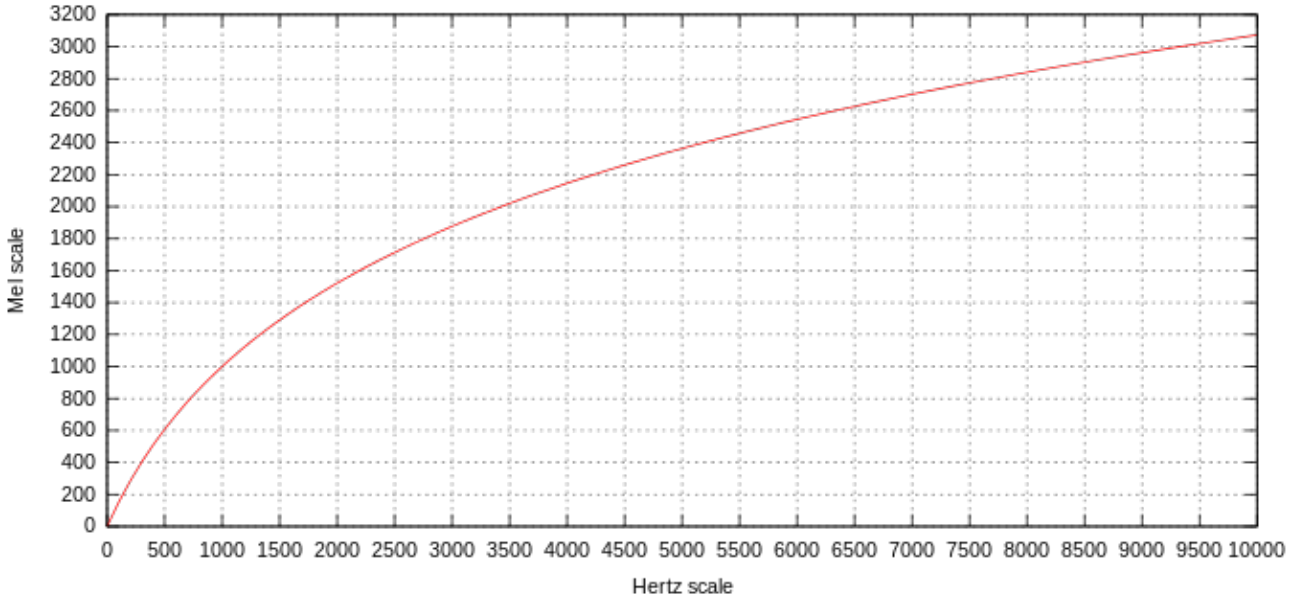


Figure 1: The Mel Scale vs Hertz Scale [9]

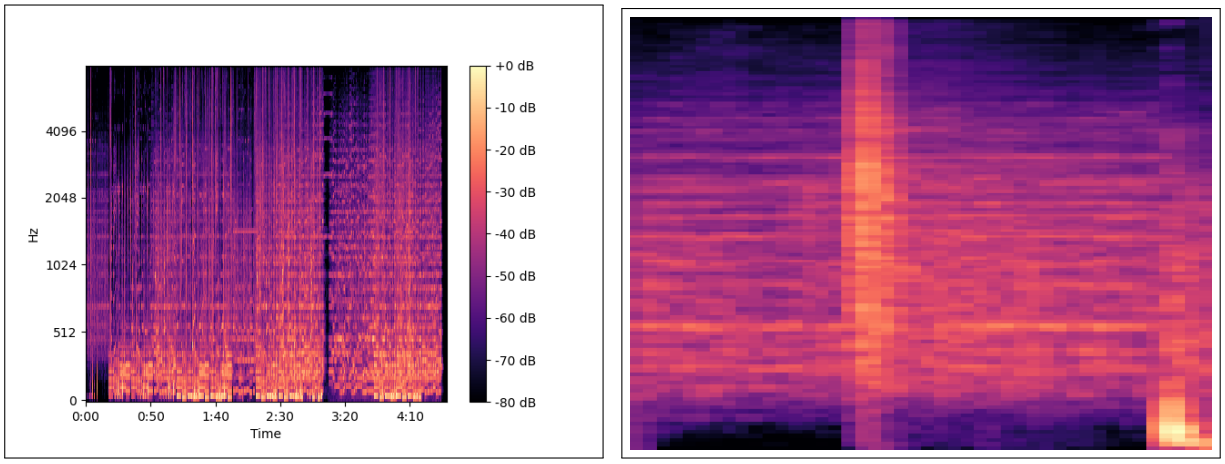


Figure 2: (a) A mel-spectrogram of the song 'First Fires' by Bonobo. (b) A mel-spectrogram of a one-second snippet of (a).

Given a mel-spectrogram, there must also be a tool to convert it back into raw audio form. One straightforward way of doing this is by using the Griffin-Lim algorithm, which approximates

the Short-Time Fourier Transform magnitude of the mel spectrum to convert the spectrogram into wave form. This method, however, has limitations as the quality of the output is severely degraded compared to the original audio. This can be remedied by using sound synthesisers known as Vocoders, which are pre-trained neural networks with the purpose of generating audio from spectrograms. An example of this is the MelGAN, which is a non-autoregressive, fully convolutional GAN used for sound synthesis [10]. Some more state-of-the-art vocoders integrate these network architectures, such as in [11], which result in networks that provide very high-quality conversions from spectrograms to audio.

The tools mentioned above are ways that would allow the media that would otherwise be very difficult to process be converted to more tangible representations. The use of spectrograms and mel-spectrograms will allow the problem of audio generation from EEG signals to be reduced into image generation from other images, which simplifies the design of the neural network.

2.1.2 Neural Networks

The generative model that will be used for the construction of mel-spectrograms from EEG data is one of the most essential aspects of this project. As mentioned, this project can be described as an image translation task (from EEG representation to mel-spectrogram), which narrowed down the options when it came to potential architectures that can be used. In the end, the GAN architecture was chosen over other networks such as VAEs due to their prevalence in state-of-the-art application and literature, and their relevance when it comes to image translation tasks. A CNN was chosen to be used as a baseline model to compared the performance of the generative model. An overview of the architectures used in this project is provided below. A more detailed discussion of the architecture, including the individual layers and hyperparameters, can be found in Chapter 3.

Convolutional Neural Networks

The first neural network model that will be explored is the convolutional neural network, which will serve as the baseline model. There exists various convolutional architectures and implementations, with the field advancing rapidly in recent years. Two of the architectures that will be tested in this project is the UNet and ResNet architectures.

UNet is a popular architecture primarily used for semantic segmentation tasks in computer vision, known mostly for its effectiveness in capturing fine-grained details while maintaining a high-level contextual understanding of the input image [12]. The UNet consists of an encoding path and a decoding path, where the encoding path is a series of convolutional and pooling layers that progressively reduce the spatial dimensions of the input image while increasing the number of channels, while the decoding path involves transposed convolutions or upsampling layers that recover the spatial dimensions of the features while gradually reducing the number of channels. The encoding and decoding paths are connected using a skip connection, which allow the network to merge feature maps from different levels, combining detailed information from the encoding path with contextual information from the decoding path. The UNet implementation used in this project uses a LeakyReLU activation in the downsampling blocks and ReLU in the upsampling blocks, with instance normalisation in both directions.

The other architecture to be used is the ResNet [13], which, similarly to the UNet, uses skip connections but with the addition of residual blocks. In a residual block, the input is split into two paths - one path directly propagates the input to the output, and the other path applies a series of convolutional layers to learn the residual or the difference between the output and the input. The outputs from both paths are then added element-wise, creating a ”residual” connection. This approach enables training of much deeper networks by avoiding the vanishing gradient problem and allowing the network to learn incremental changes to the input. The specific implementations of ResNet in this project will be the 6 and 9 block architectures. This ResNet implementation uses ReLU activation with instance normalisation.

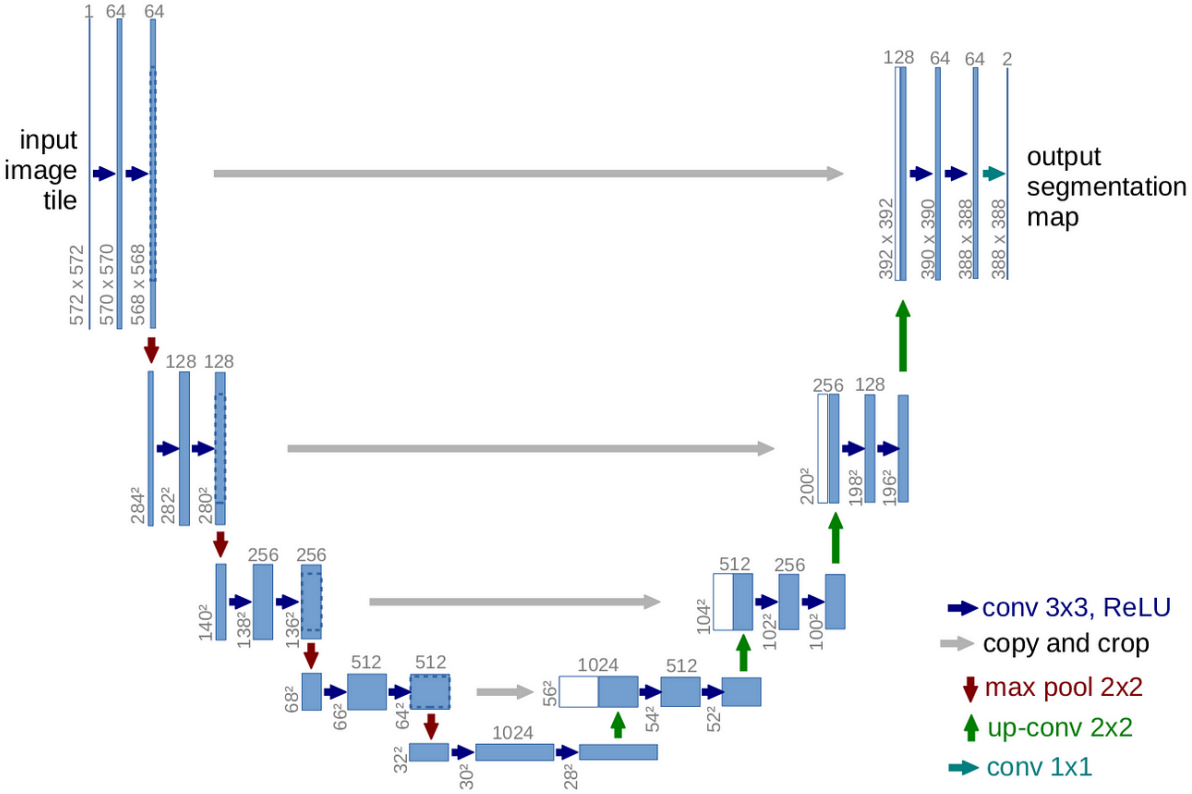


Figure 3: The UNet Architecture [12]

Generative Adversarial Networks

A generative adversarial network (GAN) is a generative model architecture that involves two types of networks: the generator and the discriminator [14]. When it comes to image generation, the goal of the generator is to generate images that closely resemble a sample set of ’true’ images, while the discriminator aims to distinguish between the ’fake’ and ’real’ images. The two networks engage in a training process resembling a zero-sum game - the generator aims to improve its generation skills to fool the discriminator, while the discriminator strives to become better at distinguishing real from fake data. This structure of networks is said to be analogous to the relation between counterfeiters and cops, where each entity tries to beat the other. This adversarial process is guided by the minimax loss function (Equation 1), where the generator minimizes the likelihood of being caught (maximizing the log-probability of the

discriminator making an incorrect prediction), while the discriminator aims to maximize its accuracy in distinguishing real data from fake samples. This competition drives the generator to create increasingly realistic data.

$$\mathcal{L}_{GAN}(G, D) = \min_G \max_D \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[1 - \log D(G(z))] \tag{1}$$

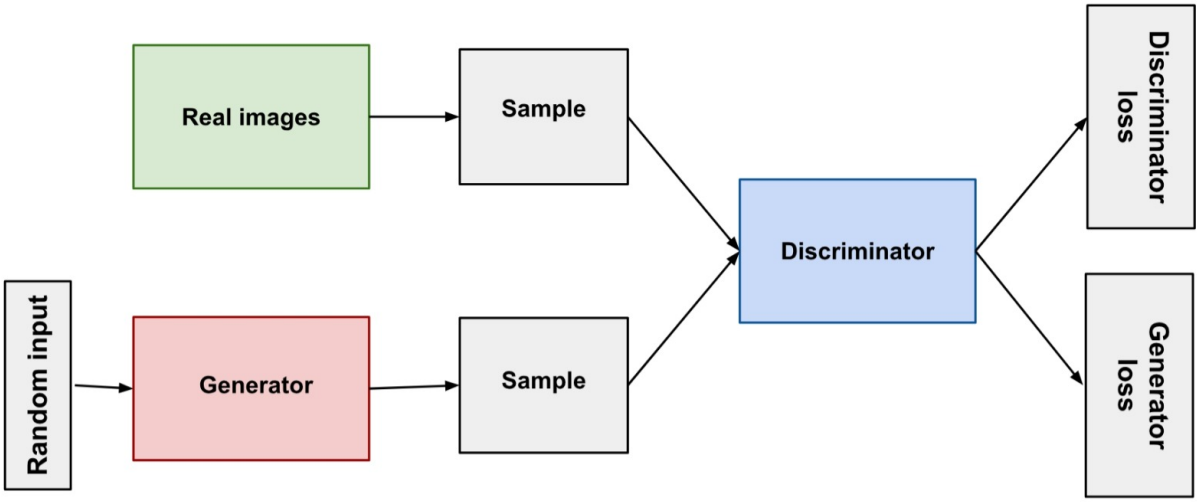


Figure 4: Generative Adversarial Network [15]

Conditional GAN

One variant of the GAN architecture is the Conditional GAN (cGAN) , which incorporate conditional information into the generative process. The main difference between traditional GANs and cGANs is that cGANs take additional input variables, known as conditional information, to guide the generation process, as opposed to generating samples from random noise [16]. In cGANs, the generator network receives both random noise and the conditional information as input, where the conditional information could be any additional data that provides constraints or context for generating the desired output. For example, in image generation tasks, the conditional information could be class labels or input images with specific attributes. The discriminator network in cGANs also receives the conditional information along with the generated or real samples, and it learns to distinguish between real and generated samples conditioned on the provided information. By incorporating conditional information, cGANs enable more precise control over the generated samples, which allows them to generate samples that align with specific conditions or exhibit desired characteristics. This makes cGANs suitable for various applications, such as image-to-image translation, text-to-image synthesis, and style transfer.

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y}[logD(x, y)] + \mathbb{E}_x[log(1 - D(x, G(x)] \tag{2}$$

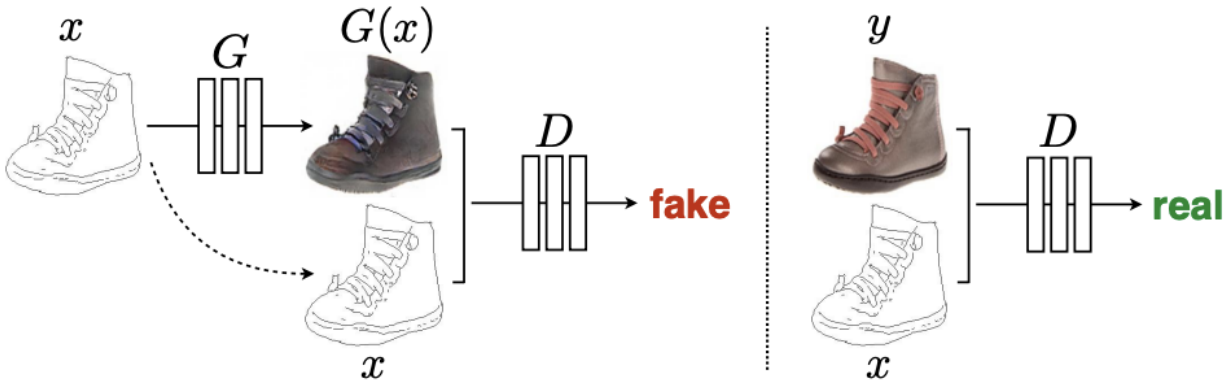


Figure 5: Pix2Pix Model

An application of the cGAN architecture is the Pix2Pix model, which is used in image-to-image translation tasks [17]. For instance, Pix2Pix focuses on the problem of transforming one type of image into another, such as converting sketches to realistic images or turning day-time scenes into night-time scenes (see Figure 5). Based on the cGAN architecture mentioned above, the generator of a Pix2Pix network takes an input image and aims to generate a corresponding output image that matches the desired translation, while the discriminator tries to distinguish between the generated output images and the real target images. The original Pix2Pix model employs a UNet generator with skip connections, while using a 70x70 PatchGAN discriminator to distinguish between real and fake tuples of images in domain A and B. The Pix2Pix model incorporates the L1 loss in the cGAN loss (Equation 2) to create the objective as shown in Equation 3.

$$\mathcal{L}_{pix2pix}(G, D) = \min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G) \tag{3}$$

This is a model architecture that has seen numerous applications in computer vision and image translation tasks, and provides a reliable foundation for the task at hand.

