# From Brainwaves to Images: EEG Data and GAN-Powered Image Generation

UNIVERSITY OF
LINCOLN

Tim Tanner
27207749

27207749@students.lincoln.ac.uk
School of Computer Science
College of Science
University of Lincoln

Submitted in partial fulfilment of the requirements for the
Degree of MSc Intelligent Vision

*Supervisor:* Dr. Vassilis Cutsuridis

September 2024

# Acknowledgements

Firstly, I want to thank Andrea for her patience and compassion whilst I completed this work. I would also like to thank Dr Tessa Flack & Dr Matt Craddock for their time and valuable insight in the subject of EEG signals.

# Abstract

This study explores the integration of Electroencephalography (EEG) data and Generative Adversarial Networks (GANs) to generate visual representations of perceived images. Utilizing the Mind Big Data (MBD) dataset and Emotiv EPOC® device, EEG signals were captured and pre-processed to remove noise and artifacts. A Convolutional Neural Network (CNN) was employed to classify and encode these signals into latent space vectors. These vectors, along with class conditioning labels, were fed into an Auxiliary Classifier GAN (ACGAN) to generate images. The study faced challenges such as signal complexity, noise, and limited training data. Despite these, the ACGAN model, particularly with a modulation layer and concatenation with embedding, demonstrated the ability to produce images with a Structural Similarity Index (SSIM) score of approximately 0.3, indicating a 65% similarity to the ground truth. The results validate the hypothesis that EEG signals can be used to generate representative images, although further improvements in data preprocessing and classification accuracy are needed. Future work could explore reinforcement learning to enhance the model's performance.

# Table of Contents

# List of Figures

# List of Tables

**Chapter 1**

# Introduction

There has been a growing interest in the interface between the brain and computers, the brain computer interface (BCI). Alongside this perceptual brain decoding (PBD) has developed as a contemporary research trend. “Over the past 15 years there has been an intensification in interest and research in ways of applying AI to help the BCI in various applications such as cursor control, neuroprosthetics and patients with paralysis” (Zhang et al. 2020). In the following work I intend to introduce the use of main components, Electroencephalography (EEG) data and generative adversarial networks (GANs) and by combining these techniques collaboratively I hope to show how sufficient feature data can be extracted, classified and used to recreate the perceived representative image being viewed by the subject. The study involves a combination of signal processing, artificial intelligence, deep neural network, and machine learning techniques along with image processing to achieve the desired outcomes.

## 1.1 Electroencephalography, background, and potential for feature extraction

Electroencephalography (EEG) signal capture has become more common with commercially available cost-effective solutions becoming more widespread. This technology is a non-invasive method used to record electrical activity in the brain. This activity is primarily the synchronized activity of large groups of neurons in the cerebral cortex. Electrodes placed on the head detect the small electrical potentials generated by brain activity. In this study the data has been collected using the Emotiv EPOC® device which has been validated (Badcock et al. 2013) and used in similar studies and projects and used a subset of 10-20 electrode positioning system. Fig 1 shows the electrodes and positions which follow with

the standard 10-20 system which is the standardised method for placing EEG electrodes on the scalp (Jasper, H. 1957)(Homan et al. 1987).

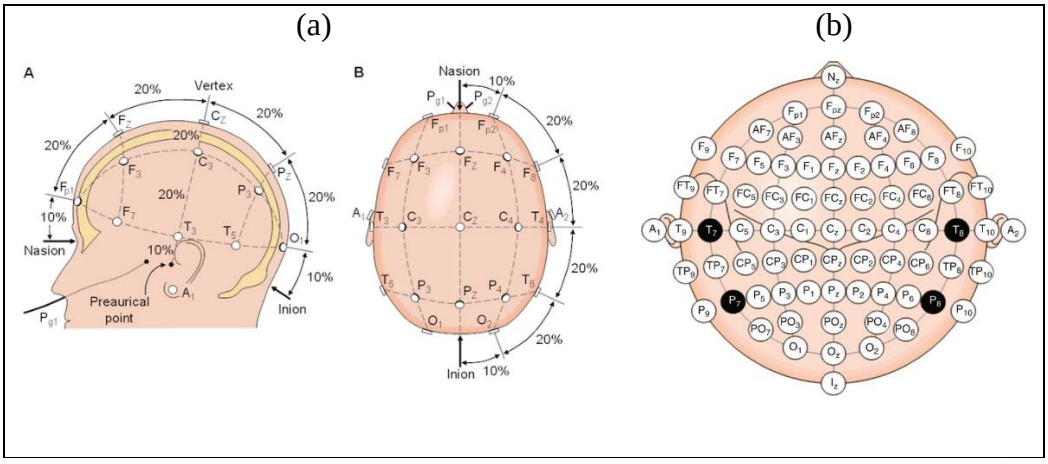


*Figure 1: 10-20 standard electrode position (a) spatial schematic layout (b) electrode labelling. Images from (Homan et al. 1987)*

The distribution of electrodes around the skull allows for the capture of results with a degree of spatial as well as temporal relativity. The resulting signals are amplified and recorded as time-series waveforms, which fluctuate in response to brain activity over time. Such data have been used to study sleep patterns, seizures, and neurological disorders (Prerau, M et al. 2017). In this study I propose to build on some of the signal processing techniques (Chaddad et al. 2023) to extract features from the data which can then be used to generate a visualization corresponding to the image being viewed as a stimulus at the time of data recording. The significance of using EEG is its ability to provide a window on the brain functionality. Being non-invasive and with low-cost commercial equipment it is easier to access than other neuroimaging techniques such as functional magnetic resonance imaging (fMRI). EEG has been instrumental in advancing our understanding of brain function and has significant applications in both clinical and research settings.

## 1.2 Generative Adversarial Networks, background, and principles

Generative adversarial networks (GANs) since their introduction by (Goodfellow et al. 2014) have become a powerful and popular technique in the field of AI, particularly for generating new, or synthetic data that resembles real-world data (Russakovsky et al. 2015). Their features include a two-network structure comprising of a generator G and Discriminator D. The generator is trained to create synthetic image data from a latent space vector.

$$I = G(z) \qquad (1)$$

$$I = G(z_{eeg}) \qquad (2)$$

The latent space vector z in eq.1 is typically a sample distribution (.i.e. gaussian or uniform), in this study I propose to use an encoded vector representation of the subject EEG data eq.2. The discriminator network is trained adversarial against the generator and tries to distinguish between real and synthetic data. These two networks are trained simultaneously in a competitive process which leads to adversarial nature of training which is described in the name. As the training progresses the generator produces increasingly better image data in attempting to fool the discriminator, and the discriminator becomes better at identifying fake data. The final output of the trained basic GAN is a confidence in the generated image with lower confidence being attributed to fake or non-conforming distribution data in the latent space vector, higher confidence indicating the image generated is more representative.

$$p_{eeg}\left(G\left(z_{eeg}\right)\right) = 0.5 \qquad (3) \qquad \text{(Goodfellow et al. 2014)}$$

Their uses include image generation, text-to-image conversion, and data augmentation, among others. Advantages of GAN's include the ability to mimic complex data distributions and generation of high realistic synthetic data.

The Auxiliary Classifier Generative Adversarial Network (ACGAN) (Odena et al. 2017) is an extension of the traditional GAN that incorporates an auxiliary classifier to improve the quality and control of generated images. In this case the discriminator D expresses the results as a distribution over both source and class.

$$Ls = E[\log P(S = real\ |X_{real})] + E[\log P(S = fake\ |X_{fake})] \quad (4)$$

$$Lc = E[\log P(C = c\ |X_{real})] + E[\log P(C = c\ |X_{fake})] \quad (5)$$

Ls, Lc are log likelihood of correct source and class, D maximizes Ls + Lc and G minimizes Lc - Ls

GAN's do come with challenges of difficulty to train and achieve stability in results. However, GAN's have significantly advanced the field of generative AI and continue to be an active area of research and development. Indeed, they have achieved some remarkably successes in image generation notable.

- Creating images of animals, vehicles, and other objects.
- Filling in missing or damaged parts of images
- Generating additional training data for other machine learning models

They have enabled improved image processing techniques and provided valuable tools for data augmentation in AI development.

## 1.3 Difficulties with EEG and GAN's

Current EEG-to-image generation techniques encounter several significant challenges. I have identified three which will need special attention during the project:

- Signal Complexity and Variability: EEG signals exhibit high complexity and variability, differing between individuals and even within the same individual

over time. This variability complicates the task of developing consistent and reliable mappings between EEG patterns and visual imagery.

- Noise and Artifacts: EEG recordings are prone to various sources of noise and artifacts, such as muscle movements, eye blinks, and electrical interference. These disturbances can degrade the quality of the generated images.
- Limited Training Data: There is a scarcity of high-quality datasets that pair EEG recordings with corresponding visual stimuli or mental imagery. This shortage hinders the ability to train robust machine learning models effectively.

In this study the focus is on decoding the signals recorded indirectly from the brain. Reconstructing these decoded features is a demanding task as the stimuli are potentially infinite along with the influence from artifacts such as eye blinks or movement and facial muscle movements. In fact, there could be feature data which is decoded that is predominantly influenced by these gesture and muscular signals as opposed to the more subtle signal data underlying the EEG data.

As mentioned previously the standard 10-20 system for EEG electrode positions have been developed to give a greater depth to spatial resolution in the signal data gathered. The numbers refer to either 10% or 20% of the total distance (front-back or left-right) of the skull when refereeing to the distances between adjacent electrodes. The limited availability of quality-controlled EEG data sets means that we rely on publicly available datasets such at that used in this study Mind Big Data 2022 (MBD) (Vivancos and Cuesta 2022). However, the Emotive EPOC device uses a subset of the total electrode position, see Fig 2, and therefore the spatial resolution is substantially less than that of the full system. I show how the lack of electrode positions across the top of the skull could be a factor in difficulties handling and processing the data. The other issues related to EEG data is the lack of reference signal and detail in the data collection procedure. Typically, the electrodes do need some diligence when they are applied, i.e. if the subject is perspiring the sweat can affect the signal conductivity. To accurately mark

artefacts such as eye movements or muscular movements reference electrodes are position in addition to those on the scalp in positions such as under the eye. There are no such reference electrodes in the MBD data.

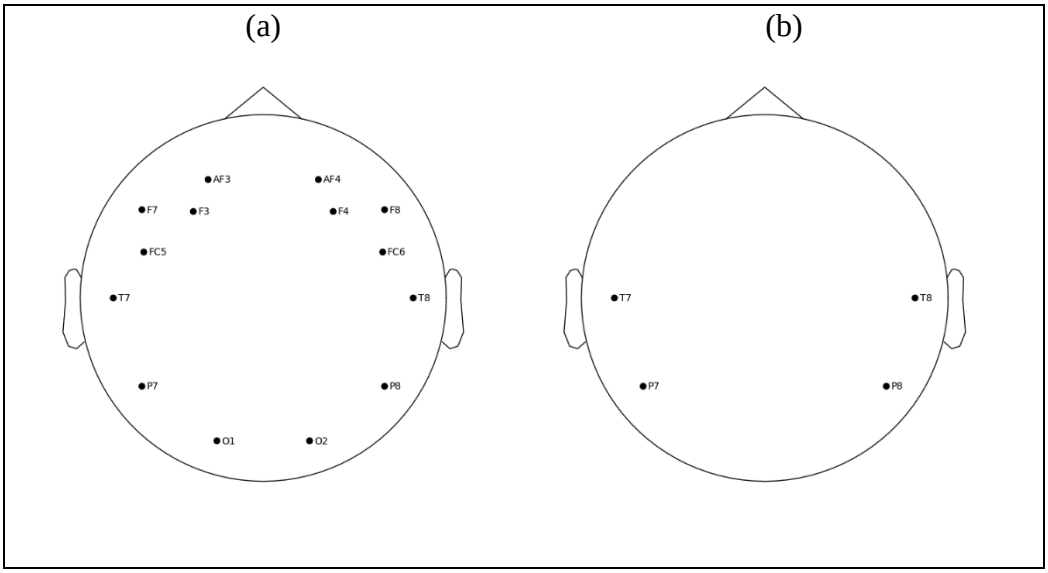


*Figure 2: (a) EEG electrode positions for Emotiv EPOC device 14 channels. (b) 4 channels used for signal data input to the classification/encoder network.*

In this study I will attempt to overcome some of these obstacles such as using data from a single subject to minimise variation across subjects, preprocessing the EEG data to remove or mitigate for noise and artefacts and using an extensive data set which has been made publicly available. Despite these limitations, ongoing research in the field is steadily advancing, continually pushing the boundaries of what is achievable in in the field of PBD and BCI.

**Chapter 2**

# Literature Review

The interest in decoding brain activity and using the results to interact with and provide feedback for the physical and clinical sectors has been growing over the past decade. The interest in BCI and its use in communication is discussed in (Kumar et al. 2018) for decoding speech from EEG signals. The data set is a small sample of some 23 subjects and the sample period is 10 seconds. Which is quite different to the MBD (Vivancos and Cuesta, 2022) dataset used in this study. In the MBD dataset there is only one subject used for collection of all data points, the stimulus period is 2 seconds, and only mono stylised representations of the numeral digits [0 ... 9] are used as stimulus. These are then interpreted as the MNIST (LeCun et al. 1998) dataset of digits are used as the stimulus images. The data set from (Kumar et al. 2018) seems to form the basis for a starting point for other studies as I will expand on below. The key difference about this data set compared with MBD is that the subject is asked to imagine the concepts of objects as opposed to the stimulus being a visual one. Whilst the subject is shown a visual representation of the object, the stimulus data which is recorded in the EEG data is that from the period when the subject is asked to imagine the concept. This study collects data using image of objects, letters and digits and attempts to apply feature extraction calculating several key statistical values derived from the EEG signals such as mean, sum value, energy, standard deviation and root mean squared error. These features are packed into a vector and then applied to a two-stage random forest classifier for classification whilst they do not attempt to recreate the envisioned image the classification accuracy is reported as a reasonably ~80/90%. Taking the MBD dataset and selecting a sub sample of 400 signals per class I investigated the various features which may be suitable to be extracted from this data set so a similar classification could be applied. Using the highly comparative time-series analysis framework (Fulcher et al 2013) the sample data was evaluated

for suitability. The study shows how this method was not suitable in the classification of the MBD data as input into the generation of images for the ACGAN, chapter 5 gives more detail. Fig 3 below shows the relative distributions between classes in a low dimensional space.

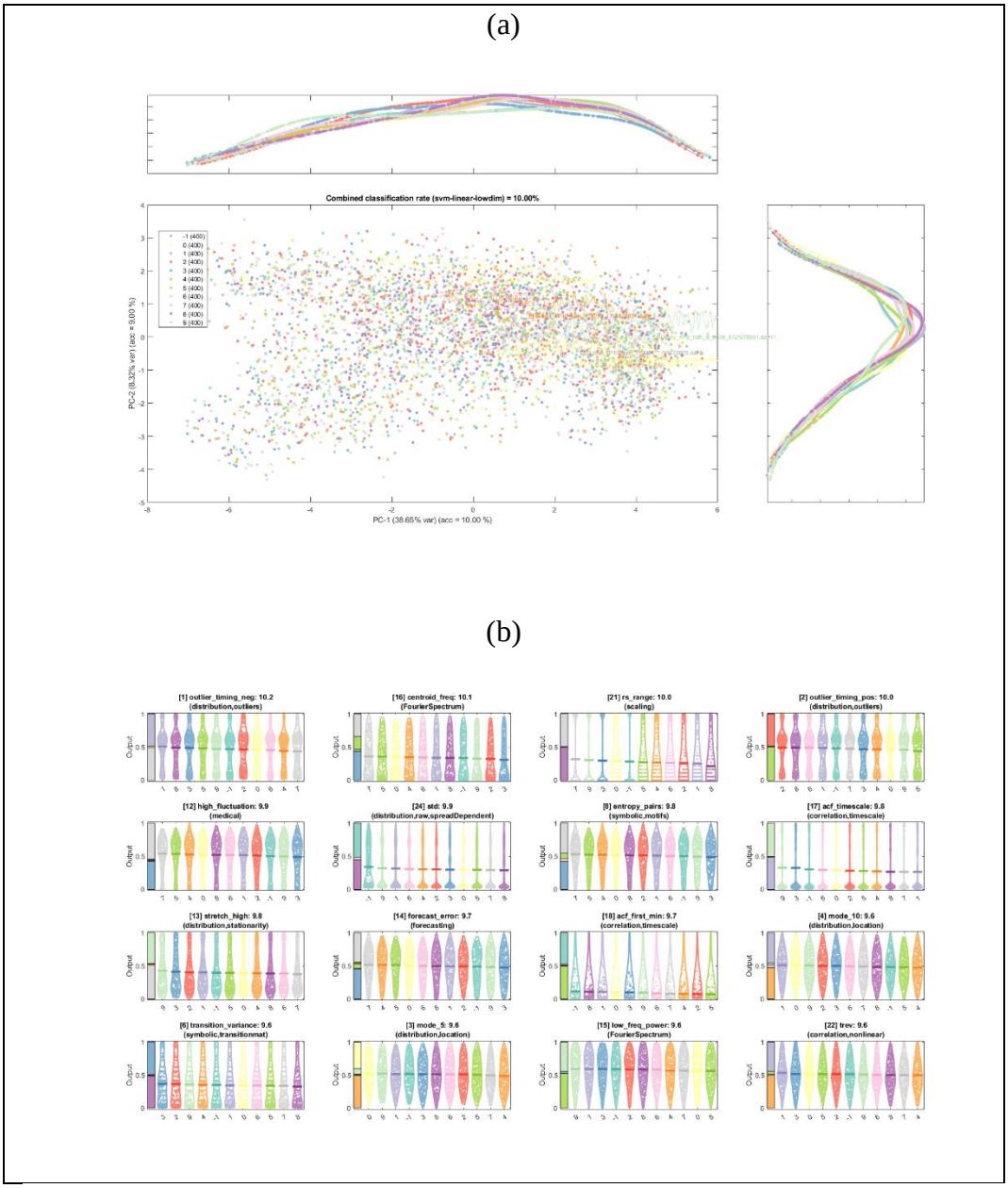


*Figure 3:* (a) PCA low dinesional space distribution using statistical and deterministical features, based on 1000 samples per class label using HCTSA "Catch24", (b) Top 16 feature measures base don 1000 samples per class, Note: no statistical difference in feature variation for class sepeartion.