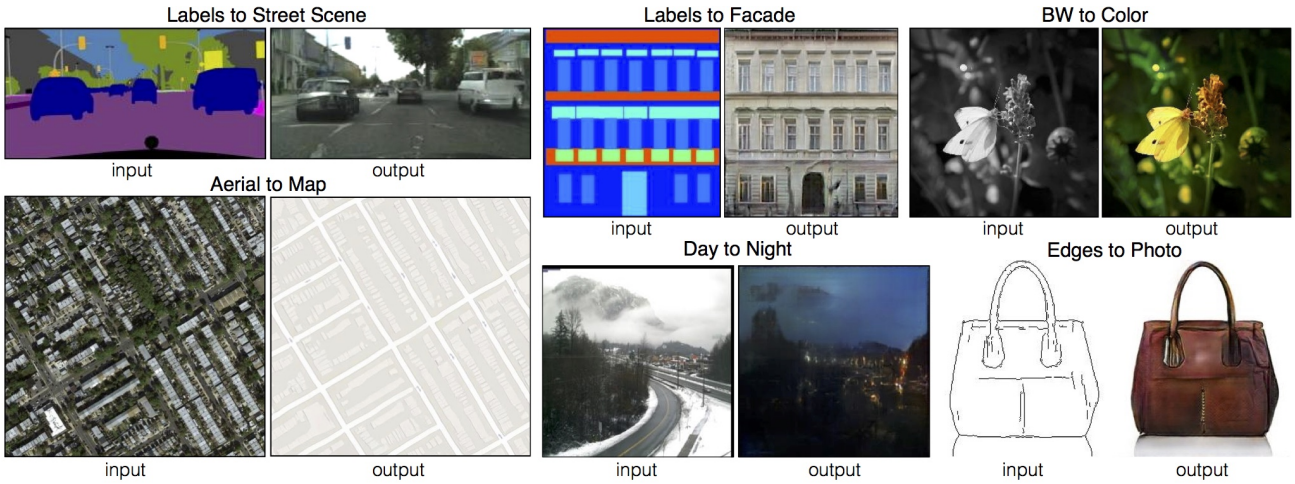


Figure 6: Pix2Pix Examples [17]

CycleGAN

Another GAN architecture that can be explored is the CycleGAN [18]. CycleGAN is a GAN architecture designed for unsupervised image-to-image translation, enabling the transformation of images from one domain to another without paired training data. Unlike Pix2Pix, which requires paired images, CycleGAN focuses on learning the mapping between two domains using unpaired images. The key idea behind CycleGAN is the introduction of cycle consistency, where it is assumed that if an image from domain A is translated to domain B and then translated back to domain A, it should remain similar to the original image - i.e., $F(G(x)) \approx x$ and $G(F(y)) \approx y$. This concept is leveraged to enforce a cycle consistency loss (Equation 4), which encourages the model to produce consistent translations in both directions.

$$\mathcal{L}_{cyc}(G, F) = \mathbb{E}_{x \sim p_{data}(x)}[\|F(G(x)) - x\|] + \mathbb{E}_{y \sim p_{data}(y)}[\|G(F(y)) - y\|] \tag{4}$$

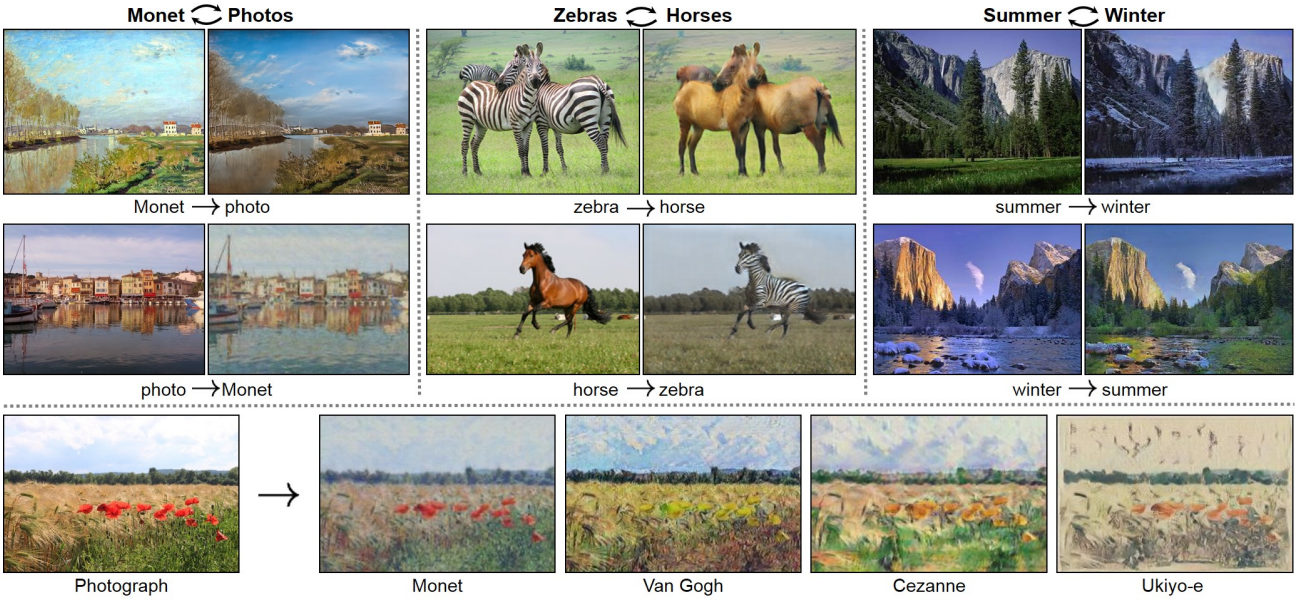


Figure 7: CycleGAN [18]

A key difference between CycleGAN and traditional GAN architectures is that CycleGANs have two generators and two discriminators. One generator focuses on the transformation from domain A to domain B, effectively learning to convert images from one domain to another. Simultaneously, the other generator works in the reverse direction, mapping domain B back to domain A. This bidirectional approach equips CycleGAN with the ability to comprehend the complex relationships and mappings between the two domains comprehensively.

The two sets of discriminators complement this setup. While one set evaluates the authenticity of generated images in domain B when compared to the ground truth, the other evaluates the fidelity of the translated images from domain B back to domain A. This duality ensures that the model not only produces convincing translations but also maintains the quality and consistency of the transformations in both directions. This concept of cycle consistency between between the generators and discriminators is reflected in the CycleGAN's objective function, as shown in Equation 5.

$$\mathcal{L}_{CycleGAN}(G, F, D_x, D_y) = \mathcal{L}_{GAN}(G, D_y) + \mathcal{L}_{GAN}(F, D_x) + \mathcal{L}_{cyc}(G, F) \quad (5)$$

As a result, CycleGAN stands as a versatile and powerful GAN architecture, finding applications in artistic style transfer, image-to-image translation tasks, domain adaptation, and various scenarios where unpaired data needs to be seamlessly transformed and adapted between domains.

2.2 Related Works

Now that some of the key concepts relevant to the project have been discussed, some past studies that have been conducted in related topics will be reviewed.

Decoding the Brain

There have been numerous attempts at directly decoding neural information from the brain. This is typically done by exposing a group of subjects to visual or auditory stimuli and recording how the brain reacts using EEG or fMRI. The recordings can then be used for various tasks, including but not limited to, stimulus reconstruction and emotion recognition. Some of the methodologies used for such tasks will be reviewed in this section, and these methodologies will then be evaluated to help design a methodology suitable for this specific project. It is worth noting that both EEG and fMRI (and a combination of both) are a valid and widely used source for stimulus reconstruction in the field of neural decoding, but methodologies involving only EEG will be reviewed here, as that is what the scope of the project is concerned with.

In the past, many of the studies that have been conducted in this area were limited to classification and recognition tasks. However, with the advent of deep learning and neural networks, most notably Convolutional Neural Networks (CNN), Generative Adversarial Networks (GAN), and Long-Short-Term Memory (LSTM) models, attempts were made to go beyond classification tasks, and more emphasis was placed in recreating the actual stimulus from signals received from the brain. This review will therefore also put more emphasis on studies that apply deep learning techniques in their methodologies.

One example where EEG recordings were used for stimulus music classification was the GuessTheMusic project [3]. Here, the researchers proposed a method for identifying songs based on EEG responses, employing a Convolutional Neural Network. Various methods were tested, including representing the data in frequency and time domains, and CNN architectures. This achieved a 84.96% accuracy rate in classifying songs based on EEG data, indicating that specific brain patterns are associated with listening to different songs, even with variations among individuals. These results prove that there is in fact a relationship between the external musical stimulus and the brain's reaction, making the topic of reconstruction an intriguing avenue to explore. However, this alone does not offer much insight with regards to the methodology or feasibility of reconstruction.

Another notable study is the Brain2Image model, which attempts to recover images seen by humans from the EEG signals by developing a deep learning framework that involves an

LSTM encoder and a VAE/GAN decoder, where the two decoder networks were compared for performance [19]. The authors firstly collected EEG data from six subjects while they were shown images of 2000 objects. The EEG data was then fed into an encoder that consisted of an LSTM layer followed by a nonlinear layer, in order to represent the EEG series as a more compact and discriminative feature vector to be used when training the decoder network(s). To generate the actual images, the authors compared two different networks – a variational autoencoder (VAE) and a generative adversarial network (GAN). The results of this investigation showed that GANs outperform VAEs when the inception scores are compared, meaning that the GAN decoder was able to generate images that were more resemblant of the object class. The overall inception score of the GAN decoder was 5.07 versus the VAE's 4.49.

There are a few conclusions that can be drawn from this study that is relevant to this project. The first is that GANs seem to be a very strong candidate when it comes to deciding on a generative model. This is supported by further studies done on a similar topic [18, 19], where different variations of GANs look to be a favoured choice of network when generating images. The other point worth noting is the choice of loss function and evaluation. The authors here used the sum of the Kullback-Leibler divergence and image generation loss (mean squared error between the generated image and the target) for the VAE decoder. For the GAN, they use the negative log-likelihood for the discriminator loss and the generator loss. These are all points that can be considered for the model used in this project.

Another, perhaps more relevant study attempts to reconstruct music from joint fMRI-EEG recordings, where the participants were listening to a set of synthetic and classical music while their fMRI and EEG signals were being recorded [17]. In this study, a bidirectional LSTM was trained on the musical stimuli and fMRI-informed EEG features, which was then used to recover the original music. This resulted in a mean rank accuracy of 71.8%. The results were evaluated using Pearson's correlation coefficient as well as the structural similarity. A major takeaway from this study is the choice of neural network, which was the bidirectional LSTM. This is an appealing option as the network would be able to capture and learn representations from both past and future signals, which allows it to process sequences such as EEG in an accurate and robust manner.

Another related study in the music stimulus reconstruction domain is the EEG2MEL model, which used EEG data from the NMED-T (Tempo) and NMED-H (Hindi music) datasets to train a sequential CNN-based deep regressor to output a target spectrogram of the input EEG [20]. In this study, various input-target combinations were compared, such as raw voltage or Power Spectral Density (PSD) for representation of the EEG input, and linear-spectrogram to mel-spectrogram for the output. Results showed that the PSD input to mel-spectrogram output achieved the highest accuracy of 80.80%. The CNN regressor was evaluated based on two criteria – the Structural Similarity Index Measure (SSIM) and Peak Signal to Noise Ratio (PSNR), both of which are commonly used for evaluating the quality of images [24]. Once a mel-spectrogram is obtained, this was converted back to raw audio using the Griffin-Lim algorithm.

This study acts as the main inspiration as to the methodology of this project. As per the discussion on mel-spectrograms above, the conversion of the EEG signals to spectrograms and use of mel-spectrograms as output recasts the problem of dealing with raw audio signals and has yielded satisfactory results. This will therefore be the general approach to the problem at

hand. However, there are limitations to the EEG2MEL model that can be improved. The first modification that can be made is the neural network itself. Despite a convolutional network being a straightforward approach that is proven to work, other avenues such as GANs, and even attention models, have proven to match and even outperform convolutional networks in other domains, which make them a very appealing option. Another potential change is in the evaluation metrics, although this largely depends on the network itself. One potential addition is an evaluation of the audio samples by audio inspection by humans, which would provide a subjective analysis of the final output.

As such, a number of studies have been conducted in the past to decode neural information, each with their own methodologies and results. It has also been shown that many of these methodologies have their respective limitations, and this project aims to combine some of these ideas and integrating new ones into the methodology, which will be discussed in the section that follows.

2.3 Legal, Social, and Ethical Considerations

Although this project is by nature a machine learning project with no serious ethical issues, there are a few points that must be acknowledged for the sake of academic and ethical integrity. First of all, the datasets used in this project were all open-sourced, meaning that they are free to be used with acknowledgement. The nature of EEG data is obviously one that is collected from people, and the necessary measures have been taken by the creators of the datasets to anonymise the participants and ensure transparency in the use of the dataset. Another aspect of this project that requires human participation is the user study conducted to evaluate the results, where participants have been ensured that the results are completely anonymous and for research purposes only.

Chapter 3

Method

Achieving the goals of this project entails several steps, including preprocessing the data, setting up a deep learning training and testing pipeline, extracting the results, and finally evaluating the results. In this case, the training and testing involved two steps. The first step consisted of training and evaluating the baseline model (specifically, a straightforward CNN generator) in order to obtain initial results that can be used for comparison. The second step was to then train the GAN model, which was expected to outperform the CNN, and compare the results to the baseline while also optimising its results. The execution of each of these steps are crucial to the success of the project, and they will be discussed in detail in the following sections.

3.1 Dataset

The dataset that will be used to train the model is arguably the most important aspect of any deep learning project, as it will determine the training and testing methodology and the quality of the final output. This is especially true for projects involving brain data such as EEG, since the data can be very noisy without the proper preprocessing steps. Two datasets will considered for the purpose of this project: Daly et al.'s Film Music dataset [21] and Losorelli et al.'s NMED-T dataset [22]. There are various aspects of each dataset that need to be taken into consideration when choosing the final dataset to work with, such as the amount of data contained in the dataset, and the type and quality of the preprocessing methodology applied to the dataset, if any.

Once the dataset to be used was identified, the data must first be processed in a format that can be used to train a neural network. As discussed, since the neural network is a GAN designed for image-to-image translation, both the EEG recordings and audio stimuli need to be converted into their respective visual representations (see Chapter 2.1.1). The steps necessary to achieve this is also outlined below, along with implementation details and their justifications.

3.1.1 Film Music Dataset

The Film Music dataset [21] is a collection of EEG recordings of 31 participants listening to 10 second clips of film music of various genres, which have been known to evoke different emotions in participants [23]. The recordings were done using a standard 10/20 configuration with 19 electrodes. However, the recordings provided in this dataset were raw, without any offline (post-recording) preprocessing applied, meaning that the data needed to go through some extra steps before it was suitable for use in this project.

Preprocessing Pipeline

There exists a variety of methodologies and libraries when it comes to EEG preprocessing pipelines, as outlined in [24]. The pipeline selected for preprocessing the Film Music dataset follows that outlined by EEGLAB [25], which suggests the order of preprocessing (high pass filtering, resampling) and artifact rejection (removal of bad channels, automated rejection, and independent component analysis for component removal). MATLAB's EEGLAB plugin was used to apply all the preprocessing steps.