## The progression of GAN models using fMRI/EEG data

Whilst (Koide-Majima et al. 2024) uses an fMRI data set as opposed to EEG data, they propose a decoding method of the signals using deep neural networks combined with a pretrained image generator based on a GAN architecture. The evaluation of the decoded activations from the fMRI signals is processed using a Bayesian estimator to create a latent space vector which after combination with a gaussian noise vector is used by the image generator. The image reconstruction and identification are reasonably with accuracy values ~ 70/75% they have shown that it is possible to generate the recreated the envision image. The authors have built on the previous work by (Shen et al. 2019) and they have used brain activity from both visual stimulus and imagined activity. In both studies the stimulus was controlled using time locked signals and recording where over 8 seconds for each stimulus event. With a control group of subjects limited to 3 individuals.

The use of GANs and their variety of designs has grown since their inception the authors (Aggarwal et al. 2021) provide a comprehensive list of several application and model designs. Particularly in image processing the deep neural network-based GAN proposed by (Zhang et al. 2020). Deep convolutional GAN architecture DCGAN is a type of GAN network which has been used with EEG data. The DCGAN was first proposed (Radford et al. 2016) as an improvement of the original GAN design and further improved in (Salimans et al. 2016). The key factors of the DCGAN design were as follows.

- The addition of batch normalization in both the generator and discriminator networks.
- The final layer of generator uses tanh activation to clip the convolution activations between [-1 , 1].
- The use of LeakyRelu with an activation slope of 0.2 for the discriminator.

The DCGAN design is used in the paper (Seeliger et al. 2018) this paper also uses fMRI data from subjects viewing both digit data and object data. In this paper we

start to see the use of encoded data from the fMRI as the input latent space vector used as input to the pretrained generator network of the GAN. These papers are really focusing on the image generation and do not give full details on the signal processing or the relative feature data which is encoded in the fMRI data. The data sets used are not public and there is no way to validate the results. At this time DeLiGAN (Gurumurthy et al. 2017) introduced a method of parameterizing the input noise latent space vector used by the generator of GAN network architecture. This method involves adding custom trainable weights which are used to modulate the generative latent space. The paper also introduces the use of the inception score as a means of measuring the assessment of image generation. The Inception Score (IS) is a metric used to evaluate the quality of images generated by generative models, such as GANs, without direct comparison to ground truth images. Using the inception score to evaluate generated MNIST images against the original MNIST dataset is generally not suitable. The Inception Score relies on a pre-trained Inception network, which is trained on ImageNet, a dataset containing diverse and complex images from numerous classes. This model is not optimized for the simpler, grayscale digit images found in MNIST, which can lead to inaccurate or meaningless evaluations. The model uses a DCGAN convolution architecture, and by introducing the learnable weights to the latent space they can show improved image generation above that of the base GAN model. This study does not entail the use of EEG data it is focused on the improvement of the generation of realistic images in the case of limited data from diverse sources. But the key feature of the trainable modulation layer is a concept that could be used in augmenting the latent space vector from the EEG data.

The introduction of an additional classifier network further improves the GAN/DCGAN design by introducing an auxiliary conditioning parameter (Odena et al. 2017). This conditional parameter introduces the prediction of the class label and well as a predicted validity probability distribution. The class-conditional image synthesis model was able to demonstrate that class information can be encoded in the image representation. The paper also uses the concept of the

inception score to assess the discriminability of the image generation. The key concept taken from this paper is the auxiliary classifier which is used in this study to create the class label guiding code and to encode the EEG latent space.

In the paper ThoughViz (Tirupattur et al. 2018) the authors take a pre-processed set of the data from (Kumar et al. 2018) which is an EEG data set based on imagined concepts. They focus on the process of decoding thoughts generated in the brain as opposed to decoding the signals from a visual stimulus. The study attempts to synthesize visual representations from three different datasets of objects, characters, and letters. They use a modified ACGAN structure with the addition of re-parameterization of the latent space introduced in (Gurumurthy et al. 2017), in addition they introduce the preprocessing of the EEG signals by sampling the channel signal into windowed sections including an overlap this technique enables a greater localization of features for classification and was introduce in (Kumar et al.2018). The data samples where 10 seconds at 128Hz sample frequency and split into 32-sample periods with an 8-sample overlap. This equates to ~ 32/128 = 250m second sample length. Similarly, and more recently NeuroGAN (Mishra et al. 2023) taking the same data set as earlier studies applied a modified conditional GAN model to the task of decoding the thoughts stimulus data. This paper took an alternative approach to providing the conditional parameter by using an attention-based auto-encoder to both generate the image but also provide the latent space vector from the encoder decoder bottleneck interface. The attention block took the raw signals and provided embedding to the encoder network which outputs the EEG signal encodings then using the re-parametrizing function as in previous papers as the input the image generating decoder. Finally, the generated image was then passed to the discriminator and the classifier. The key difference here is the classification and discrimination are done in the final block in separate model blocks. The key feature taken from this study is the use of embeddings and this study uses the embeddings techniques to encode the class label in the input layers of the generator.

In this study the MDB data is much shorter in duration and the stimulus recorded is of the subject viewing the image, each epoch is 2 seconds in length at 128Hz sample frequency versus 5 or 10 seconds from the previous data sets used. Whilst the amount of data points is greatly increased in the MBD data from the that of (Kumar et al. 2018), there is a lot more noise and artifacts which must be pre-processed. It has presented a critical difference in the method applied to achieving classification performance required to provide the conditioning parameters for use with the ACGAN model structure. The authors of (Mishra et al. 2021) highlight methods using convolutional neural networks to provide classification, but importantly they discuss the importance of the pre-processing of the EEG data required especially with short duration signals.

## Preprocessing of EEG data for use in machine learning

Pre-processing EEG data is a critical step in ensuring that the signals obtained are as close to the "true" neural signals as possible. This process involves various techniques to remove noise and artifacts, which can obscure the weaker EEG signals that are of interest. In the MDB dataset the author expresses that none of the signal data has previously been pre-processed. In examining the methods and benefits of pre-processing EEG data, certain methods become a recurring theme.

- The need to attenuate the noise to signal ratio, remove potential DC and AC components and electrode external noise.
- The need to remove epochs which have significant artifacts from eye movement of muscular effects.

The authors (Radüntz, 2018) give a comprehensive summary of signal quality evaluation for various EEG devices and includes the Emotiv EPOCH device which is used by MDB to collect the dataset used in this study. The article shows that with careful treatment the EEG signal data collected from these devices can be effectively pre-processed for further use in the realm of machine learning as required for this study. The benefits of pre-processing such as noise reduction,

artifact removal, improved signal quality and standardization and are essential for several reasons. The authors (Chaddad et al. 2023) discuss many aspects and review the importance of the pre-processing EEG signal pipeline.

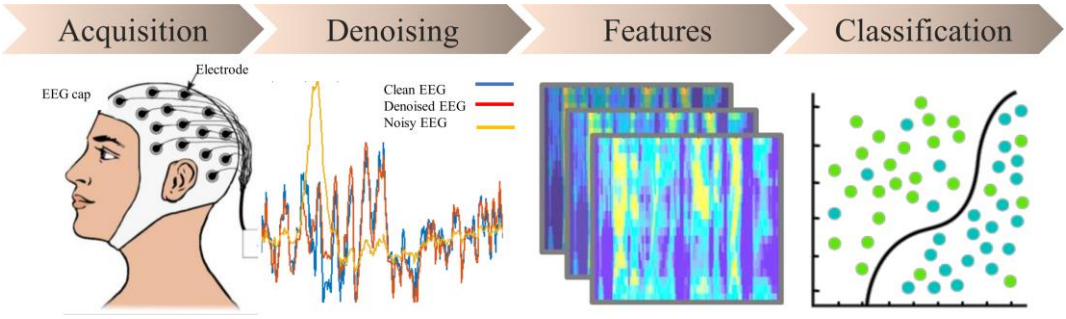


*Figure 4:* Example 4 Step processing pipeline typically used with EEG signal processing, Image from (Chaddad et al. 2023).

They discuss the use of bandpass filtering between the ranges of 0.4 to 60Hz to reduce noise but still preserve neural oscillation bands (Delta, Theta, Alpha, Beta, Gamma) all of which contribute to the visual processing task, but in particular Gamma (30-60 Hz) and Beta (13-30 Hz) which have been shown to have significance in the visual processing tasks. The paper outlining powerline effects and ocular denoising (Tibdewal et al. 2016) show how using methods such as notch filter and wavelet transforms can effectively remove unwanted noise and artifacts. Artifacts often overpower the more delicate neural oscillation bands and are caused predominantly by eye movements, heartbeat or cardiac events and muscle movements. As well as wavelet transforms, PCA, independent component analysis ICA and ensemble empirical mode decomposition (EEMD) with PCA are other methods which have been used as preprocessing techniques in signal processing. To fully use these methods time-locked stimulus events and event related potentials (ERPs) are generally required to be present in the EEG data in order to provide a feature reference. The MBD data set using the Emotiv EPOC device does not have any predefined reference channel or channels to use as a reference for eye movements (EOG). However, the common average reference (CAR) solution can be used when no formal reference signal is provided. A statistical justification for using CAR is given in (Ludwig et al. 2009) and (Mishra

et al. 2021) take a similar approach in averaging all signals across an epoch and then taking the correlation coefficient. The correlation coefficient is then used as a threshold in determining the suitability of the epoch data. While there is no direct evidence from search results supporting the use of correlation coefficients between CAR and individual channels for identifying good channels. The authors of (Mishra et al. 2021) use this method to exclude signals which are not well correlated, this approach could be a novel method for quality control in EEG preprocessing. It aligns with the general principles of ensuring data quality and consistency across channels, which are critical for reliable EEG analysis, and it is computational simple to implement. EEG data analysis with MNE-Python (Gramfort et al. 2013) enables comprehensive preprocessing, including artifact rejection and epoch selection. This package is a well-used and respected set of tools which are used for EEG preprocessing, rejecting epochs containing artifacts like eye movements, muscle activity, or noise which is crucial to ensure clean data for analysis. MNE-Python provides tools for automated epoch rejection based on predefined criteria, such as signal amplitude thresholds or peak-to-peak values, which help identify and exclude contaminated epochs. This approach improves data quality, enhances signal interpretation, and supports robust statistical analysis. Efficient epoch rejection contributes significantly to the reliability of EEG studies using MNE-Python.

## 2.1 Aims

The following aims identify the broad problems which will be explored in this research project. Processing the EEG data to generate a meaningful set of feature vectors will help to focus the learning of the GAN's and achieve stability and effective results.

- To develop a method for converting EEG signals into meaningful feature vectors to enable generation of visual representations.

- To explore the potential of GANs in interpreting and visualizing complex neurophysiological data.
- To explore and improve understanding of the relationship between brain activity and visual perception.

## 2.2 Objectives

The following objectives will provide guidance in the research process.

- Design and implement a preprocessing pipeline to clean and prepare EEG data for analysis and image generation.
- Develop a Classification/GAN architecture specifically tailored for translating EEG features into visual elements.
- Train the GAN model on a diverse dataset of EEG recordings paired with corresponding visual stimuli or mental imagery.
- Evaluate the quality and accuracy of the generated images compared to ground truth.

These aims and objectives will provide a structured approach to addressing the overarching research question.

**Chapter 3**

# Methodology

This study will follow a methodology which will attempt to treat and process the EEG data so that it can be used in a deep learning model and attempt to decode the visual stimulus data encoded in the collected EEG data. I will use the data collected from a signal subject over a period as discussed on detail in section 3.1. The encoded feature data contained in the processed EEG signals will then be used in a classification CNN model trained on a portion of the data to give a high degree of accuracy and feature separation as discussed in section 3.2. From this CNN-based model the output, the EEG encoding and class label conditioning parameter, will be passed to the image generation process. This image generation process will be achieved using an ACGAN designed model using key adaptations from previous related work as outline in previous chapters and discussed in detail in section 3.4. The complete methodology pipeline can be seen Fig 3 below.

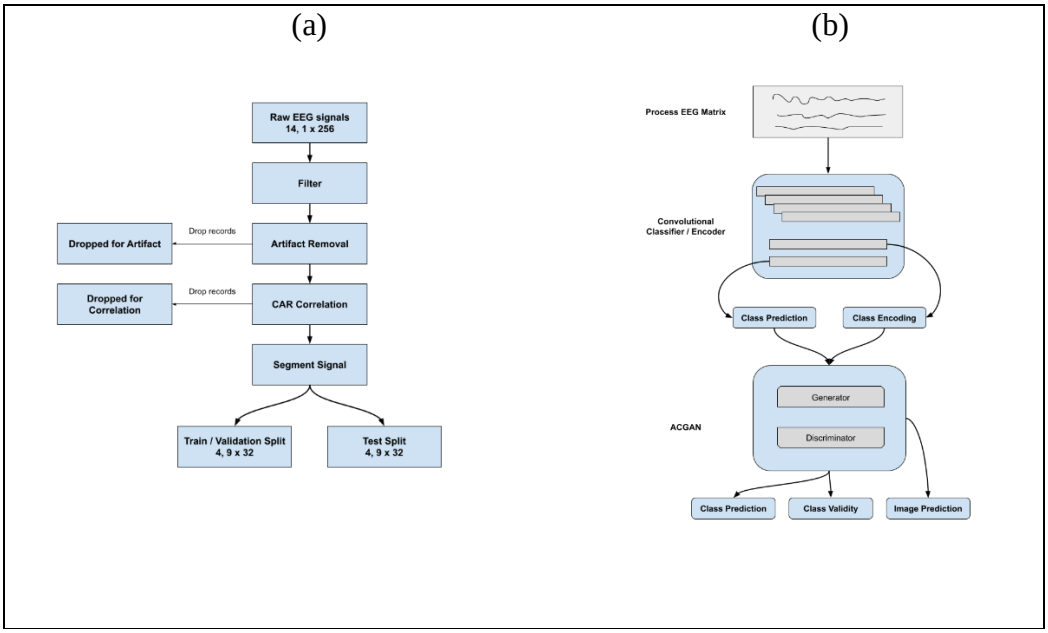


*Figure 5:* Proposed processing pipeline used for this study. (a) preprocessing of raw EEG signals to data format for model training and evaluation, (b) model training and evaluation pipeline.

## 3.1 MNIST data and EEG data

The MNIST data set (LeCun et al. 1998) is a widely used data set of handwritten digits which has been used as a benchmark in machine learning and computer vision. consisting of 70,000 grayscale images of handwritten digits represented as single channel 28x28 pixel images of digits [0 … 9] (10 class labels). Due to its simplicity, standardized format, and diverse digit representations. It has proved to be a reliable testbed for new methods in image recognition and ML learning tasks.

The EEG signals in the MNIST of Brain Digit 2022 (MBD) (Vivancos and Cuesta. 2022) data set consist of collected data in the form of 14 channels of 256 sample points at 128Hz sample rate which equates to 2 seconds of data per image stimulus. The MBD data consists of ~52000 epochs of 14 x 256 data series recorded over a period starting from 2014. There is a single subject, who was shown a single digit on a 65” TV screen for a period of 2 seconds. In between each digit the subject was shown a blank black screen. In the data set these blank images are labelled with -1, and during the processing these data have been removed from the data set. It is worth noting that the images presented to the subject are idealised version of the digits and not the actual MNIST images. See Fig 6 below which give an idea of the variation between the visual stimulus and the training set of images used in the classifier/encoder and ACGAN training. Each recording is giving a label which corresponds to the digit being viewed. In previous data set from the same author the index corresponding to the digit in the MNIST data set was also included in the raw data, however this data has been omitted from the data set used in this study, as the idealised image was shown and not an actual image. This adds a variational complexity to the challenge as the digit images are not consistent.

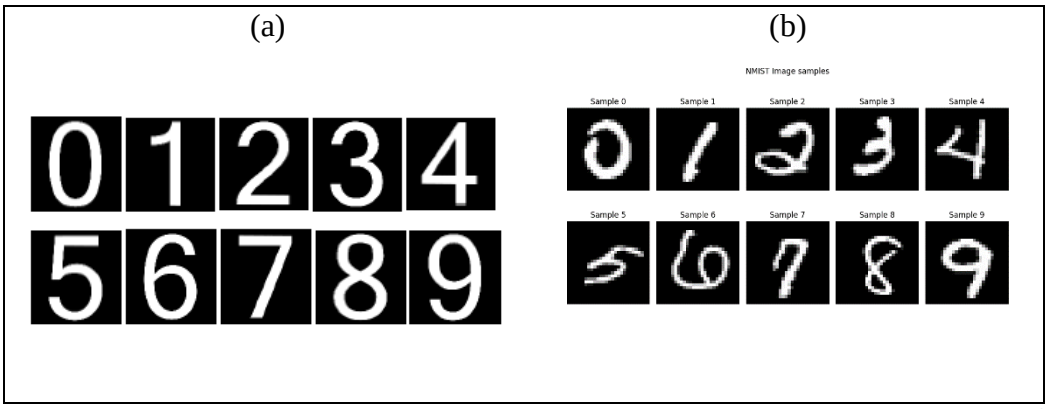


*Figure 6:* Visual stimuli images, (a) the images displayed to the subject on a tv screen, (b) a selection of images per class from the MNIST data set used for training.

The Emotiv EPOC device was used for collecting all data. The author mentions that the subject is seated and relaxed in the same surroundings for all recordings, minimising all movements, muscular and eye movements. Whilst all efforts were taken to minimise external effects "It is fair to say that some muscular artifacts could still be present in some of the recordings, since even slight muscular tension can alter EEG signals, and the device performance can be degraded over usage time or there can simply be other unaccountable variations in performance, so post processing and filtering should be considered**" (**Vivancos and Cuesta. 2022)

In this study the training data will be split 80 / 10 with the 10 % being kept for testing purposes and not used in any of the training schemes. With the 80 % being further split into training and validation 75 / 25 for training of the various networks.

## 3.2 Data Pre-Processing

The EEG signals are typically noisy and span a high dimensionality, and based on the searches of techniques, several filtering and artifact removal techniques are widely used to process the data. Typically, AC Notch filter, Band Pass filters and Artifact removal form the main techniques, and this study will follow those general guidelines as detailed below. However, care must be taken to not create unwanted effects from applying the filtering. After these filtering techniques a

further selection criterion based on correlation with the CAR is applied to achieve the final training/testing data set.