The first step of the preprocessing pipeline is to import the data and annotate the recording using the events timestamps provided by the dataset. This indicates which section of the recording corresponds to which stimulus in the given set. What follows is locating the channels with the respective electrodes using the channel map, which allows us to visualise and correspond the electrodes on the scalp with the channels.

Once this is done, the preprocessing begins by firstly applying a high-pass filter on the data. This is to remove linear trends in the data, and to obtain good ICA decompositions. A high pass filter of 1 Hz was applied, following [26]. This was followed by resampling the data at 128 Hz (the original sampling rate was 1000 Hz), as this would allow for faster processing in subsequent steps.

The next step is the removal of artifacts, which is arguably the most crucial step in the pipeline. The EEG is a data collection method that involves a lot of artifacts - this includes muscular artifacts coming from movement from the participants, eye blinks, pulses, as well as other external and internal factors. It is therefore crucial to remove/reduce these artifacts to obtain clean data while also preserving the integrity of the data. EEGLAB provides various tools for this purpose. This includes the Clean Rawdata plugin, which is an algorithm that rejects bad channels and portions of data based on the parameters provided [27]. What is left then is to perform Independent Component Analysis (ICA) to remove artifacts embedded in the data. ICA is a statistical signal processing method that aims to separate a multivariate signal (such as EEG data) into additive, independent components. It assumes that the observed data is a linear combination of independent source signals, some of which may be neural activity (signal of interest) and others artifacts (unwanted noise). ICA decomposes the mixed EEG data into a set of independent components, with each component representing a linear combination of the original channels. To goal of ICA is to identify components that are statistically independent, allowing which then enables the separation of signals into true neural signals from unwanted artifacts. EEGLAB provides a number of different ICA algorithms, such as *runica*, which can used to identify the components, and these components can be accepted or rejected based on their likelihood of being an artifact (see Figure 8).

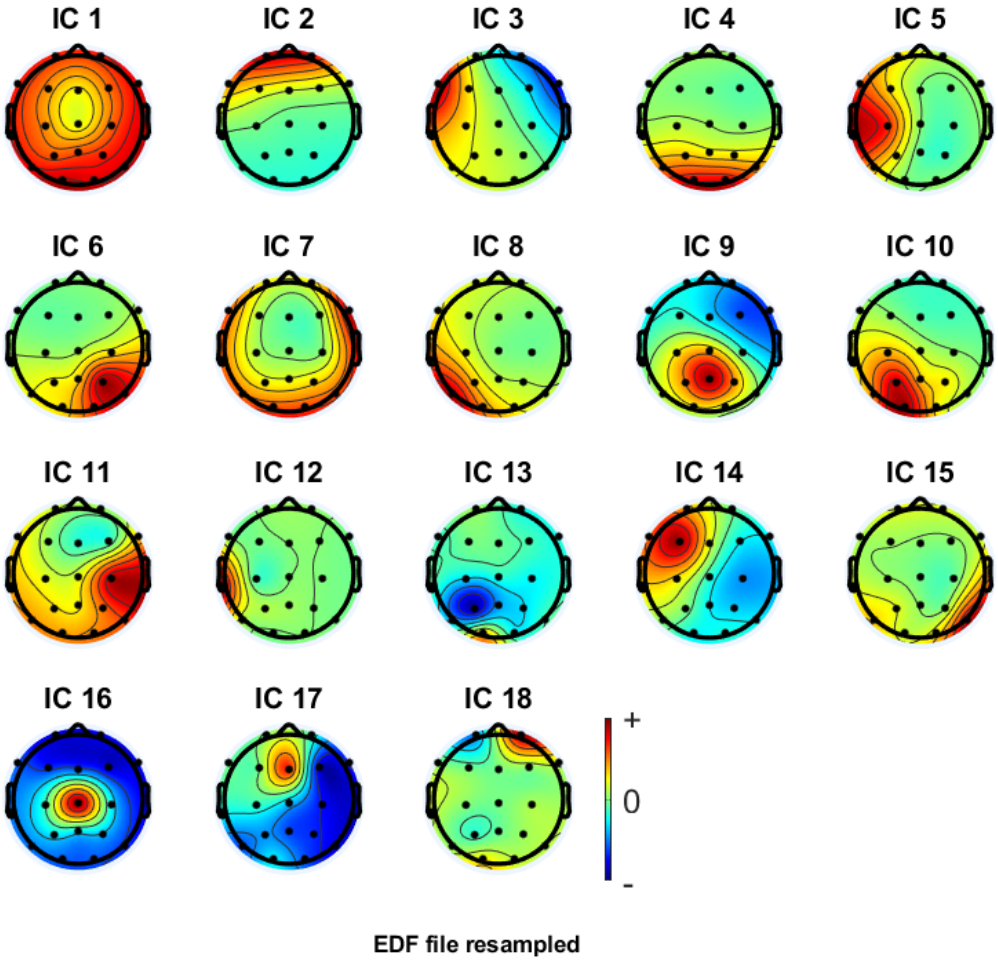


Figure 8: Example ICA Components. It can be seen, for example, that Component 3 is most likely an ocular artifact, which can be rejected.

The final step is to segment the EEG data into one-second snippets, so that they can be converted to spectrograms that will subsequently be used to train the neural network. The final dataset will therefore contain segments of EEG data that have a corresponding audio stimulus. However, segments that have been rejected from EEGLAB's automated rejection algorithm have been omitted, meaning that some of the data has been lost. This dataset would end up providing around 15,000 segments of EEG and stimulus samples that could be used for training the model.

3.1.2 NMED-T Dataset

The NMED-T dataset is a collection of EEG recordings of participants listening to a selected group of stimuli [22] (see Table 1). In this experiment, 20 participants were exposed to 10 different stimulus music in a single run, each with a length between 270 to 300 seconds. The EEG responses were recorded using the Electrical Geodesics, Inc. GES300 system, with 128 electrodes.

Stimulus No.	Title	Artist	Length (min)
1	First Fires	Bonobo	4:38
2	Oino	LA Priest	4:31
3	Tiptoes	Daedelus	4:36
4	Careless Love	Croquet Club	4:54
5	Lebanese Blonde	Thievery Corporation	4:49
6	Canopée	Polo & Pan	4:36
7	Doing Yoga	Kazy Lambist	4:52
8	Until the Sun Needs to Rise	Rüfüs Du Sol	4:52
9	Silent Shout	The Knife	4:54
10	The Last Thing You Should Do	David Bowie	4:58

Table 1: List of stimuli used in NMED-T dataset

Unlike the Film Music dataset, NMED-T was provided having already been preprocessed. The collected data was then preprocessed with a low-pass filter of 50 Hz, followed by removal of bad channels and Independent Component Analysis for removal of ocular and muscular artifacts.

In the end, the dataset selected for the project was NMED-T, due to several factors. Firstly, the NMED-T dataset is set up with 125 electrodes versus the 19 of Film Music. This means that the NMED-T recording would contain more refined details in its recorded signals, which means there are potentially more features to be captured. The other advantage of NMED-T was that the preprocessing steps were done reliably by professionals, whereas the raw data in Daly et al.'s dataset would have to preprocessed manually, which would be more prone to errors despite providing more license over how the preprocessing is done. NMED-T also provided more data in terms of pure quantity, as it contains around 57,000 segments of EEG and audio, as opposed to the 15,000 of Film Music.

3.1.3 Data Preparation

EEG Spectrogram

As discussed in the previous chapter, the segmented EEG data will be converted into spectrograms after computing the PSD. The other option was to use the raw EEG signals by putting the channels in the y-axis with time on the x-axis, which would be a more raw, direct conversion of the EEG. However, using the PSD in the spectrogram was in fact proven to perform better when used for neural decoding purposes [20], which is the reason it has been chosen over the raw EEG in this project.

The PSD was computed using Welch's method from the raw EEG values [28], which was then used to produce a mapping between the time and frequency bins of the EEG recordings, with the variation in voltages across the channels represented in the density of the graph. Python's MNE library has a built in functionality for calculating the PSD using Welch's method, which was used to facilitate the computation [29].

In order to compute the PSD graphs, the recordings were firstly segmented into 1 second snippets. Since the NMED-T dataset was recorded with a sampling rate of 125 Hz, this meant that there are 125 samples for each 1-second snippet, with 125 EEG channels in each second of the recording. This segmentation results in the dataset providing 57280 images combined for the EEG recordings.

One thing to keep in mind is that, for the purpose of facilitating training of the network model, the input and output images were made to have the same shape. For this reason, the input EEG spectrograms were padded around the edges to match the size of 128x128. The ResNet architecture also requires the image size to be a multiple of 4, which the 128x128 size already fulfills.

Stimulus

The stimuli corresponding to the EEG recordings were also to be converted to image representations in the form of mel-spectrograms, as discussed in Chapter 2. This was done by firstly segmenting the full audio clips into 1 second snippets, which, in the case of the NMED-T dataset, correspondingly resulted in 57280 audio samples. These audio samples were then individually converted to mel-spectrograms using the Librosa library in Python, which provides functionalities for various audio analysis tasks including mel-spectrogram conversion [30].

To stay consistent with the EEG spectrograms in terms of the image dimensions, the mel-spectrograms were created using a hop size of 173, in order to match the 128x128 dimensions required by the network.

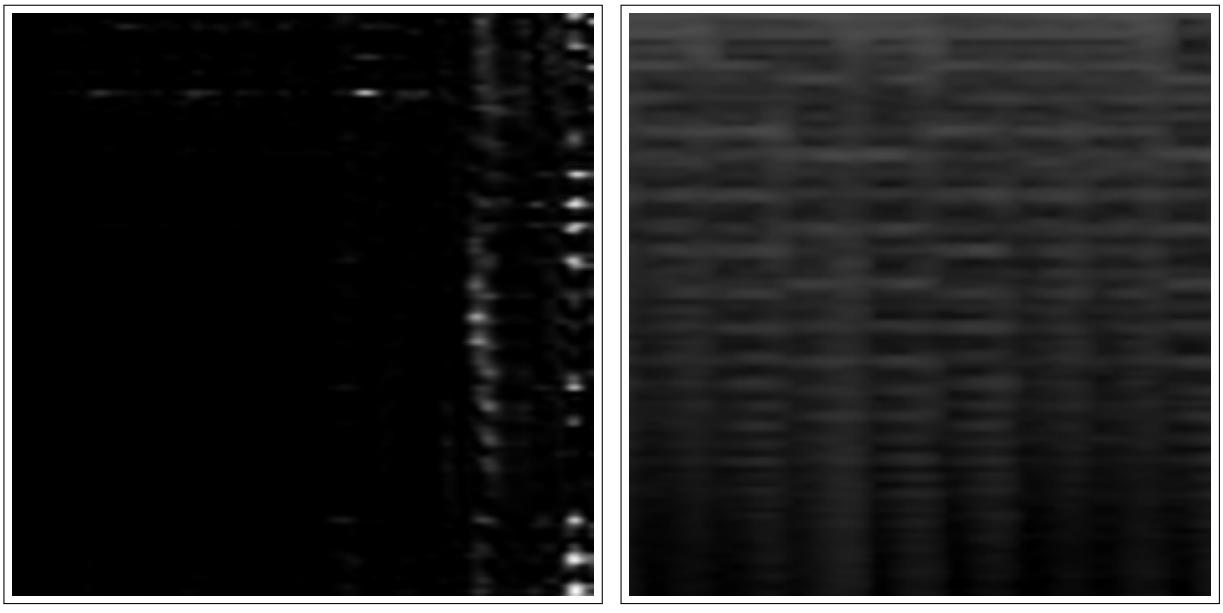


Figure 9: (a) The EEG spectrogram of a one-second segment of the recording in grayscale. (b) A mel-spectrogram of the corresponding stimulus in grayscale.

3.1.4 Neural Processing Delay

An interesting phenomenon with regards to the nature of brain signals and neural processing is worth noting at this point. It is sensible to assume, and thus has been largely proven, that there is a delay in time when the brain processes information provided by external stimuli [31]. For instance, when a subject is exposed to a auditory stimulus, there is a lag in time between the sound being played, the vibrations of the sound waves reaching the eardrums, and the sound being processed by the neurons and thus evoking an overall reaction in the cognitive faculties of the brain. It is predicted that the brain takes approximately 150 to 200 ms to process sound [32], although this depends on various factors such as the subject's age. This is a relatively significant delay in this context, since we are dealing with 1-second segments of EEG and audio samples. A delay of 200 ms would mean that there is a margin of error of up to 20% for a mismatch between the EEG samples and stimulus, which needs to be taken into consideration.

One straight-forward way of taking this into account is by shifting the EEG dataset by certain time lag, say 200 ms. This would make the EEG samples align with the theoretical delay in neural processing, which would make the correspondence between the EEG and audio stimulus more accurate. Although this didn't have an apparent impact on the results in terms of the metrics, it is a necessary consideration that would make the pipeline more representative of reality.

3.2 Model Implementation

This section outlines the model architectures and overall training methodologies used, such as evaluation metrics, hyperparameter tuning, and regularisation techniques to train and optimise the model. The code used to build and train the model has been adapted from the original Pix2Pix and CycleGAN papers [17, 18]. Training and testing of the models were performed on Imperial College London's High Performance Computing Cluster, using Nvidia RTX6000 GPUs. The results and analysis of how these decisions and application of techniques impacted the model will be discussed in detail in Chapter 4.

3.2.1 Network Models

CNN

The first major endeavor of the project was to build a baseline model, which would help determine what improvements need to be made and how much further implementation and training can improve results. To obtain baseline results to use for comparison, a convolutional neural network (CNN) was trained to generate mel-spectrograms based on eeg spectrogram inputs. Various combinations of CNN architectures were tested, including 9-block and 6-block ResNet and UNet architectures, to determine which architecture would be optimal for the final GAN generator. Results showed that the 9-block ResNet architecture performed the best overall, as it will be shown below in Chapter 4. The CNN used an L1 loss, where the Mean Absolute Error (MAE) is minimised between the target and generated outputs (see equation 6).

$$\mathcal{L}_1(G) = \sum_{i=1}^{D} |x_i - y_i| \tag{6}$$

Conditional GAN

The model that is an improvement on the vanilla convolutional network is the Conditional GAN for image-to-image translation - specificially the Pix2Pix model [17]. As mentioned in Chapter 2, the task of mel-spectrogram generation is an image-to-image translation task. In this case, since we are working with paired data (i.e. the images in the X domain are labeled with corresponding ground truth in the Y domain), the conditional approach can be used. The use of such conditional information allows the network to learn the mapping between the input and target domains, which is a more efficient and accurate approach than the straight-forward CNN generator.

In order to optimise the performance of the model, various combinations of architectures and techniques will be explored. This not only includes the typical hyperparameters, but also additional regularisation techniques that can improve the model. These techniques will be discussed in detail in the next section.

Conditional CycleGAN

Here we propose a model architecture that is a potential improvement on the traditional Pix2Pix cGAN model, namely the Conditional CycleGAN (cCGAN). The idea of introducing cycle consistency for image-to-image translation tasks has seen many applications, and this seems to be a promising avenue for EEG spectrogram to mel-spectrogram translation as well. Although this model has not been tested to the same extent, it is a model that has been theorised to outperform the cGAN.

CycleGAN, as discussed in Chapter 2, is a GAN architecture used for unpaired image-to-image translation across domains, using two generators and two disciminators and the cycle consistency loss (see Figure 10). However, the original CycleGAN was designed for *unpaired data*, which, since we are using a paired dataset, creates the issue of under-constraining the problem. Therefore, we introduce the idea of combining the conditional information provided by the paired dataset with the notion of cycle consistency.

This model would largely follow the structure of the CycleGAN, with two generators and discriminators, each with the goal of generating realistic images in the target domain - i.e., G(x) and F(y). In contrast to the CycleGAN, however, the ground truth label in the target domain for each of the images will be visible for the generators and discriminators. This allows the label image to act as the conditional information, which employing the conditional approach as discussed in Chapter 2.1.2. This would be combined with the cycle consistency loss, which would enforce the similarity between the generated and ground truth images.

This model is therefore one that incorporates both aspects of conditioning and cycle consistency - a Conditional CycleGAN. The idea of incorporating cycle consistency in a conditional setting has seen some, albeit limited use [33, 34]. Although this is a proposed improvement on the Pix2Pix GAN, it has not been thoroughly tested due to time and resource constraints.

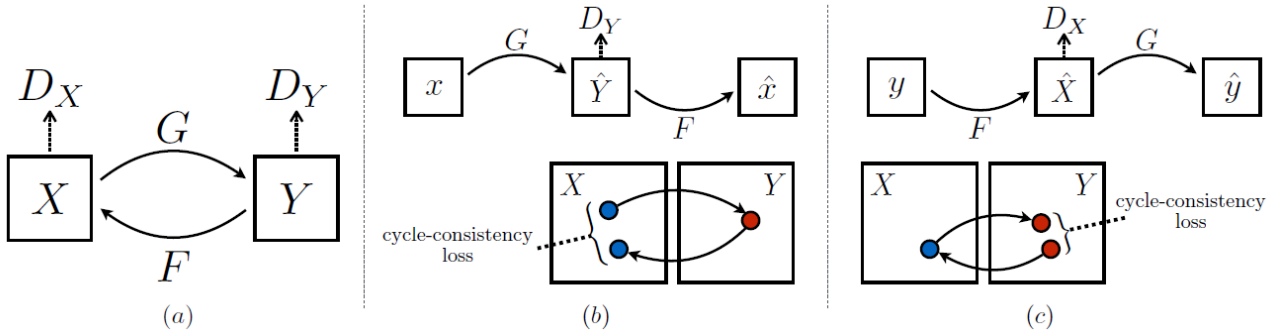


Figure 10: CycleGAN Architecture [18]

3.2.2 Training Techniques

In terms of initialising and optimising the model, a lot of the decisions were made based on extant literature [35], as well some preliminary testing on a smaller dataset. For instance, all of the network weights were initialised using the Kaiming method [36], which performs better when used in non-linear activation functions, such as ReLU (which is used in both UNet and ResNet architectures). The Adam algorithm was used for the optimiser, which is a widely used stochastic gradient descent algorithm [37]. In addition, following what is common practice in machine learning tasks, the dataset was split into train/validation/test sets, with a split of 8:1:1. This allows the model being trained to be evaluated at the end of every epoch of training, so its performance can be monitored.

Other techniques that have been explored include dropout and a learning rate scheduler. Using dropout ensures that the model will generalise well and not be overfitting on the data, by deactivating a fraction of neurons or units in a layer during each forward and backward pass of training [38]. This introducing randomness and diversity during training, effectively ending up training an ensemble of neural networks within a single network architecture. The network was trained with a dropout rate of 0.5, which is the rate used in the original Pix2Pix papers.

Another feature used to improve training is a learning rate scheduler. The learning rate is a hyperparameter of a neural network used in the stochastic gradient descent algorithm that determines the size of steps or updates made to the model's parameters during the training process, which affects the speed and precision of convergence. A learning rate scheduler allows the model to dynamically adjust the learning rate during training according to a predefined schedule or based on certain conditions. This leads to faster convergence and improved generalisation, as it allows the network to escape local minima and optimise with more stability. A linear scheduler was used for the networks in this project, while the learning rate was initialised at 0.0002.

Label Smoothing is a another regularisation technique for deep learning models that introduces noise for the labels [39]. Instead of having one-hot encoded labels (i.e., binary values), the labels used for training will be more of a probability distribution, so that the model will be less confident in making its predictions. This prevents the model from overfitting to the training data, which can potentially improve the outputs from the model. In particular, one-sided label smoothing is a technique that is found to improve GAN training (see equation 7)

[40]. This means that only the positive label (α) is smoothed, while the negative label (β) is set to 0. The effects of label smoothing will be explored in more depth, as the cGAN will be trained with and without label smoothing to determine whether or not it improves training.

$$D(x) = \frac{\alpha p_{data}(x) + \beta p_{model}(x)}{p_{data}(x) p_{model}(x)} \tag{7}$$

α : positive label
β : negative label

3.2.3 Evaluation Metrics

Ideally, evaluation of the performance of the models would be done in two stages - on the generated mel-spectrogram images, as well as the final generated audio clips. To achieve this, it was first necessary to decide which evaluation metrics to use at each stage, each of which will be discussed in detail below. Various metrics have been used in the past for image quality and similarity analysis, such as the Inception Score and the Frechet Inception Distance, among others. The two main candidates for evaluating the generated mel-spectrograms were the Peak Signal to Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM), both of which have been extensively employed in image generation and analysis tasks [41]. Both of these metrics were tested as evaluation metrics for the generative models to more robustly evaluate the models' performance on the output images. Finally, the Mel Cepstral Distortion (MCD) between the original and generated audio was used to evaluate the final output. These metrics provide ways to objectively inspect the output images, which is a necessary step in evaluating the model's performance. However, it will subsequently be shown that these metrics are not an absolute measure of overall output quality, and subjective analyses is also required to extensively evaluate the model's performance. This will be discussed in further detail in Chapter 5.

Peak Signal to Noise Ratio

The first evaluation metric that was used for the generated images was the Peak Signal to Noise Ratio (PSNR) [42]. PSNR is a metric that is commonly employed in image and video processing tasks to evaluate the impact of compression algorithms or image denoising techniques, and is also a widely used metric for quantifying the quality of reconstructed images. It measures the fidelity of an image by comparing it to a reference image, typically an original or ground truth (in this case, the original mel-spectrogram). This is done by calculating the ratio of the peak signal power to the mean squared error (MSE) between the original and the reconstructed image (see Equation 8). A higher PSNR value ($\sim$40 dB) indicates a closer match between the images and implies lower distortion, while a lower value ($\sim$20 dB) implies dissimilarity and more noise.

$$PSNR(x, y) = \frac{10 \log_{10}[\max(\max(x), \max(y))]^2}{|x - y^2|} \tag{8}$$

Structural Similarity Index Measure

Another metric that was employed to analyse the models' performance was the Structural Similarity Index Measure (SSIM) [43]. Structural Similarity Index is a widely utilized metric for evaluating the similarity and quality of images. Unlike PSNR, SSIM also takes into account both luminance and structural information by comparing local patterns of pixel intensities in the ground truth and generated images, making it more aligned with human visual perception. It quantifies the similarity between these patterns and produces a value between -1 and 1, where values closer to 1 indicate greater similarity. See Equation 9.

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1) + (2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{9}$$

μ_x : average of x
μ_y : average of y
σ_x^2 : variance of x
σ_y^2 : variance of y
C_1, C_2 : $(k_1L)^2$, $(k_2L)^2$, where
k_1, k_2 : 0.01, 0.03 (by default)
L : the dynamic range of pixel values ($2^{bits/pixel}$)

Mel Cepstral Distortion

The two evaluation metrics above are measures that assess the similarity of images, which, in the context of this project, is not enough. This is for two reasons. Firstly, mel-spectrograms are not natural images, so the PSNR and SSIM metrics may have limitations when assessing the model's outputs. Secondly, the ultimate goal of the model is to generate audio, not images. Although a high PSNR/SSIM score would be an indication of proximity in the generated and ground truth mel-spectrograms, this does not guarantee that the resemblance is relfected in the actual audio.

Keeping this in mind, the final evaluation metric that will be used is the Mel Cepstral Distortion (MCD), which is a measure often used to assess the quality of speech synthesis or voice conversion systems [44, 45]. It is calculated by quantifying the similarity between two sets of Mel Frequency Cepstral Coefficients (MFCCs), which are features extracted from audio signals to represent their spectral characteristics. It calculates the root mean square of the differences between corresponding MFCCs, considering both spectral shape and temporal alignment. A lower MCD value indicates higher similarity and better quality between the synthesized/converted and reference speech.

$$MCD(x, y) = \frac{10\sqrt{2}}{\ln 10}\frac{1}{T}\sum_{t=1}^{T}\sqrt{\sum_{i}(x_{ti} - y_{ti})^2} \tag{10}$$

x, y : Mel Cepstral Coefficients
T : number of time slices/frames

The use of the MCD as an additional evaluation metric to the PSNR and SSIM provides a way of assessing the final audio output, which makes the evaluation of the model's performance more robust and reflective of what the ultimate goal is. However, it must also be noted that the MCD is often used in analysis of speech synthesis tasks, not for musical notes. This could prove to be an important distinction when using the MCD as a metric.

These evaluation metrics provide an objective measure of the quality and similarity of the outputs, which demonstrates how the model can be improved with subsequent additions of techniques and iterations. However, as mentioned, these metrics are by no means an absolute measure of the models performance, which is why subjective analysis such as visual inspection also needs to be taken account when evaluating the model's results.

3.2.4 Hyperparameter Tuning

Given the evaluation metrics, it is then essential to choose the hyperparameters, such as batch size, number of layers, and type of network that would result in an optimised model. Various combinations of state-of-the-art techniques were employed in search of these optimal hyperparameters. One of such techniques was Ray Tune library's HyperOptSearch [46, 47], which uses a Tree of Parzen Estimators (TPE) to approximate the performance of hyperparameters based on past measurements, subsequently choosing new hyperparameters to test based on this model. This was used in combination with the Async Successive Halving Algorithm scheduler [48], which is an early stopping algorithm that is used to tune hyperparameters efficiently.

Chapter 4

Results

In this chapter, the training results of each of the models will be presented and analysed, in order to evaluate the models' performance and discuss how they can be optimised. As discussed above, the two main metrics by which the models were evaluated were the Peak Signal to Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM). The baseline results were tested to identify which architecture to use for the cGAN generator, while the cGAN was further tested to optimise other parameters such as batch size and number of layers. The results were also evaluated using the Mel Cepstral Distance (MCD) of the final audio output, in order to provide a sense how the actual audio clip compares to the ground truth.

4.1 Baseline

The baseline CNN generator was trained with two main hyperparameters in mind - the architecture and batch size. The results of the baseline model show that the generators using ResNet 9 layers tend to out perform their UNet and ResNet 6 counterparts, although the difference itself is not too significant. Smaller batch sizes seem to also perform better in both metrics, though this too did not signify a large difference. Both the PSNR and SSIM values seemed to converge at around 100 epochs, after which there is some fluctuation of values. This could mean that any training beyond 100 epochs is sub-optimal in the case of the CNN model. It is also worth mentioning that the UNet model with batch size 32 performed almost as well the ResNet with batch size 8 in both metrics, which is unexpected.

Figure 11 shows the CNN model's performance in the selected evaluation metrics with different hyperparameter configurations. It can be seen that the metrics do improve with further training, until around epoch 100, where the trend becomes stationary or starts decreasing, which is potentially a sign of overfitting. Another major takeaway is that the lower batch sizes tend to perform better, with the single best performing architecture being the ResNet 9 block CNN. This result is supported by the Mel Cepstral Distortion values in table 2, which show that the ResNet 9 with batch size 8 has the best MCD score. The difference between these results are, however, not significant enough to conclude that one combination of hyperparameters is definitively better than the others - although it does give an indication of which avenues are more promising and can be further explored.

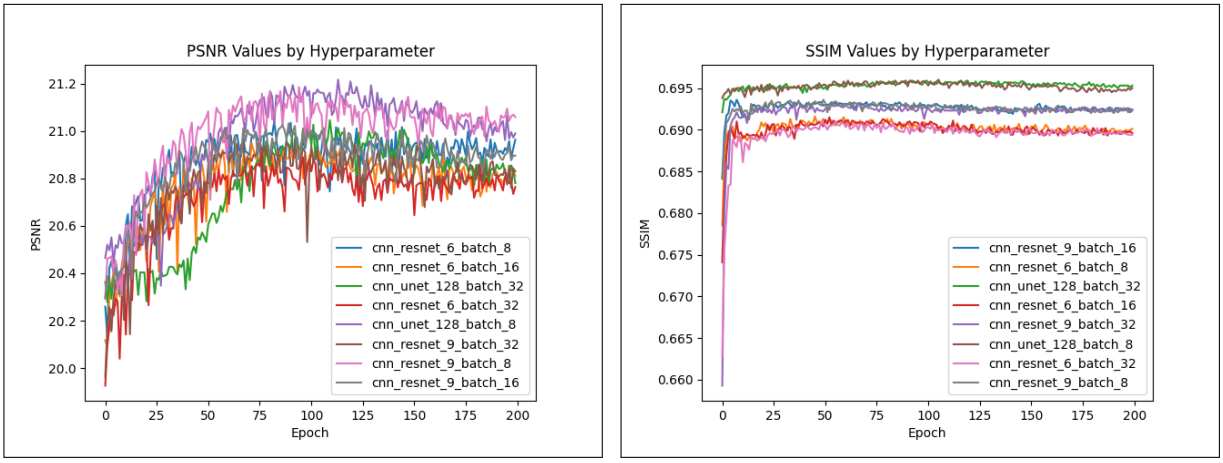


Figure 11: (a) PSNR values by Epoch for different hyperparameters of the CNN model (b) SSIM values by Epoch for different hyperparameters of the CNN model

Model	MCD Score
UNet, Batch Size 8	18.979
UNet, Batch Size 32	19.541
ResNet 6, Batch Size 8	18.494
ResNet 6, Batch Size 16	17.875
ResNet 6, Batch Size 32	17.965
ResNet 9, Batch Size 8	17.272
ResNet 9, Batch Size 16	17.975
ResNet 9, Batch Size 32	17.708

Table 2: Model Hyperparameters for Baseline Model with MCD Scores

The true significance of these baseline results are noticeable upon visual inspection of the output images. When visually inspecting the outputs from the CNN, the mel-spectrograms seem to have quite a large discrepancy in how they look. Figure 12 shows how the CNN output is in fact missing many of the features of the ground truth mel-spectrogram. It can be seen that the ground truth mel-spectrogram has more horizontal lines in various shades, which represents the power of the frequency at any given time point. The shape is reflective of the fact that musical notes are essentially continuous decibels of sound in a particular frequency. However, the outputs from the CNN seem to have a much smoother gradient when compared to the ground truth mel-spectrograms, which can signify either blurriness in the generated images, a lack of features captured by the network, or a combination of both.

These results leave a lot to be desired, although it did provide some use insight with regards to the configuration of the final model. They are also suitable to be used and compared with as a baseline, as we will see how the conditional GAN, along with various techniques, improve the quality of the generated images.

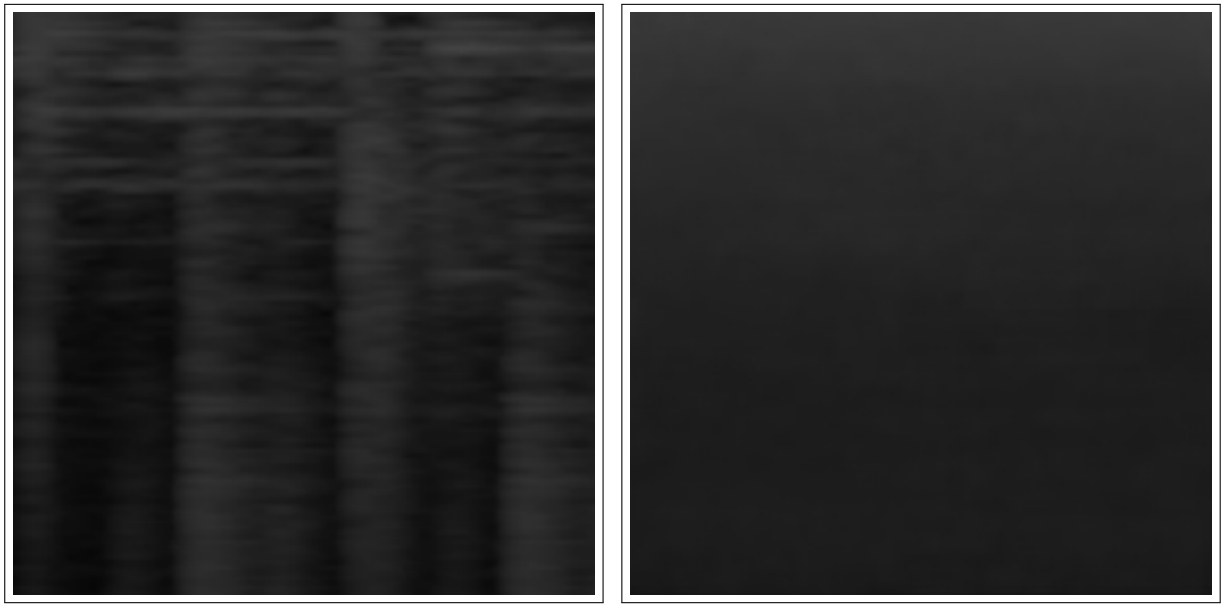


Figure 12: (a) A ground truth mel-spectrogram. (b) An output mel-spectrogram from the CNN.