AC notch filer  is used to remove any effects from line noise potentially created by interference from the mains 50hz circuits. Since the oscillation signals are typically in the range of μVs AC fluctuations can completely overwhelm these signals. A zero-phase notch filter at 50Hz is applied, with a 1hz band.

Since we are interested in the oscillation bands which coincide with a range between 0.2Hz to 65Hz a $5^{th}$ order Butterworth, non-causal band pass filter is applied using 0.4hz low-pass and 60Hz high-pass band limits. A -6db cutoff frequency is applied to both band limits and Hamming window with 0.0194 passband ripple.

Each sample of 14 channels is defined as an epoch of length 2 seconds. Artifact remove and signal quality are managed by first removing bad epochs by thresholding the peak-to-peak variation in all channels of the epoch. The epochs deemed as good are then further refined by selecting correlation with the CAR for selected channels. Two methods of implementing the correlation selection are evaluated. As introduced in (Mishra et al. 2021) channels T7, P7, T8 and P8 are selected as channels containing discriminative feature associated with visual stimulus. See Fig 2 (b) for electrode positions.

Finally, the selected channel data which now consists of 4 channels x 256-time samples and the associated class label are resampled using a sliding window of 32 samples with a 4-sample overlap. This is consistent with previous studies, first used (Kumar et al. 2018) and further used in ThoughtViz: (Tirupattur et al. 2018) and NeuroGAN: (Mishra et al. 2023). This results in each channel being split into 9 x 32 matrix x 4 channels per class label. The result of this segmentation is to increase the number of training examples and concentrate the potential spectral data.

## 3.3 Classification Network

The classification and encoding of the EEG data forms a crucial part of the methodology of this study. The selection of an auxiliary conditioning GAN network model requires both the encoded latent space vector and a conditioning vector as input to the image generation process. This is based on the hypnosis that the encoded latent space vector contains conditional data which can be modulated with trainable weights to improve the generation of realistic images. The EEG data is presented in the form a time series per channel and contextual data may be important in both the time axis as well as the channel axis. So, the ability of CNN to learn contextual information indicates that these networks might be suited to this problem. This has been seen in the success of CNN in learning contextual data from matrix representations such as images. In earlier papers using EEG data 1D convolutions have been used. In this study I will use two sets of 2D Convolutional network layers and change the kernel size so that both time and channel axis are passed through the convolution together keeping any relation within the same set of filters. To achieve this the first layer has its kernel size oriented to the time axis, the second layer changes its orientation to the channel axis after which a max pooling layer is used to concentrate the key activations. A further two 2D convolutional layers are used to increase the filters with the intention of producing a more detailed set of filters to identify the smaller interactions. The output from the convolutional block is flattened then before passing onto a set of fully connected (FC) layers. Batch normalization is used before and after the convolutional block to help with regularization. The FC block takes the output vector from the convolutional block and reduces the latent dimension down to the final output dimension of 128, using a 10% dropout between each FC layer again to help with regularization and stop potential over training. The final FC layer is also batch normalised; this layer will be used as the latent space vector for input to the GAN generator. This layer is finally passed to the FC layer with a softmax activation and 10 nodes to determine the classification probability distribution.

The loss function is based on categorical cross-entropy, as the class labels are one-hot encoded before being used and the training label.

$$L = -\sum_i (t_i * \log(f(s_i)) \quad (6)$$

For class i, t & s are the ground truth and CNN score, $f(s_i)$ is the activation function. This can be simplified when the softmax activation is used, which is typical for multiclass where the logits are converted into a probability distribution.

$$f(s_i) = exp(s_i) / \Sigma(exp(s_j)) \quad \text{softmax} \quad (7)$$

$$L = -log(exp(s_p) / \Sigma(exp(s_j))) \quad (8)$$

$S_p$ is the CNN score for the correct class, $S_j$ is the sum taken over all classes.

By passing the final normalised layer before the classification layer i.e. the logits we are using the encoding which is ultimately converted to a discrete classification probability distribution so enabling the image generation to start with a distribution which contains class specific data.

Optimization of the loss function is achieved using adaptive moment estimation (Adam) over the alternative stochastic gradient descent (SGD). The Adam method has benefits over SGD descent as it incorporates an adaptive learning rate and momentum. The Adam optimizer converged quicker and to a higher degree of accuracy than an SGD optimizer.

## 3.4 Image generation and validity

In this study I have proposed the use of an auxiliary conditional GAN (ACGAN) based on that proposed by (Odena et al. 2017) which is an improved version of the original GAN proposed by (Goodfellow et al. 2014). The key differences are the discriminator has an additional class label output and the image generation is conditioned on the class label which leads to a more targeted image generation

process. As discussed previously in section 1.3 of the introduction. In addition to this I have proposed to use the generated latent space vector (logits) output from the classifier encoder as a replacement for the generated noise vector. In combination with this I have included the additional trainable layer which is applied to this distribution so give class specific spatial representations. I have examined two variations of implementing this modulation of the latent space distribution. The discriminator is then trained to give probability distributions over both the image source and the class label. Whilst the structural design of the ACGAN over the standard GAN is similar the modifications have been shown to increase the suitability when increasing the data set and permits separation into subsets of classes.

Using the trained classifier/encoder network to provide the predicted class labels and encoded latent space from the EEG data. The generator will be used to train the discriminator along with a sample of real images from the MNIST training set. The discriminator will then be used in an inference mode to train the generator to produce more realistic images. This will be achieved in a custom training loop, this forms part of the adversarial training methodology which is one of the key concepts of GAN architecture. The training loop is completed for ~2000 (n) epochs, where n is the number of epochs determined by both training time and compute resources.

**Chapter 4**

# Implementation

This section I will detail the implementation of the key parts developed during the research study. The hardware used for development and training consists of the following.

| | |
|---|---|
| Processor | 11th Gen Intel(R) Core(TM) i7-11800H @ 2.30GHz, 2304 MHz, 8 Core(s), 16 Logical Processor(s) |
| Memory | 16.0 GB |
| OS Name | Microsoft Windows 11 Pro |
| Version | 10.0.22631 Build 22631 |
| GPU | NVIDIA GeForce RTX 3070 |
| Memory | 8.0 GB |

The environment used for the whole study is based on a python 3.8, MSC v.1929 64 bit. On top of this base python environment, I have used a number of widely accepted standard tool packages such as Numpy, Pandas, Scikit, Keras and Tensorflow for example. A full listing of top-level packages and detailed dependency requirements for the environment are given in the appendix. The code scripts and notebooks are commented and where applicable citations have been given when code blocks have been used from papers or other sources.

## 4.1 Data handling

There are two main datasets which need to be handled during the study, MNIST and EEG data. To this end I have developed several software tools to enable the handling of the datasets. Whilst MNIST is widely used and built in as a dataset to most the key frameworks I chose to use an offline dataset for consistency and availability. The code supplied in the supplementary supporting documentation

has several methods used to load and sample data from the offline dataset and replicate some of the interfaces that would be used if accessing the online versions.

The EEG data is available from the MBD website[1] the data comes in the form of csv file with each line containing as class label and subsequent time step signal data for each electrode position. The raw CSV data was processed in chunks and saved to a Pandas data frame to make it easier to manipulate and use in subsequent steps. The raw data is saved to a pickle file and then loaded back into the script tools as required. The Emotiv EPOC data set, which contains 14 electrode positions has been used in this study. More detail is given in section 4.2 on the methods and tools developed and used for interacting with the data.

## 4.2 Preprocessing

The nature of EEG leads to potential for noisy signals, as has been discussed in earlier chapters there is a need for pre-processing of the EEG signals given in the dataset. From various earlier studies mentioned each have attempted to extract various features from the signals to use in the classification process. The nature of the EEG data in these earlier studies is significantly different from that of the MDB data set however it was worth investigating some of the techniques. Early investigations into features extraction used the highly comparative time-series analysis tools HCTSA (Fulcher et al. 2013). This package was used to assess if there were any key features such as entropy, relative band power. The authors (Kumar et al. 2018) proposed the use of a vector containing such statistical features. It is worth noting that they use the same capture device as that used in MDB however the period of sample is 10 seconds and there is specific time-locked stimulus. As opposed to 2 secs sample length and no specific stimulus for the MBD data. Using HCTSA I was able to identify some key top features and

[1] MindBigData 2022, https://huggingface.co/datasets/DavidVivancos/MindBigData2022. (accessed 2024)

establish that classification on these features alone would not achieve the required accuracy needed to provide a conditional label for image generation in the GAN. So, a combination of MNE epoch rejection, filtering, CAR, and correlation have been used to reduce the number of raw signals by excluding “bad” epochs. Epochs, in EEG signals being defined as a sample defined by a time interval, in this case 2 seconds, in which the subject receives a stimulus, and all available channels are recorded.

The preprocessing pipeline outlined is implemented as follows.

1. Load raw CSV data from dataset into a format which can be easily processed.
2. Using MNE inspect each epoch and reject “bad” data, where the maximum peak to peak of any of the 14 channels exceeds a threshold of 100μV.
3. Select from the raw data of all epoch indexes which are deemed good.
4. Apply bandpass and notch filtering to all channels.
5. Calculate the common average reference for all epochs.
6. Further refine the remaining data by removing all epochs in which the correlation coefficient of the channels T7,P7,T8,P8 do not meet the threshold value.
7. Finally apply a window function to the 4 selected channels to segment the epoch into sections of 32 samples, with each window having an overlap of 4 samples. 256 // (32 – 4) = 9. Save each of the resampled nominated channel as a 9 x 32 vector with a class label.