4.2 Conditional GAN

The Conditional GAN, as outlined in Chapter 3.2, was trained and tested to be compared to the baseline results of the CNN. It was determined from the baseline results that the ResNet 9 architecture performs the best in terms the evaluation metrics, which is the reason why the cGAN's generator architecture was fixed to ResNet 9. The hyperparameters that were to be tuned in the cGAN were therefore the batch size and number of layers in the discriminator.

The results from the conditional GAN, in terms of the PSNR and SSIM metrics, were quite unexpected. It can be seen from the two graphs (Figure 13) that the cGAN actually scores slightly worse than the CNN model, which is contrary to the initial hypothesis that the cGAN is an improvement to the baseline. The values tend to be just below or around 20, which is a significant drop from the highest values achieved in the CNN of around 22. It can also be seen that the training is not very stable for some of the models, with the metrics fluctuating or having sub-par values for certain combinations of hyperparameters. This seems to be the case particularly for discriminator layers of more than 3, which suggests that a depth of 3 layers is optimal for the discriminator.

Interestingly, upon visually inspecting the results of the cGAN (see Figure 14), the outputs of the cGAN seem a lot more consistent with the ground truth, as many of the features, such as the horizontal gradients, are much more apparent than in the baseline outputs. This makes the metrics at first glance make even less sense, as the cGAN outputs seem like they should produce higher values than the baseline. The reason behind the inconsistency between the metrics and visuals could be that the PSNR and SSIM metrics are an objective measure of similarity between the pixels of the generated and ground truth outputs. This in turn means that any discrepancies between the images, such as shifts in position of the gradients, will cause the values to drop significantly. On the other hand, the images produced by the baseline model could score higher as the smoothing out of the gradients would normalise the extremity of the pixels, which could produce a higher score on average. This indicates that the PSNR

and SSIM values are not perfect measures of evaluation for this purpose, potentially because they are mainly used for natural images, where the groups of pixels will be more coherent compared to mel-spectrograms.

This inference is supported by the MCD values of the conditional GAN outputs (see Table 3). The MCD values show that, when converted into audio, the outputs do in fact perform better than the baseline counterparts, averaging closer to 17, which is lower than the average of 18 of the baseline models. However, despite this improvement, the values themselves are not at a very satisfactory level. Going forward, it must be noted that the PSNR and SSIM values are not an absolute indication of the model's performance, but rather a guideline of how the model improves with further training. There will therefore be more emphasis on evaluation using the MCD values and visual inspection rather than the image quality metrics.

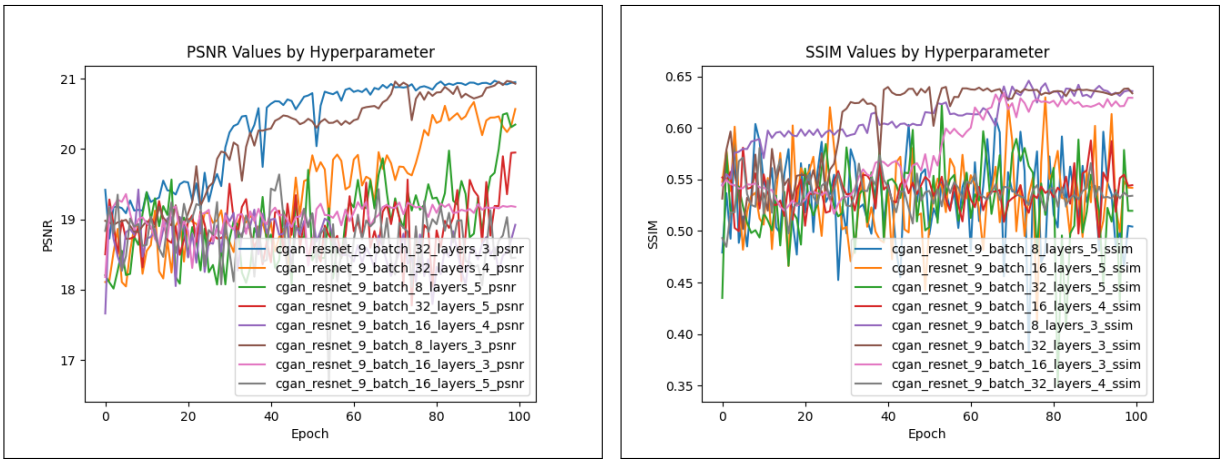


Figure 13: (a) PSNR values by Epoch for different hyperparameters of the CGAN model (b) SSIM values by Epoch for different hyperparameters of the CGAN model.

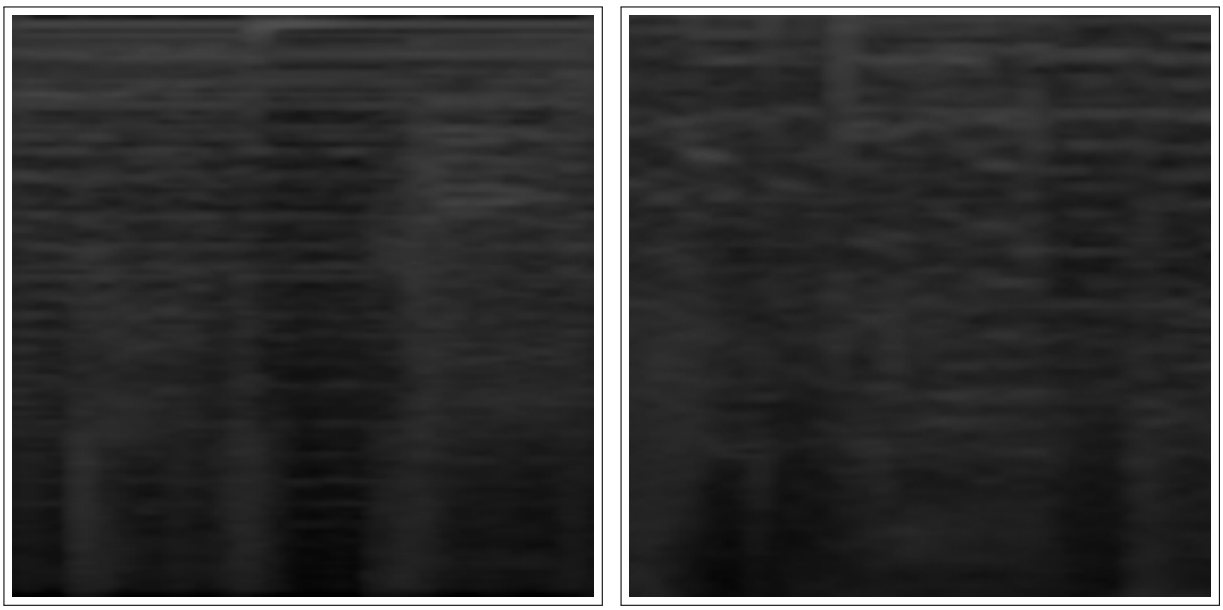


Figure 14: (a) A ground truth mel-spectrogram. (b) An output mel-spectrogram from the CGAN trained with batch size 8, 3 layers.

Model	MCD Score
Batch Size 8, 3 Layers	16.759
Batch Size 8, 4 Layers	17.341
Batch Size 8, 5 Layers	19.181
Batch Size 16, 3 Layers	18.620
Batch Size 16, 4 Layers	19.252
Batch Size 16, 5 Layers	18.272
Batch Size 32, 3 Layers	16.750
Batch Size 32, 4 Layers	16.818
Batch Size 32, 5 Layers	16.847

Table 3: Model Hyperparameters for cGAN with MCD Scores

4.2.1 Label Smoothing

A potential reason behind the sub-par performance of the model may be attributed to the fact that the discriminator model learns very quickly to differentiate between the real and fake images, leading it to be overconfident in its decisions. This is evidenced by the loss curve of the conditional GAN (see Appendix B), which shows that the discriminator loss gets close to 0 after a few epochs of training. This is not ideal, since it means that the generator is not producing high-quality images, and the discriminator is overconfident in its competition with the generator. In an ideal scenario, the training of the generator and discriminator would reach a Nash Equilibrium, where the discriminator's prediction would be no better than random chance, meaning that the generator is performing well.

One way to address this problem and improve on the current model would be to incorporate the regularisation technique of label smoothing. Label smoothing, as discussed in Chapter 3.2, addresses the issue of discriminators becoming too confident and achieving extreme loss values (0 or 1) too quickly. With label smoothing, instead of using binary (0 and 1) labels, a range of values that are slightly shifted from the extreme values are used. For example, an option would be to use 0.1 for fake data and 0.9 for real data. These values would encourage the discriminator to be less confident in its predictions, which in turn allows the generator to catch up with the discriminator's performance. The label smoothing implemented in this project used 0.9 and 0 as the labels, following the guidelines outlined in [40], which suggest that one-sided label smoothing of the positive labels is optimal.

This addition seems to improve the results quite significantly, with the improvements reflected in the MCD score of the generated audio samples. It can be seen in Table 4 that the MCD values have dropped to almost 15, which is about an 8% improvement on the cGAN, and an 11% improvement on the baseline. It can also be seen in the mel-spectrograms (Figure 15) that there is still a consistency between the generated samples and the ground truths, although it is difficult to visually recognise an improvement from the previous cGAN model without label smoothing. It is nonetheless a promising improvement, and represents by far the best results obtained so far.

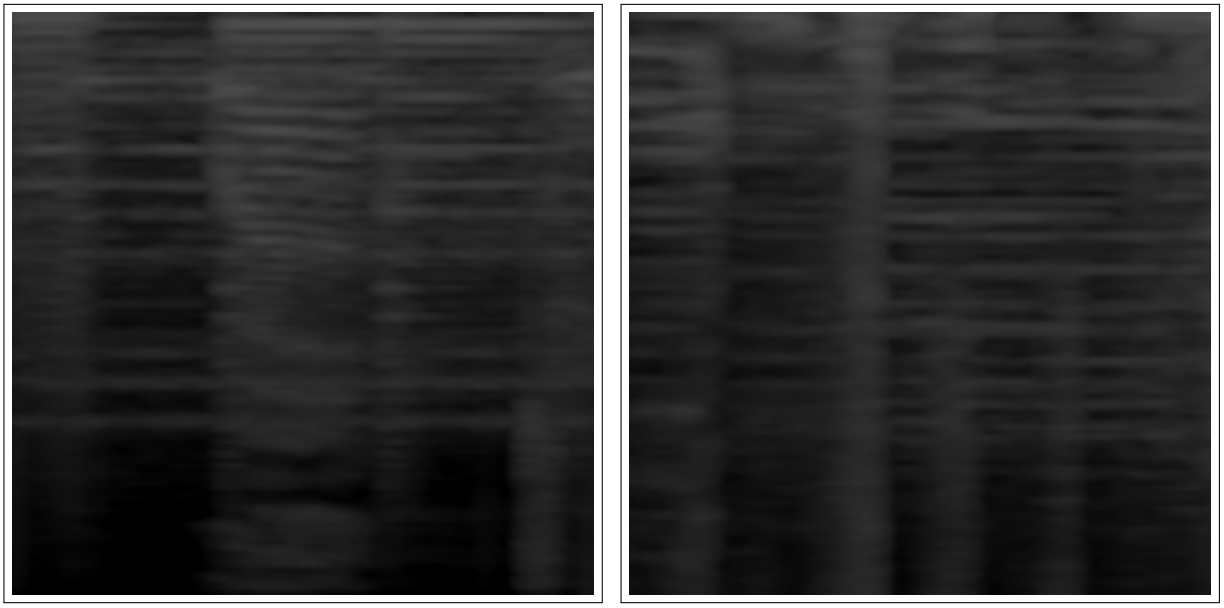


Figure 15: (a) A ground truth mel-spectrogram. (b) An output mel-spectrogram from the CGAN with label smoothing. It can be seen that the vertical 'beats' are more apparent than in the previous sample, with the horizontal gradients being more representative of the actual ground truth.

Model	MCD Score
Batch Size 8, 3 Layers	15.465
Batch Size 32, 3 Layers	15.848

Table 4: MCD Score for cGAN with Label Smoothing

Chapter 5

Discussion

In this chapter, the results as shown in Chapter 4 will be evaluated and discussed in further detail, with an emphasis on the final audio output of the model. This will be done by a combination of quantitative and qualitative analyses, which will be essential when determining the degree of success of this project, as well as the steps to follow (which will be discussed in the final chapter). The performance of the best performing cGAN model will be compared to that of the baseline to determine how much of an improvement the cGAN makes, and whether that improvement is enough.

5.1 Model Evaluation

5.1.1 Metrics

As discussed briefly in Chapter 4, the results have shown that the evaluation metrics may have some limitations when it comes to evaluating the quality of mel-spectrograms. The PSNR and SSIM metrics have both improved with training in all models, but the improvement in values did not necessarily equate to an improvement in image quality. This has more to do with the nature of the mel-spectrogram and the metrics' application in evaluating mel-spectrograms, rather than an inherent limitation of the metrics themselves, since both values are designed to evaluate natural images. It was observed that the switch from CNN to cGAN actually made the model perform slightly worse in these metrics, which was unexpected but could be explained by how the metrics are calculated.

A metric that has been consistent with the hypothesis and observations was the Mel Cepstral Distortion, which seemed to consistently improve as the supposed improvements were made to the model. It has been observed that the results from the best performing cGAN model was an improvement on the CNN baseline by around 11%, which is a significant increase. However, these values are still less than ideal, since a 'good' MCD value is still much less (<5) than the values from these results. It is therefore difficult to suggest that the model's outputs are objectively at a satisfactory level. The fact that the metrics are improving as more improvements to the model are implemented is still a promising sign, and further refinements could be the key to improving results.

5.1.2 Waveform Analysis

Another, more subjective way in which the models' outputs can be analysed and evaluated is by looking at the waveform of the final audio samples. The waveform is a representation of a signal in the form of a wave, with characteristics of the signal such as amplitude, frequency, and shape being presented in a visual manner. This allows the generated audio to be evaluated based on how similar it is to the ground truth with regards to certain attributes. Python's Librosa library [30] was used to generate these waveforms.

In the case of the baseline model, its output waveform is essentially pure noise (Figure 16). There is no consistent structure so to speak to the sound waves, and the generated audio is practically unrecognisable. This was the case for all outputs of the CNN. This was to be expected, since the mel-spectrograms produced by the baseline model lacked most of the features of a typical mel-spectrogram (see Figure 12). The smoothed out gradients is an indication of blurry sounds, which is reflected in the waveform.

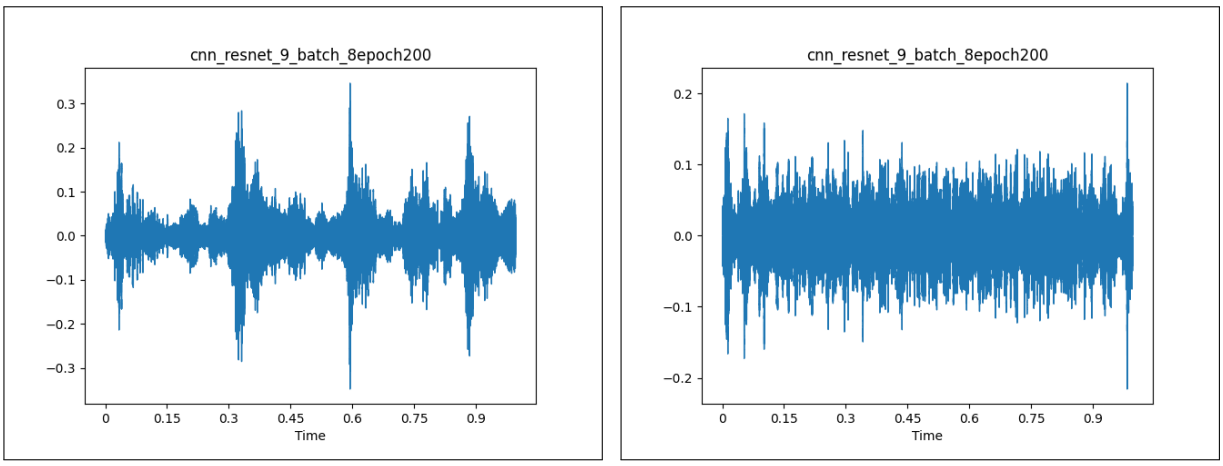


Figure 16: (a) A ground truth waveform. (b) An output waveform from the CNN.