See Fig 5 (a) for a visualization of preprocessing pipeline. Implementation in code is available in the python script files and juypter notebooks available in the support documentation.

## 4.3 Classification and Encoding

The classification and encoding task will provide the key input vector and conditioning parameter to the GAN, the ACGAN design proposed in this study

and detailed in the section 4.4. In this study I have taken the approach to use a CNN model to classify the EEG signal data. As discussed previously there are several related papers which have proposed similar classification or encoder/decoder, attention models for this purpose. The MBD site has listed several studies which have reported varying success in this task. But notably only two of the authors have submitted code supporting their claims.

In this study I propose a CNN network using the Keras[2] deep learning API (Chollet et al. 2015)

The network consists of a set of 2D convolutional layers, which have the kernels oriented to the time axis for the first layer and then the channel axis for the second layer. This block is followed by a set of MaxPooling layers combined with 2D convolutions after which the feature vectors are flattened and passed to a series of dense fully connected layer with drop out, until the final layer using SoftMax activation to perform the classification probability distribution.

*Table 1:* EEG classifier/encoder CNN design

| ***Layer*** | ***Output Shape*** | ***Attributes*** | ***Parameters*** |
|---|---|---|---|
| Input | (None, 9, 32, 1) | | |
| BatchNormalization | (None, 9, 32, 1) | | 4 |
| 2D Convolution | (None, 9, 32, 128) | 128 filters, (9,1),pad 1, same | 640 |
| 2D Convolution | (None, 9, 32, 64) | 64 filters, (9,1),pad 1, same | 73792 |
| MaxPooling2D | (None, 9, 16, 64) | (1,2) | 0 |
| 2D Convolution | (None, 64, 13, 40) | 64 filters, (4, 25), pad 1, valid | 57644 |
| MaxPooling2D | (None, 64, 6, 40) | (1,2) | 0 |
| 2D Convolution | (None, 15, 5, 128) | 128 filters, (50,2), pad 1, valid | 512128 |
| Flatten | (None, 9600) | | 0 |
| BatchNormalization | (None, 9600) | | 38400 |
| Dense | (None, 512) | 512 nodes, Relu | 4915712 |
| Dropout | (None, 512) | 0.1 | 0 |
| Dense | (None, 256) | 256 nodes, Relu | 131328 |
| Dropout | (None, 256) | 0.1 | 0 |
| Dense | (None, 128) | 128 nodes, Relu | 32896 |
| Dropout | (None, 128) | 0.1 | 0 |
| BatchNormalization | (None, 128) | | 512 |
| Dense | (None, 10) | Softmax, L2 regulariztion | 1290 |

[2] Keras, https://keras.io/api/ (accessed 2024)

The final two layers are outputs from the network, predicted label, and encoded EEG latent space vector. The network is then trained on a portion of the total training data with a validation split of 0.25. The network is trained for a maximum of 150 epochs, with a batch size of 128. Optimization was performed by Adam with a starting rate of 0.001 and momentum 0.8.

## 4.4 Generation

The ACGAN design employed for this study has been derived from the original paper (Odena et al. 2017) with the modifications of a modulation layer applied to the latent space and embedding applied to the class label as discussed earlier in the paper.

The modifications are in the generator where the latent space distribution is provided by the encoded EEG feature space from the classifier/encoder, this replaces the gaussian noise vector used by previous papers. This is further modulated by two trainable parameters. Following this the modulated vector is multiplied with the class conditioning provided by creating an embedding based on the class label. It is worth noting that the final convolutional layer which create the generated image array uses a TanH activation. This is because TanH provides outputs in the value range of [-1, 1] which match the normalised range of many images. For example, when normalising an image matrix with values of $[0 \dots 255]$ in the form of $( p_x - 127.5) / 127.5$ $p_x$ being the pixel value a x in the matrix.

The discriminator follows the design as outlined in the original paper. The generator and discriminator are then combined with the discriminator trainable parameter set to false and the generator is trained inside the combined GAN end to end.

*Table 2:* ACGAN Genertor network model design

| ***Layer*** | ***Output Shape*** | ***Attributes*** | ***Parameters*** |
|---|---|---|---|
| Input (class label) | (None, 1) | | 0 |
| Embedding | (None, 1, 128) | | 1280 |
| Flatten | (None, 128) | | 0 |

| | | | |
|---|---|---|---|
| Input (EEG latent space) | (None, 128) | | 0 |
| Modulation of EEG Layer | (None, 128) | | 256 |
| Multiply | (None, 128) | | 0 |
| Dense | (None, 6272) | Nodes 6272, Relu | 809088 |
| Reshape | (None, 7, 7, 128) | | 0 |
| BatchNormalization | (None, 7, 7, 128) | | 512 |
| UpSample2D | (None, 14, 14, 128) | nearest | 0 |
| 2D Convolution | (None, 14, 14, 128) | 128, kernel 3, stride 1, pad same, Relu | 147584 |
| BatchNormalization | (None, 14, 14, 128) | | 512 |
| UpSample2D | (None, 28, 28, 128) | Nearest | 0 |
| 2D Convolution | (None, 28, 28, 64) | 64, kernel 3 stride 1, pad same, TanH | 73792 |
| BatchNormalization | (None, 28, 28, 64) | | 256 |
| 2D Convolution | (None, 28, 28, 1) | 1, kernel 3, stride 1, pad same, Tanh | 577 |

*Table 3:* ACGAN Disciminator design

| ***Layer*** | ***Output Shape*** | ***Attributes*** | ***Parameters*** |
|---|---|---|---|
| Input (generated image) | (None, 28, 28, 1) | | 0 |
| 2D Convolution | (None, 14, 14, 16) | 16, kernel 3, stride 2, pad same | 160 |
| LeakyRelu activation | (None, 14, 14, 16) | 0.2 | 0 |
| Dropout | (None, 14, 14, 16) | 0.25 | 0 |
| 2D Convolution | (None, 7, 7, 32) | 32, kernel 3, stride 2, pad same | 4640 |
| ZeroPad | (None, 8, 8, 64) | | 0 |
| LeakyRelu activation | (None, 8, 8, 64) | 0.2 | 0 |
| Dropout | (None, 8, 8, 64) | 0.25 | 0 |
| BatchNormalization | (None, 8, 8, 64) | 0.8 | 128 |
| 2D Convolution | (None, 4, 4, 128) | 64, kernel 3, stride 2, pad same | 18496 |
| LeakyRelu activation | (None, 4, 4, 128) | 0.2 | 0 |
| Dropout | (None, 4, 4, 128) | 0.25 | 0 |
| BatchNormalization | (None, 4, 4, 64) | 0.8 | 256 |
| 2D Convolution | (None, 4, 4, 128) | 128, kernel 3, stride 1, pad same | 73856 |
| LeakyRelu activation | (None, 4, 4, 128) | 0.2 | 0 |
| Dropout | (None, 4, 4, 128) | 0.25 | 0 |
| Flatten | (None, 2048 | | 0 |
| Dense (validity confidence) | (None, 1) | | 2049 |
| Dense (class prediction) | (None, 10) | | 20490 |

The modulation of the EEG latent space vector (MoG) is performed using the custom modulation (MoG) layer which has been derived and updated from the original proposed layer in (Gurumurthy et al. 2017) however, unlike in previous papers this is applied directly to the EEG latent space *z*. The trainable weights in the form of mean (μ) and variance (σ) which are applied to the sampled point $z(e)$

$$z = \mu_i + \sigma_i \epsilon_i \qquad (9)$$

$$w = z_{eeg} * (\mu_i + \sigma_i \epsilon_i) \qquad (10)$$

The GAN was trained using the training images from the MNIST data set along with the training EEG signals to train the discriminator and generator. The training was left to run for 2000 epochs after which time image generation was stabilised. An Adam optimizer with learning rate of 0.0002, beta = 0.5 and decay of 1e-6 was used for both the generator and discriminator.

**Chapter 5**

# Results & Discussion

This section I will present the findings of this work and discuss them in the context of the original aims & objectives. Taking each section of the pipeline in turn.

## 5.1 Signal processing

Starting with the raw data set from MBD there are ~52000 epochs of data consisting of 14 channels, with 256 samples at a sample rate of 128Hz, which equates to 256 / 128 = 2 seconds time for each epoch. Each sample points equates to ~ 8msec. Each class i.e digit number in the range [0 … 9] has roughly the same number of epochs leading to a relatively even class distribution. I have endeavoured to keep this even class distribution throughout the signal processing step.