What is more promising is the waveform produced by the cGAN-generated mel-spectrograms. What is noticeable in the generated samples is that it is no longer just gray noise, as it was the case in the baseline model. In fact, there are some striking similarities between the generated and ground truth waveforms. For instance, in the particular examples presented in Figure 17, there seems to be a peak amplitude at timestamps 0.1, 0.4, 0.65, and 0.9 in the ground truth waveform, which is reflected in the generated sample, albeit weakly. This suggests that some of the main characteristics of tempo and beat that is present in the stimulus is reflected in the brain waves, which means it can be regenerated to some degree. This finding turns out to be consistent with several neuroscience studies that have explored the effect of tempo on the brain, where EEG recordings seem to reflect temporal characteristics from external stimuli [49, 50]. These resemblances however, are not the strongest indication of the tempo being preserved in the EEG recordings, and nor are resemblances present in every reconstructed sample. More of these samples can be seen in Appendix C.

Apart from the tempo and beat, however, the regenerated sample is severely underwhelming with regards to other aspects such as frequency and timbre, both of which are significant characteristics of music. It can be seen from the waveform that the duration of certain sections with the peak amplitudes do not match, and the magnitude of the amplitude itself also does

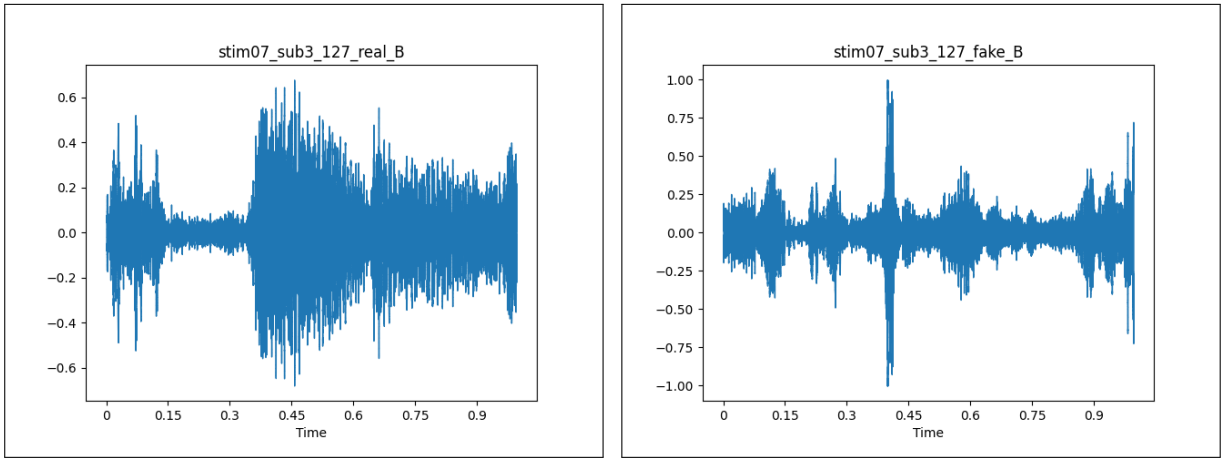


Figure 17: (a) A ground truth waveform. (b) An output waveform from the cGAN with label smoothing.

not correspond perfectly. This implies that the model is able to capture some main aspects with regards to the mel-spectrogram, but is unable to replicate the finer details.

An analysis of the waveforms suggests that there is some degree of coherence between the ground truth and generated audio samples. The correspondence of the peak amplitudes is a significant discovery, although this is not seen consistently in all the samples - around 35 of the 50 randomly generated samples seem to have this sort of correspondence, with the rest having no visible similarity. This points to the model having some potential in terms of its accuracy in regenerating mel-spectrograms, but lacking the consistency in overall performance.

5.1.3 User Study

To gain a more subjective overview of the results, a more qualitative analysis was conducted in the form of a user study. Participants would be asked to listen to 10 generated audio samples selected from the generated audio set, along with their corresponding ground truth audio samples. The generated audio set all came from the cGAN with label smoothing, and the samples were selected at random. The participants would then be asked to give a score of 1 to 5 on how similar certain characteristics of the audio sample are, which would be used to form the Mean Opinion Score (MOS) [51] on each of the characteristics. A score of 1 would mean that the generated audio demonstrates no similarity with the original audio, while a score of 5 represents a perfect resemblance.

The questions asked are as follows:

1. How are the temporal features (e.g. beat, rhythm) reflected in the generated output?
2. How are the melodic features (e.g. notes, harmonies) reflected in the generated output?
3. How is the texture (e.g. timbre) reflected in the generated output?

With scores collected from a total 9 participants, the results are shown in Table 5.

Question	Average Score
1	2.2
2	1.1
3	1

Table 5: MOS Score for Audio Samples

It can be seen that, as expected from the waveform inspection, the tempo and beat was the characteristic that was most well preserved in the reconstructed audio sample, with an average MOS of 2.2. This means that there is a consensus that at least some of the temporal and rhythmic structure of the music is retained and identifiable, although not necessarily to a large degree. On the other hand, it is quite conclusive that the melodic and textural features were almost completely not retained. It should also be noted that some audio samples had scores as high as 3 and 4 and as low as 1 for Q1, which implies that not all of the reconstructed samples retained the temporal features.

Through both objective and subjective evaluation methods, it can be concluded that the results are not necessarily satisfactory, and is less than what would have been hoped for prior to the start of the project. It has been shown that, to a degree, some temporal and rhythmic aspects of the stimulus can be captured and reconstructed from the EEG recordings using the method of image translation, while this is not the case for the melodic and textural features. However, the results do show some promise in terms of the viability of the model and overall pipeline, despite there existing inherent flaws and limitations attributed to many of the steps. In the wider context of the project, despite the results lacking conclusive evidence on the feasibility of accurate reconstruction, it does provide a glimpse of what may be possible with further revision to the model and the pipeline.

Chapter 6

Conclusion

It can be seen that, based on extant literature and the results of this project, decoding the brain using neurological data is a very challenging task. It was apparent from the beginning that this would prove to be the case, but it was regardless an extremely intriguing concept in need of exploration. Here we can revisit the main questions posed in the beginning:

1. To what extent, if any, can musical stimuli be reconstructed from EEG signals?
2. To what extent is the methodology of this project a viable approach to solving this problem?

Answering either of these questions requires a degree of caution. Based on the results of this investigation, it is unlikely for the answers to be resoundingly optimistic. At the same time, however, what was promising was that the developments made to the model throughout the project has shown to improve results, making it difficult to rule out the prospect of applying this methodology and developing it further. Although the final results were not perfect, it must not be overlooked that the purpose of the project was not to deliver perfectly regenerated audio samples, but rather to explore the feasibility of the methodology and the objective itself. To this degree, it can be said that the project was closer to a success, as it provides a conclusion from which future projects can infer. In this final chapter, the limitations and challenges of the project will be discussed, along with suggestions on areas and stages of the project that can be altered to potentially improve results in the future.

6.1 Challenges & Limitations

There were numerous points in the project that posed a challenge, both technical and methodological. One of the biggest challenges that must be mentioned from the start is that this was a project that lasted just over four months. This meant that once a methodology was defined, there was little leeway in altering the pipeline and overall methodology, which resulted in a limited exploration of all the possible variations that could be made to improve the model. Although important hyperparameters and architectures were extensively explored, this was by no means exhaustive. This issue was exacerbated by the fact that GPU resources were limited, as Imperial College London's HPC required long queue times to use the GPUs for training and testing. This was exactly the issue when attempting to build and train the Condi-

tional CycleGAN model, the proposed improvement on the Conditional GAN. The limitations in time and resources meant that an extensive trial was not feasible. These issues were partially overcome by prioritising and investing the time and resources into what parts of the pipeline that seemed more feasible and could be extended upon, such as testing the cGAN with label smoothing and creating a user study, among others.

A more inherent limitation is present within the data used in this project, as the multivariate nature of brain activity and recordings of signals make it difficult to capture relevant features that are affected by a certain stimulus. This means that proper preprocessing and cleaning of the data are essential, although these steps can differ based on the task at hand. Additionally, even if the data was properly preprocessed, it is not certain that much musical details such as melodies and timbre are actually captured in the EEG. This issue is exacerbated by the fact that there are individual differences amongst participants in any neuro-psychologial study conducted, meaning that it is difficult to be discern whether a pattern in the brain signal is actually relevant to the task at hand.

A final point is that the methodology of this project involves many lossy conversions, any of which could have had an impact on the final result. For instance, both the EEG recordings and musical stimulus was converted into their respective image representations, which, despite the best efforts to retain the maximum amount of information, involves loss of details. This is especially true with the mel-spectrogram, which in general is an appropriate visual representation of audio, but in cases such as this where we are generating sounds essentially from scratch from the neural network, it is difficult to obtain the exact frequencies and other characteristics of audio to be represented in the mel-spectrogram. The number of such conversions, along with the neural network's ability to capture details from those conversions, are bottlenecks which increase the margin of error of the results.

6.2 Future Works

There are numerous ways this project can be taken to the next step, with variations that can be implemented at any of step of the pipeline. These include how the EEG and stimulus data is represented, as well as the neural network model itself.

One of the stages of this project's pipeline where information was lost in conversion was converting the raw EEG and audio data into visual representations. This loss of information could be avoided if there were reliable ways to deal with raw signal data to train a neural network and output a high-quality audio sample. Although it is difficult to determine its exact viability, some LSTM/RNN models have been employed in the past for such purposes, but this is an area that requires more research. With the right adaptations, however, this could be a viable option to eliminate the need to use visual representations of data.

If we were to stick the GAN models and visual representations, one obvious avenue to explore is the model proposed in this project, the Conditional CycleGAN. This can be a promising model as the concept of cycle consistency could improve the accuracy of the generated mel-spectrograms. However, the limitations of these models have been very apparent in that they are designed for natural image generation tasks, which mel-spectrograms are not. The success of any future endeavour in this topic could may well depend on the development of a model capable of generating precise and reliably translatable mel-spectrograms.

These are only a few suggestions out of many more potential variations that could improve results. The main takeaway from this project is that the task of stimulus reconstruction does not seem like a far-fetched idea, but there are severe limitations that require repeated trials on various implementations and pipelines. Although there is no conclusive evidence on the degree at which the stimulus can be reconstructed, with ample time and resources, there is certainly plenty of room for improvement in the methodology that can lead to more accurate results.

Bibliography

[1] Robert J. Zatorre. Music and the Brain. *Annals of the New York Academy of Sciences*, 999(1):4–14, 2003. ISSN 1749-6632. doi: 10.1196/annals.1284.001. URL https://onlinelibrary.wiley.com/doi/abs/10.1196/annals.1284.001. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1196/annals.1284.001. pages 3

[2] Yuan-Pin Lin, Chi-Hong Wang, Tzyy-Ping Jung, Tien-Lin Wu, Shyh-Kang Jeng, Jeng-Ren Duann, and Jyh-Horng Chen. EEG-Based Emotion Recognition in Music Listening. *IEEE Transactions on Biomedical Engineering*, 57(7):1798–1806, July 2010. ISSN 1558-2531. doi: 10.1109/TBME.2010.2048568. Conference Name: IEEE Transactions on Biomedical Engineering. pages 3

[3] Dhananjay Sonawane, Krishna Prasad Miyapuram, Bharatesh Rs, and Derek J. Lomas. GuessTheMusic: Song Identification from Electroencephalography response. In *Proceedings of the 3rd ACM India Joint International Conference on Data Science & Management of Data (8th ACM IKDD CODS & 26th COMAD)*, pages 154–162, Bangalore India, January 2021. ACM. ISBN 978-1-4503-8817-7. doi: 10.1145/3430984.3431023. URL https://dl.acm.org/doi/10.1145/3430984.3431023. pages 3, 11

[4] Adolfo G. Ramirez-Aristizabal, Mohammad K. Ebrahimpour, and Christopher T. Kello. Image-based eeg classification of brain responses to song recordings, January 2022. URL http://arxiv.org/abs/2202.03265. arXiv:2202.03265 [eess]. pages 3

[5] Priyanka A. Abhang, Bharti W. Gawali, and Suresh C. Mehrotra. Chapter 1 - Introduction to Emotion, Electroencephalography, and Speech Processing. In Priyanka A. Abhang, Bharti W. Gawali, and Suresh C. Mehrotra, editors, *Introduction to EEG- and Speech-Based Emotion Recognition*, pages 1–17. Academic Press, January 2016. ISBN 978-0-12-804490-2. doi: 10.1016/B978-0-12-804490-2.00001-4. URL https://www.sciencedirect.com/science/article/pii/B9780128044902000014. pages 4

[6] NHS. Electroencephalogram (EEG), October 2017. URL https://www.nhs.uk/conditions/electroencephalogram/. Section: conditions. pages 4

[7] H. W. Agnew Jr., Wilse B. Webb, and Robert L. Williams. The First Night Effect: An Eeg Studyof Sleep. *Psychophysiology*, 2(3):263–266, 1966. ISSN 1469-8986. doi: 10.1111/j.1469-8986.1966.tb02650.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1469-8986.1966.tb02650.x. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1469-8986.1966.tb02650.x. pages 4

[8] S. S. Stevens, J. Volkmann, and E. B. Newman. A Scale for the Measurement of the Psychological Magnitude Pitch. *The Journal of the Acoustical Society of America*, 8(3): 185–190, June 2005. ISSN 0001-4966. doi: 10.1121/1.1915893. URL https://doi.org/10.1121/1.1915893. pages 5

[9] Mel scale - Wikipedia, . URL https://en.wikipedia.org/wiki/Mel_scale. pages 5

[10] Kundan Kumar, Rithesh Kumar, Thibault de Boissiere, Lucas Gestin, Wei Zhen Teoh, Jose Sotelo, Alexandre de Brebisson, Yoshua Bengio, and Aaron Courville. MelGAN: Generative Adversarial Networks for Conditional Waveform Synthesis, December 2019. URL http://arxiv.org/abs/1910.06711. arXiv:1910.06711 [cs, eess]. pages 6

[11] Curtis Hawthorne, Ian Simon, Adam Roberts, Neil Zeghidour, Josh Gardner, Ethan Manilow, and Jesse Engel. Multi-instrument Music Synthesis with Spectrogram Diffusion, December 2022. URL http://arxiv.org/abs/2206.05408. arXiv:2206.05408 [cs, eess]. pages 6

[12] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation, May 2015. URL http://arxiv.org/abs/1505.04597. arXiv:1505.04597 [cs]. pages 6, 7

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition, December 2015. URL http://arxiv.org/abs/1512.03385. arXiv:1512.03385 [cs]. pages 7

[14] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks, June 2014. URL http://arxiv.org/abs/1406.2661. arXiv:1406.2661 [cs, stat]. pages 7

[15] Overview of GAN Structure | Machine Learning, . URL https://developers.google.com/machine-learning/gan/gan_structure. pages 8

[16] Mehdi Mirza and Simon Osindero. Conditional Generative Adversarial Nets, November 2014. URL http://arxiv.org/abs/1411.1784. arXiv:1411.1784 [cs, stat]. pages 8

[17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-Image Translation with Conditional Adversarial Networks, November 2018. URL http://arxiv.org/abs/1611.07004. arXiv:1611.07004 [cs]. pages 9, 19, 20

[18] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks, August 2020. URL http://arxiv.org/abs/1703.10593. arXiv:1703.10593 [cs]. pages 10, 19, 21

[19] Isaak Kavasidis, Simone Palazzo, Concetto Spampinato, Daniela Giordano, and Mubarak Shah. *Brain2Image*: Converting Brain Signals into Images. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1809–1817, Mountain View California USA, October 2017. ACM. ISBN 978-1-4503-4906-2. doi: 10.1145/3123266.3127907. URL https://dl.acm.org/doi/10.1145/3123266.3127907. pages 12