*Table 4:* Sample counts per class from MBD, starting raw counts, good epochs counts per class after rejection of data using MNE thresholding, corr threshold counts after filtering for correlation.

| *Class Label* | *Raw Sample count* | *Good Epochs* | *Corr Threshold* |
|---|---|---|---|
| 0 | 5212 | 3845 | 1180 |
| 1 | 5080 | 3798 | 1090 |
| 2 | 5196 | 3809 | 1130 |
| 3 | 5294 | 3950 | 1100 |
| 4 | 5079 | 3765 | 1100 |
| 5 | 5256 | 3916 | 1180 |
| 6 | 5218 | 3868 | 1170 |
| 7 | 5069 | 3780 | 1195 |
| 8 | 5241 | 3861 | 1115 |
| 9 | 5250 | 3917 | 1150 |

Initial investigation of the data was looking at the relevance of selecting several statistical features based on relative band power or an entropy measure for example. However, it can be seen in the distributions for each class that there is little discriminable variance in these measures, so I determined that there encoded data is embedded in high dimensionality, see Fig 3 introduced in section 2. I used

the “catch24” feature set, “a set of high performing features for classification which have been selected based on performance across a wide selection of common time-series classification problems” (Lubba et al. 2019). The catch24 set is a subset of those methods used in the HCTSA (Fulcher et al. 2013).

The nature of EEG signals means there are likely some of epochs in the raw data which contain noise or artifacts which realistically mean the whole epoch is usable. This invariably leads to some usable data being thrown away. But the repair and retrieval of this data was not necessary for this study and could form part of future work. The key effects that I wanted to remove from the raw data set where those epochs which are affected by noise but mostly by eye, or muscular movements. I used the functionality in the MNE package (Gramfort et al. 2013) to remove epochs using a maximum peak to peak threshold 100µV this threshold being set based on previous publications (Sanei and Chambers 2013). Examples of the rejected epochs show the effects of such artifacts as eye movements for example. The activity is particularly notable around the frontal zones (F7,AF3,AF4,F8). This is compared to an epoch which does not contain external noise or artifacts. See Fig 7 below, which shows examples of localised electrode activation activity for (a) a rejected data sample, and (b) an accepted sample.

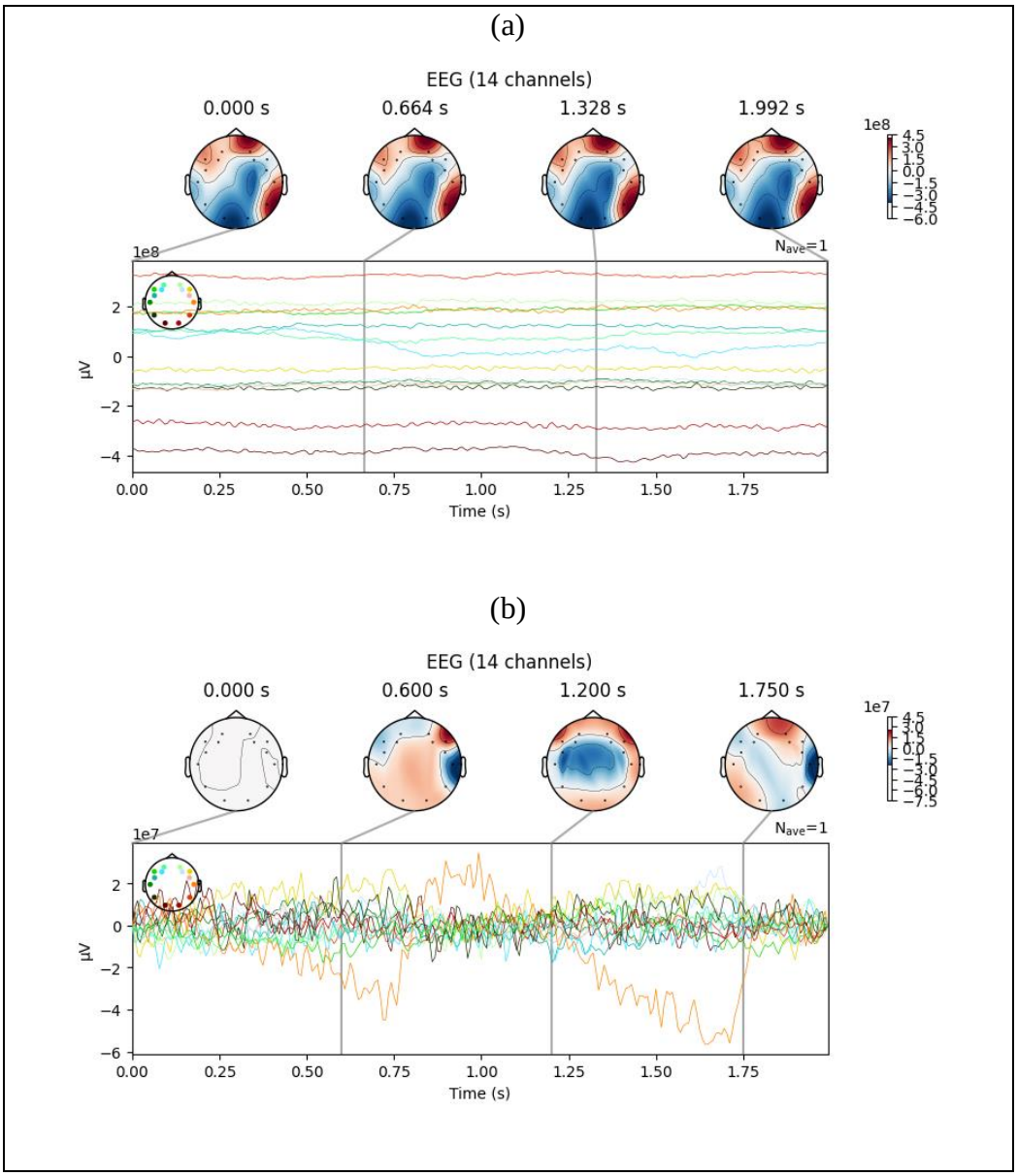


*Figure 7:* Topological plot of electrode activations from MNE routines, (a) exmaple of rejected epoch, high concentration centered on the frontal regiogns and charactised by electrode traces which have little diffiniation and wide distribution, (b) example of an accepted epoch, electrode trace distributions a clustered together around a common mean and the activations are more distributed around the electrode positions.

Taking the remaining epochs which have passed the artifact rejection stage is the band pass filter all the channel data for each epoch. To assess the value of the filtering process we calculate the signal to noise ratio of the filtered signal. Using the CAR principles in which "obtaining the best linear unbiased estimate of X is equivalent to subtracting the average of all other channels from the current channel" (Ludwig et al. 2009) where X is the true signal. The CAR is subtracted from each channel and then the SNR can be calculated as.

$$SNR = 10 * \log_{10} \frac{\sum_{(i=1)}^{N} x_i^2}{\sum_{(i=1)}^{N} (s_i - x_i)^2} \qquad (11)$$

where N is the number of sample points, $x_i$ is the noise reduced.

signal at time i, and $s_i$ is the band-pass filtered signal at time i.

Finally, the correlation coefficient of the selected four channels is taken with reference to the CAR. From the papers already mentioned a direct correlation with CAR, where the CAR is a proxy for the denoised signal . Signals are then selected which have a correlation above the threshold. As a measure of effectiveness, a ratio of filtered signal to the CAR resultant is taken which give an indication as to the relative effectiveness of the correlation selection. The SNR shown is the effective mean taken before and after the CAR correlation selection. By removing overly noisy signals the effective SNR is increased for each class, this is shown in how effective the classifier can separate the classes and provide adequate accuracy.

*Table 5:* Final selected sample per class after all step in the processing pipeline. SNR measrure of efectiveness of filtering, epoch rejection and correltion.

| | ***SNR measure (relative correlation effectiveness)*** | | |
|---|---|---|---|
| ***Class label*** | ***Before CAR*** | ***CAR > 0.90*** | ***Selected items*** |
| 0 | 0.378 | 2.126 | 203 |
| 1 | 0.521 | 2.213 | 204 |
| 2 | 0.374 | 2.131 | 191 |
| 3 | 0.352 | 2.239 | 193 |
| 4 | 0.323 | 2.191 | 193 |
| 5 | 0.312 | 2.068 | 196 |
| 6 | 0.474 | 2.139 | 197 |
| 7 | 0.299 | 1.983 | 194 |
| 8 | 0.382 | 2.155 | 195 |
| 9 | 0.280 | 2.077 | 192 |

After sampling method, the selected items per class are used as the training set of valid samples.

Once the final set of selected epochs have been collected each channel/epoch per class is then segmented into the 9 x 32 sliding windows. Each window has an overlap of 4 samples with reference to the previous window. This effectively transforms the data from X epochs of 4 x 256 in the X data of 9 x 32 for each class

label in the dataset. This approach helps capture temporal dependencies and patterns at different scales, as the overlapping regions provide context between adjacent windows. It also enables the network to learn features that are invariant to small time shifts, improving the model's robustness to temporal variations in the input signal. This is the final training dataset used to start the classification and encoding stage.

## 5.2 Classification and encoding

The processed EEG signal data now must be classified and encoded into a latent space vector. This classification task proved to be one of the most difficult parts of the entire process. This has to do with the large variance and high dimensionality of the EEG data. The rationale behind selecting some of the threshold values in the previous steps stem from the classification results which are achieved using data selected at differing thresholds. The accuracy of the classification and encoding has an impact on the following stage of image generation as the predicted class label and accompanying encoded vector are then feed into the generator section of the ACGAN.