[20] Adolfo G. Ramirez-Aristizabal and Chris Kello. EEG2Mel: Reconstructing Sound from Brain Responses to Music, July 2022. URL http://arxiv.org/abs/2207.13845. arXiv:2207.13845 [cs, eess]. pages 12, 17

[21] Ian Daly, Nicoletta Nicolaou, Duncan Williams, Faustina Hwang, Alexis Kirke, Eduardo Miranda, and Slawomir J. Nasuto. Neural and physiological data from participants listening to affective music. *Scientific Data*, 7(1):177, June 2020. ISSN 2052-4463. doi: 10.1038/s41597-020-0507-6. URL https://www.nature.com/articles/s41597-020-0507-6. Number: 1 Publisher: Nature Publishing Group. pages 14, 15

[22] Steven Losorelli, Duc T Nguyen, Jacek P Dmochowski, and Blair Kaneshiro. NMED-T: A Tempo-Focused Dataset of Cortical and Behvaioral Responses to Naturalistic Music. 2017. pages 14, 16

[23] Tuomas Eerola and Jonna K. Vuoskoski. A comparison of the discrete and dimensional models of emotion in music. *Psychology of Music*, 39(1):18–49, January 2011. ISSN 0305-7356, 1741-3087. doi: 10.1177/0305735610362821. URL http://journals.sagepub.com/doi/10.1177/0305735610362821. pages 15

[24] Arnaud Delorme. EEG is better left alone. *Scientific Reports*, 13:2372, February 2023. ISSN 2045-2322. doi: 10.1038/s41598-023-27528-0. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9911389/. pages 15

[25] Arnaud Delorme and Scott Makeig. EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis. *Journal of Neuroscience Methods*, 134(1):9–21, March 2004. ISSN 01650270. doi: 10.1016/j.jneumeth.2003.10.009. URL https://linkinghub.elsevier.com/retrieve/pii/S0165027003003479. pages 15

[26] Marius Klug and Klaus Gramann. Identifying key factors for improving ICA-based decomposition of EEG data in mobile and stationary experiments. *European Journal of Neuroscience*, 54(12):8406–8420, 2021. ISSN 1460-9568. doi: 10.1111/ejn.14992. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/ejn.14992. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/ejn.14992. pages 15

[27] Christian Kothe, Arnaud Delorme, and Makoto Miyakoshi. clean_rawdata, January 2019. URL https://github.com/sccn/clean_rawdata. original-date: 2019-01-21T00:01:49Z. pages 15

[28] P. Welch. The use of fast Fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms. *IEEE Transactions on Audio and Electroacoustics*, 15(2):70–73, June 1967. ISSN 1558-2582. doi: 10.1109/TAU.1967.1161901. Conference Name: IEEE Transactions on Audio and Electroacoustics. pages 17

[29] Alexandre Gramfort, Martin Luessi, Eric Larson, Denis Engemann, Daniel Strohmeier, Christian Brodbeck, Roman Goj, Mainak Jas, Teon Brooks, Lauri Parkkonen, and Matti Hämäläinen. MEG and EEG data analysis with MNE-Python. *Frontiers in Neuroscience*, 7, 2013. ISSN 1662-453X. URL https://www.frontiersin.org/articles/10.3389/fnins.2013.00267. pages 17

[30] Brian McFee, Colin Raffel, Dawen Liang, Daniel Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and Music Signal Analysis in Python. pages 18–24, Austin, Texas, 2015. doi: 10.25080/Majora-7b98e3ed-003. URL https://conference.scipy.org/proceedings/scipy2015/brian_mcfee.html. pages 18, 32

[31] Aditya Jain, Ramta Bansal, Avnish Kumar, and KD Singh. A comparative study of visual and auditory reaction times on the basis of gender and physical activity levels of medical first year students. *International Journal of Applied and Basic Medical Research*, 5(2): 124–127, 2015. ISSN 2229-516X. doi: 10.4103/2229-516X.157168. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4456887/. pages 19

[32] P. D. Thompson, J. G. Colebatch, P. Brown, J. C. Rothwell, B. L. Day, J. A. Obeso, and C. D. Marsden. Voluntary stimulus-sensitive jerks and jumps mimicking myoclonus or pathological startle syndromes. *Movement Disorders*, 7(3):257–262, 1992. ISSN 1531-8257. doi: 10.1002/mds.870070312. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/mds.870070312. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/mds.870070312. pages 19

[33] Yongyi Lu, Yu-Wing Tai, and Chi-Keung Tang. Attribute-Guided Face Generation Using Conditional CycleGAN, November 2018. URL http://arxiv.org/abs/1705.09966. arXiv:1705.09966 [cs, stat]. pages 20

[34] Soumya Tripathy, Juho Kannala, and Esa Rahtu. Learning image-to-image translation using paired and unpaired training samples, May 2018. URL http://arxiv.org/abs/1805.03189. arXiv:1805.03189 [cs]. pages 20

[35] Karol Kurach, Mario Lucic, Xiaohua Zhai, Marcin Michalski, and Sylvain Gelly. A Large-Scale Study on Regularization and Normalization in GANs, May 2019. URL http://arxiv.org/abs/1807.04720. arXiv:1807.04720 [cs, stat]. pages 21

[36] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification, February 2015. URL http://arxiv.org/abs/1502.01852. arXiv:1502.01852 [cs] version: 1. pages 21

[37] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, January 2017. URL http://arxiv.org/abs/1412.6980. arXiv:1412.6980 [cs]. pages 21

[38] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. ISSN 1533-7928. URL http://jmlr.org/papers/v15/srivastava14a.html. pages 21

[39] Rafael Müller, Simon Kornblith, and Geoffrey Hinton. When Does Label Smoothing Help?, June 2020. URL http://arxiv.org/abs/1906.02629. arXiv:1906.02629 [cs, stat]. pages 21

[40] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved Techniques for Training GANs, June 2016. URL http://arxiv.org/abs/1606.03498. arXiv:1606.03498 [cs]. pages 22, 29

[41] Alain Horé and Djemel Ziou. Image Quality Metrics: PSNR vs. SSIM. In *2010 20th International Conference on Pattern Recognition*, pages 2366–2369, August 2010. doi: 10.1109/ICPR.2010.579. ISSN: 1051-4651. pages 22

[42] Zhou Wang and A.C. Bovik. A universal image quality index. *IEEE Signal Processing Letters*, 9(3):81–84, March 2002. ISSN 1558-2361. doi: 10.1109/97.995823. Conference Name: IEEE Signal Processing Letters. pages 22

[43] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13 (4):600–612, April 2004. ISSN 1941-0042. doi: 10.1109/TIP.2003.819861. Conference Name: IEEE Transactions on Image Processing. pages 23

[44] R. Kubichek. Mel-cepstral distance measure for objective speech quality assessment. In *Proceedings of IEEE Pacific Rim Conference on Communications Computers and Signal Processing*, volume 1, pages 125–128 vol.1, May 1993. doi: 10.1109/PACRIM.1993.407206. pages 23

[45] Qi Chen, Mingkui Tan, Yuankai Qi, Jiaqiu Zhou, Yuanqing Li, and Qi Wu. V2C: Visual Voice Cloning. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21210–21219, New Orleans, LA, USA, June 2022. IEEE. ISBN 978-1-66546-946-3. doi: 10.1109/CVPR52688.2022.02056. URL https://ieeexplore.ieee.org/document/9878706/. pages 23

[46] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for Hyper-Parameter Optimization. In *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. URL https://papers.nips.cc/paper_files/paper/2011/hash/86e8f7ab32cfd12577bc2619bc635690-Abstract.html. pages 24

[47] James Bergstra, Daniel Yamins, and David Cox. Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures. In *Proceedings of the 30th International Conference on Machine Learning*, pages 115–123. PMLR, February 2013. URL https://proceedings.mlr.press/v28/bergstra13.html. ISSN: 1938-7228. pages 24

[48] Liam Li, Kevin Jamieson, Afshin Rostamizadeh, Ekaterina Gonina, Moritz Hardt, Benjamin Recht, and Ameet Talwalkar. A System for Massively Parallel Hyperparameter Tuning, March 2020. URL http://arxiv.org/abs/1810.05934. arXiv:1810.05934 [cs, stat]. pages 24

[49] Anna-Katharina R. Bauer, Gunter Kreutz, and Christoph S. Herrmann. Individual musical tempo preference correlates with EEG beta rhythm. *Psychophysiology*, 52(4):600–604, 2015. ISSN 1469-8986. doi: 10.1111/psyp.12375. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/psyp.12375. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/psyp.12375. pages 32

[50] Konstantinos Trochidis and Emmanuel Bigand. Investigation of the Effect of Mode and Tempo on Emotional Responses to Music Using EEG Power Asymmetry. *Journal of Psychophysiology*, 27(3):142–148, July 2013. ISSN 0269-8803. doi: 10.1027/0269-8803/a000099. URL https://econtent.hogrefe.com/doi/abs/10.1027/0269-8803/a000099. Publisher: Hogrefe Publishing. pages 32

[51] Robert C. Streijl, Stefan Winkler, and David S. Hands. Mean opinion score (MOS) revisited: methods and applications, limitations and alternatives. *Multimedia Systems*, 22 (2):213–227, March 2016. ISSN 1432-1882. doi: 10.1007/s00530-014-0446-1. URL https://doi.org/10.1007/s00530-014-0446-1. pages 33

Appendix A

Model Training Parameters

A.1 CNN

```
----------------- Options ---------------
               batch_size: 8                             [default: 1]
                    beta1: 0.5
          checkpoints_dir: ./checkpoints
           continue_train: False
                crop_size: 256
                 dataroot: ./data/nmed-t-prep            [default: None]
           dataset_mode: supervised                      [default: unaligned]
              direction: AtoB
             display_env: main
            display_freq: 400
              display_id: -1                             [default: 1]
            display_ncols: 4
             display_port: 8097
           display_server: http://localhost
          display_winsize: 256
                    epoch: latest
              epoch_count: 1
                eval_freq: 40000
              eval_metric: psnr
                 gan_mode: vanilla
                  gpu_ids: 0
                init_gain: 0.02
                init_type: normal
                 input_nc: 1                             [default: 3]
                  isTrain: True                          [default: None]
          label_smoothing: False
                 lambda_A: 10.0
                 lambda_B: 10.0
                lambda_L1: 100.0
```

```
          lambda_identity: 0.5
                load_iter: 0                                [default: 0]
                load_size: 286
                       lr: 0.0002
           lr_decay_iters: 50
                lr_policy: linear
          max_dataset_size: inf
                    model: pix2pix                          [default: ccgan]
                 n_epochs: 50
           n_epochs_decay: 50
               n_layers_D: 3
                     name: cgan_resnet_9_batch_8            [default: experiment_name]
                      ndf: 64
                     netD: basic
                     netG: resnet_9blocks
                      ngf: 64
               no_dropout: False
                  no_flip: False
                  no_html: False
                     norm: instance
              num_threads: 4
                output_nc: 1                                [default: 3]
                    phase: train
                pool_size: 0
               preprocess: resize_and_crop
               print_freq: 100
             save_by_iter: False
          save_epoch_freq: 5
         save_latest_freq: 5000
           serial_batches: False
                   suffix:
              super_epoch: 50
        super_epoch_start: 0
              super_start: 1
         update_html_freq: 1000
                use_wandb: False
                  verbose: False
        wandb_project_name: CycleGAN-and-pix2pix
----------------- End -------------------
```

A.2 cGAN

```
---------------- Options ---------------
               batch_size: 8                                [default: 1]
                    beta1: 0.5
          checkpoints_dir: ./checkpoints
```

```
  continue_train: False
       crop_size: 256
        dataroot: ./data/nmed-t-prep            [default: None]
    dataset_mode: supervised                    [default: unaligned]
       direction: AtoB
    display_env: main
    display_freq: 400
      display_id: -1                            [default: 1]
   display_ncols: 4
    display_port: 8097
  display_server: http://localhost
 display_winsize: 256
           epoch: latest
     epoch_count: 1
       eval_freq: 40000
     eval_metric: psnr
        gan_mode: vanilla
         gpu_ids: 0
       init_gain: 0.02
       init_type: normal
        input_nc: 1                             [default: 3]
         isTrain: True                          [default: None]
 label_smoothing: False
        lambda_A: 10.0
        lambda_B: 10.0
       lambda_L1: 100.0
 lambda_identity: 0.5
       load_iter: 0                             [default: 0]
       load_size: 286
              lr: 0.0002
  lr_decay_iters: 50
       lr_policy: linear
max_dataset_size: inf
           model: pix2pix                       [default: ccgan]
        n_epochs: 50
  n_epochs_decay: 50
      n_layers_D: 3
            name: cgan_resnet_9_batch_8         [default: experiment_name]
             ndf: 64
            netD: basic
            netG: resnet_9blocks
             ngf: 64
      no_dropout: False
         no_flip: False
         no_html: False
            norm: instance
     num_threads: 4
```

```
               output_nc: 1                                [default: 3]
                   phase: train
                pool_size: 0
              preprocess: resize_and_crop
              print_freq: 100
            save_by_iter: False
          save_epoch_freq: 5
         save_latest_freq: 5000
           serial_batches: False
                  suffix:
             super_epoch: 50
        super_epoch_start: 0
              super_start: 1
         update_html_freq: 1000
                use_wandb: False
                  verbose: False
       wandb_project_name: CycleGAN-and-pix2pix
----------------- End -------------------
```

A.3 cGAN with Label Smoothing

```
              batch_size: 8                               [default: 1]
                   beta1: 0.5
            checkpoints_dir: ./checkpoints
             continue_train: False
                  crop_size: 256
                   dataroot: ./data/nmed-t-prep            [default: None]
             dataset_mode: supervised                      [default: unaligned]
                direction: AtoB
              display_env: main
             display_freq: 400
               display_id: -1                              [default: 1]
            display_ncols: 4
             display_port: 8097
           display_server: http://localhost
          display_winsize: 256
                    epoch: latest
              epoch_count: 1
                eval_freq: 40000
              eval_metric: psnr
                 gan_mode: vanilla
                  gpu_ids: 0
                init_gain: 0.02
                init_type: normal
                 input_nc: 1                                [default: 3]
```

```
             isTrain: True                                [default: None]
      label_smoothing: True                               [default: False]
             lambda_A: 10.0
             lambda_B: 10.0
            lambda_L1: 100.0
      lambda_identity: 0.5
            load_iter: 0                                  [default: 0]
            load_size: 286
                   lr: 0.0002
       lr_decay_iters: 50
            lr_policy: linear
     max_dataset_size: inf
                model: pix2pix                            [default: ccgan]
             n_epochs: 50
       n_epochs_decay: 50
           n_layers_D: 3
                 name: cgan_label_smoothing               [default: experiment_name]
                  ndf: 64
                 netD: basic
                 netG: resnet_9blocks
                  ngf: 64
           no_dropout: False
              no_flip: False
              no_html: False
                 norm: instance
          num_threads: 4
            output_nc: 1                                  [default: 3]
                phase: train
            pool_size: 0
           preprocess: resize_and_crop
           print_freq: 100
         save_by_iter: False
      save_epoch_freq: 5
     save_latest_freq: 5000
       serial_batches: False
                suffix:
          super_epoch: 50
    super_epoch_start: 0
         super_start: 1
     update_html_freq: 1000
           use_wandb: False
              verbose: False
wandb_project_name: CycleGAN-and-pix2pix
```

Appendix B

Loss Curves

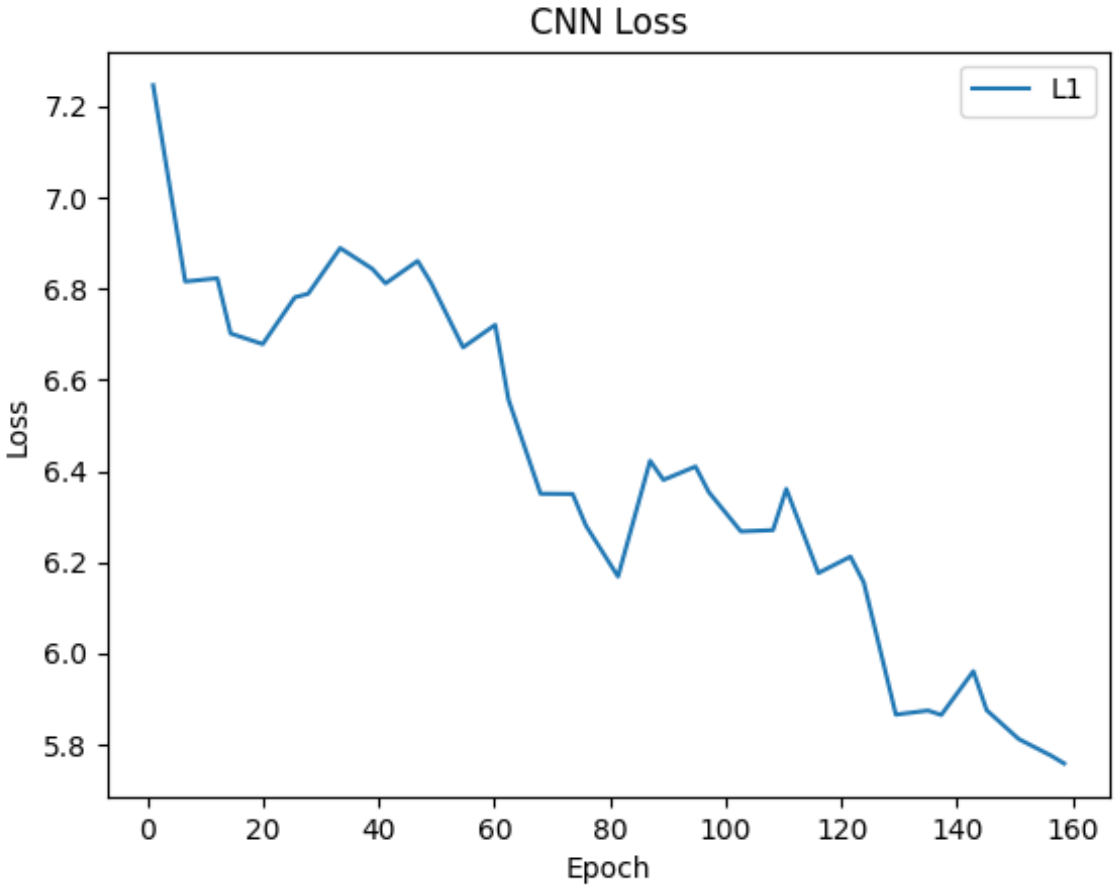


Figure 18: Loss Curve of CNN

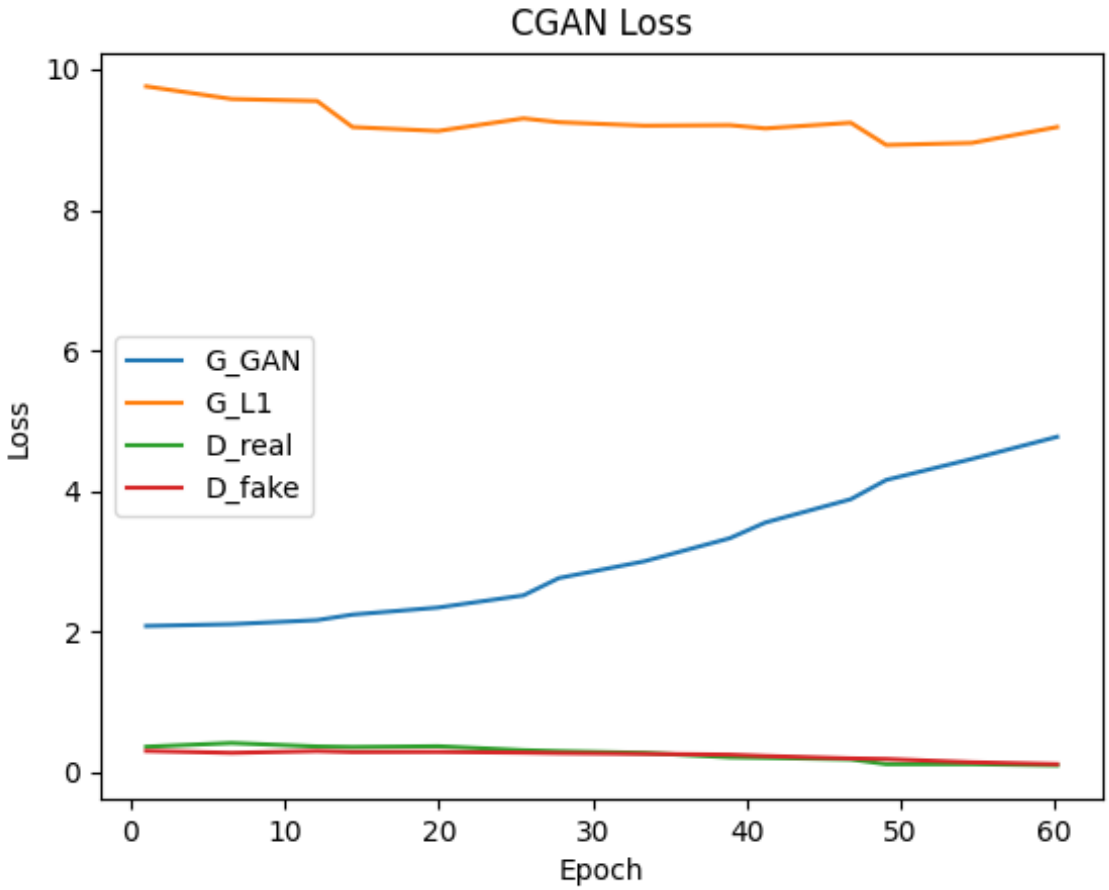


Figure 19: Loss Curve of cGAN

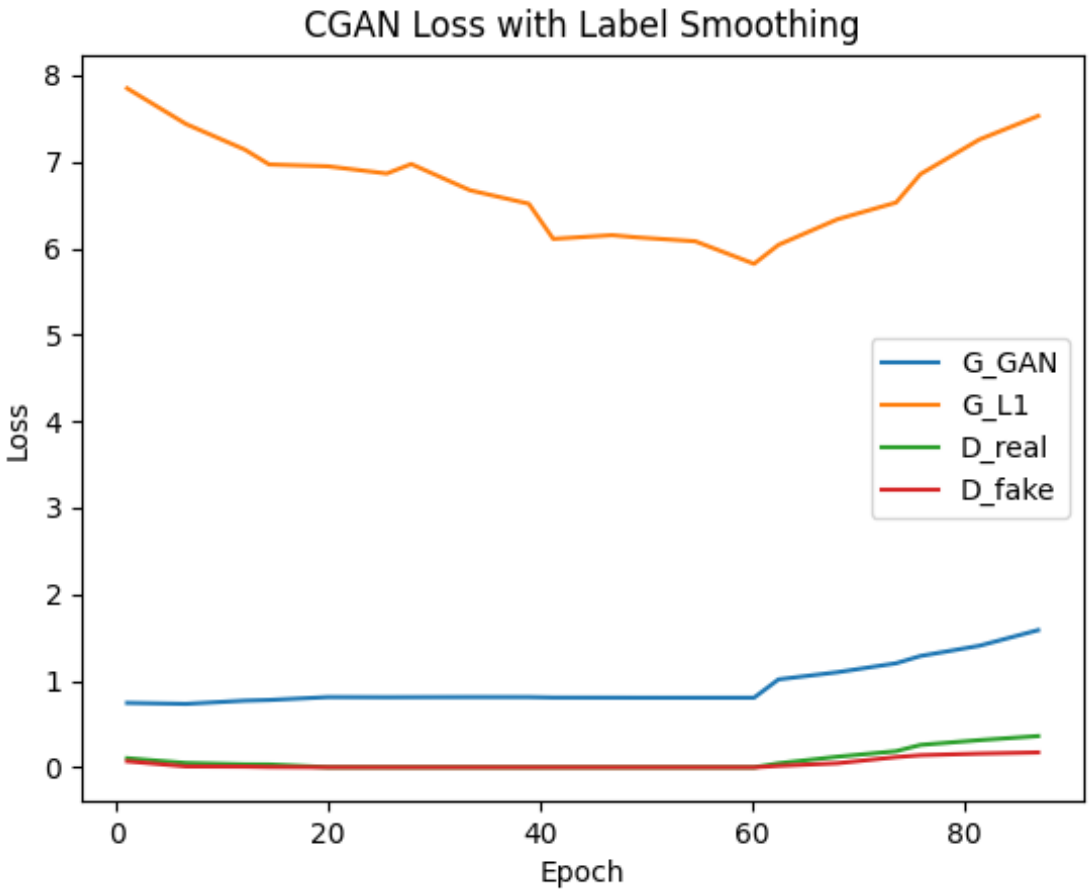
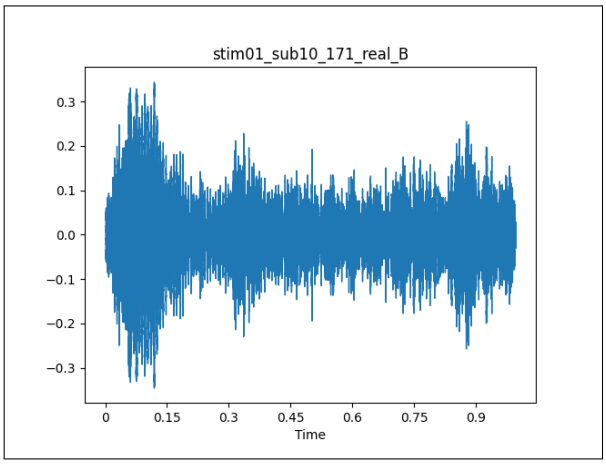
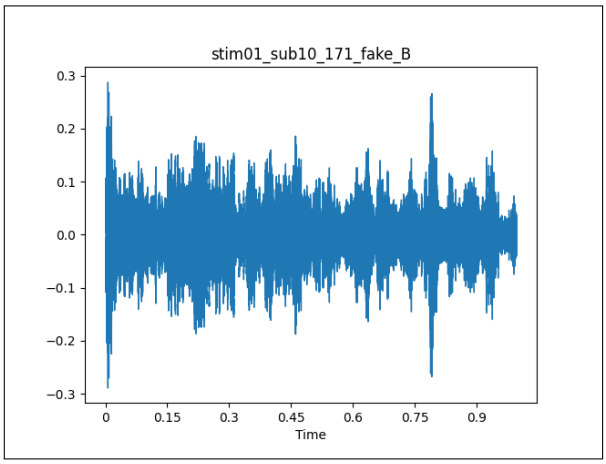


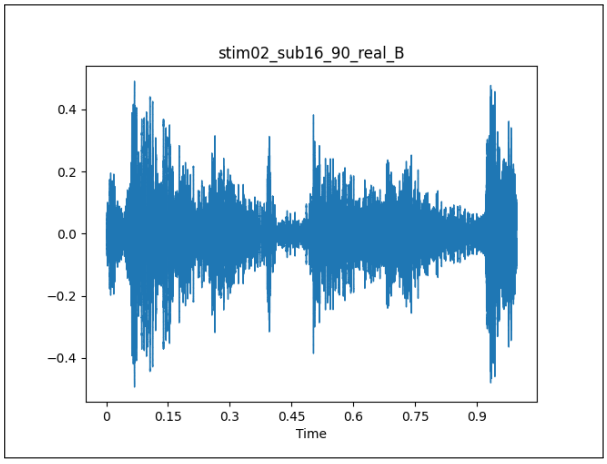
Figure 20: Loss Curve of cGAN with Label Smoothing

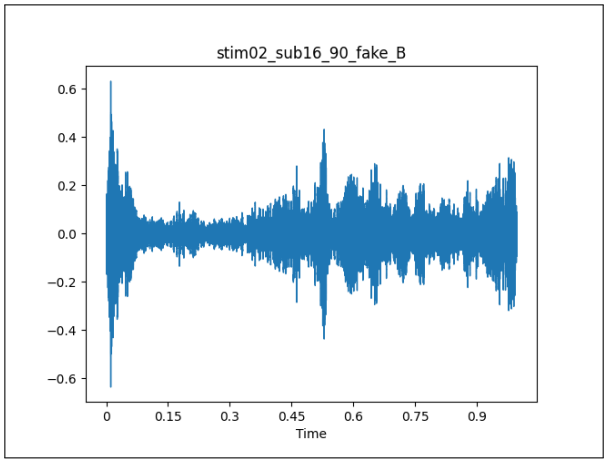
Appendix C

Waveform Comparisons









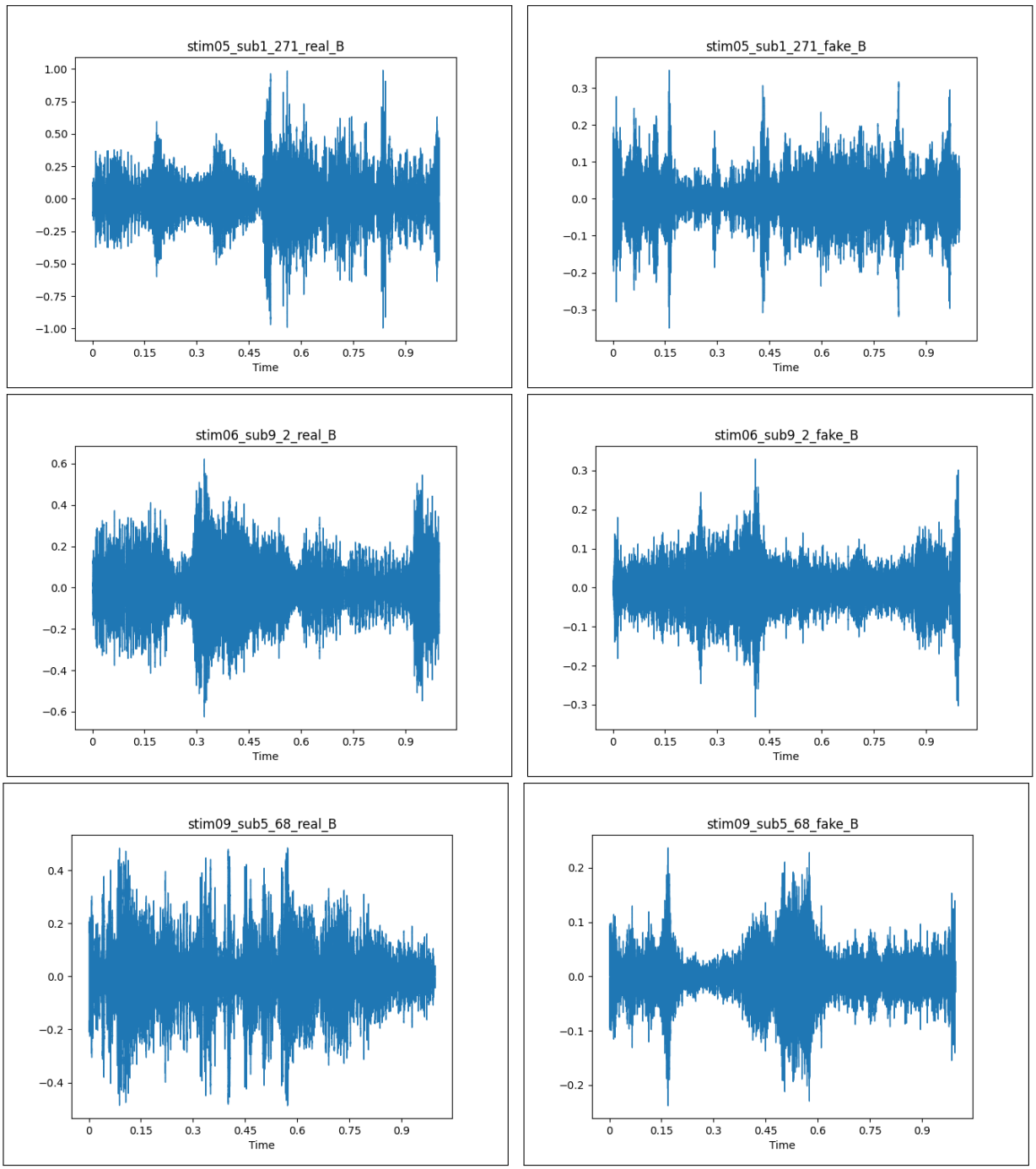


Figure 21: Waveform samples of real vs generated audio