The classifier model was trained on two sets of data. The post filtered data prior to correlation selection and the data set after final correlation selection. From the following figures and tables, the difference in accuracy improves from ~ 50% i.e. little better than random choice, up to ~90% after only taking the highly correlated signals. I have used a combination of precision and recall scores to give the F1 harmonic mean of precision and recall as the final accuracy evaluation score of the network.

$$Precision = \frac{True\ positive}{(True\ positive + False\ positive)} \qquad (12)$$

$$Recall = \frac{True\ positive}{(True\ positive + False\ negative)} \qquad (13)$$

$$F1 = \frac{2}{(\frac{1}{Precision} + \frac{1}{Recall})} \qquad (14)$$

Table 6 below shows the accuracy measures for the results of classification on a per class basis, along with the overall average. Fig 8 shows the confusion matrix between class predictions. Finally, Fig 9, show the t-SNE map of class separability this clearly show the extent of class confusion which is caused by the remaining noise in the signals prior to extracting the data which had a high correlation threshold.

*Table 6:* Evaluation of EEG classifier/encoder after training without the correlation selection.

| ***Label*** | ***precision*** | ***Recall*** | ***F1-score*** |
|---|---|---|---|
| Class 0 | 0.48 | 0.48 | 0.48 |
| Class 1 | 0.46 | 0.43 | 0.45 |
| Class 2 | 0.43 | 0.40 | 0.41 |
| Class 3 | 0.50 | 0.51 | 0.51 |
| Class 4 | 0.46 | 0.39 | 0.42 |
| Class 5 | 0.45 | 0.48 | 0.46 |
| Class 6 | 0.45 | 0.46 | 0.45 |
| Class 7 | 0.41 | 0.47 | 0.47 |
| Class 8 | 0.43 | 0.43 | 0.43 |
| Class 9 | 0.50 | 0.47 | 0.47 |
| | | | |
| Accuracy | | | 0.46 |
| Macro avg | 0.46 | 0.46 | 0.46 |
| Weighted avg | 0.46 | 0.46 | 0.46 |

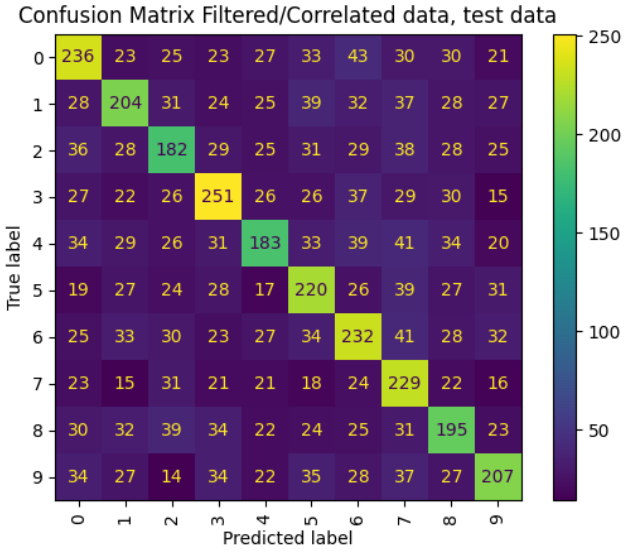


*Figure 8:* Class confusion matrix evaluated on the model trained with data prior to final correlation.

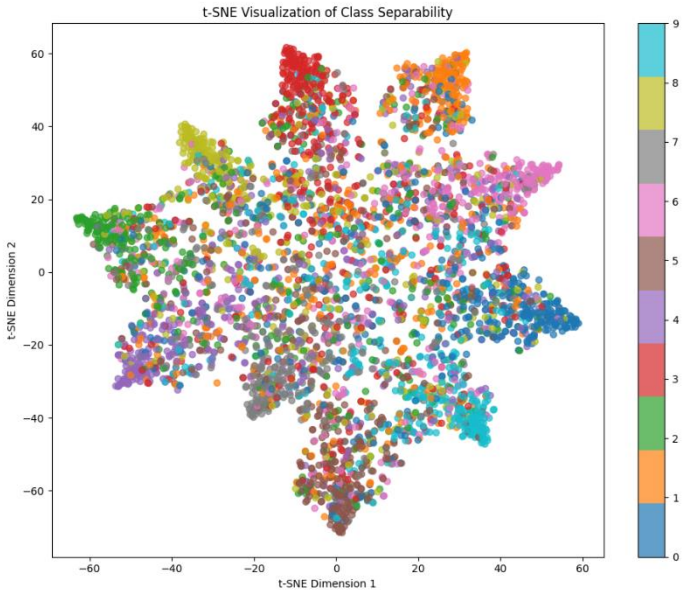


*Figure 9:* t-SNE Class seperability, clearly shows whilst classes are strating to be seperated there is still a large vaiance amons the class data leding to confusion in the network.

Comparing Table 6 accuracy of 0.46, with Table 7 below of 0.92 the effect of poorly correlated noisy signals can be seen clearly. Fig 11, also clearly show the improvement in class separation when using the correlated data.

*Table 7:* Evaluation of EEG classifier/encoder after training with the correlation selection.

| ***Label*** | ***precision*** | ***Recall*** | ***F1-score*** |
|---|---|---|---|
| Class 0 | 0.92 | 0.88 | 0.90 |
| Class 1 | 0.93 | 0.90 | 0.92 |
| Class 2 | 0.91 | 0.87 | 0.89 |
| Class 3 | 0.94 | 0.94 | 0.94 |
| Class 4 | 0.91 | 0.93 | 0.92 |
| Class 5 | 0.89 | 0.91 | 0.90 |
| Class 6 | 0.92 | 0.89 | 0.90 |
| Class 7 | 0.90 | 0.96 | 0.93 |
| Class 8 | 0.93 | 0.95 | 0.94 |
| Class 9 | 0.92 | 0.93 | 0.93 |
| | | | |
| Accuracy | | | 0.92 |
| Macro avg | 0.92 | 0.92 | 0.92 |
| Weighted avg | 0.92 | 0.92 | 0.92 |

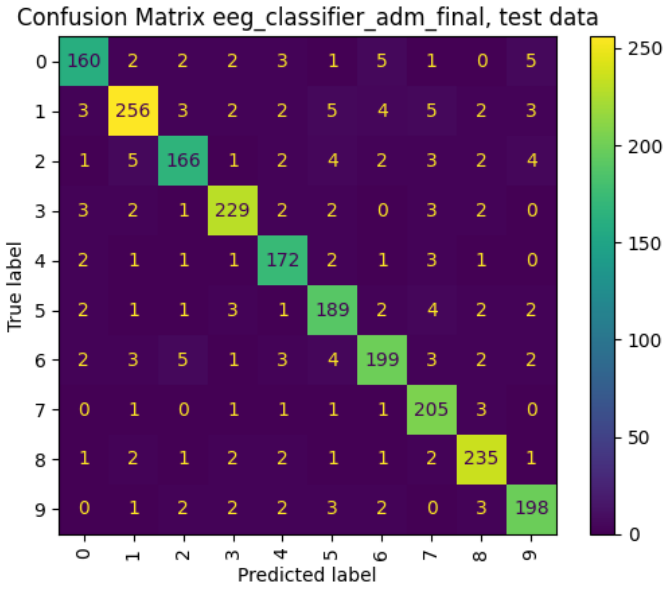


*Figure 10:* Class confusion matrix evaluated on the model trained with data after final correlation.

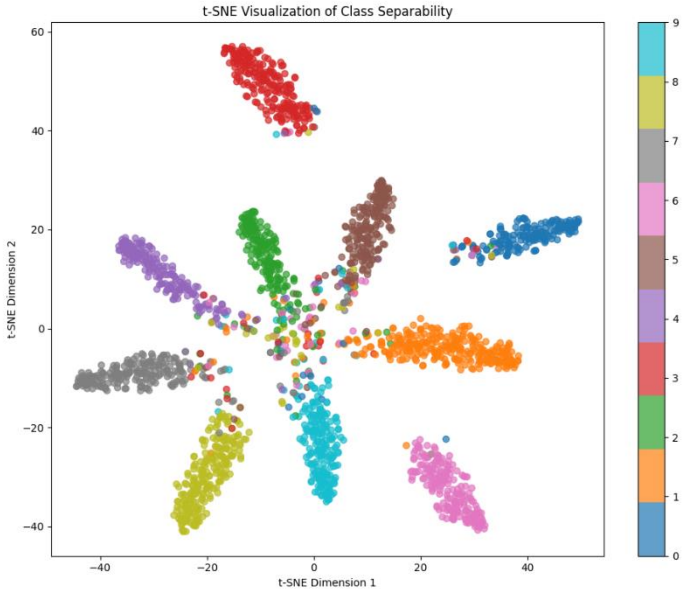


*Figure 11:* t-SNE Class seperability, classes are well seperated but there is still a vaiance amonst the class data around the origin.

There is clear correlation between the quality of the data and the success of the classification network. However, the filtering and correlation selection process is most likely throwing away good data which possibly could be used. Given more work on the design of the classification network the selection criteria could be reduced allowing more data to be used, or the network could be trained with more

finely tuned hyperparameters or K-fold cross validation which may help the network in generalising more to identify the key features whilst still tolerating additional noise.

Comparing the final good model class separation in Fig 11 with that of the class separation from a top performing CNN classifier specifically for the MNIST data set. See Fig 12 below. Particularly classes 4 & 9, there is still an amount of confusion with some outlier data within each class. This variance within the MNIST data set is similar to that in the EEG classifier and together their variance I believe could be a contributing factor to the variability of the generated image from the ACGAN.

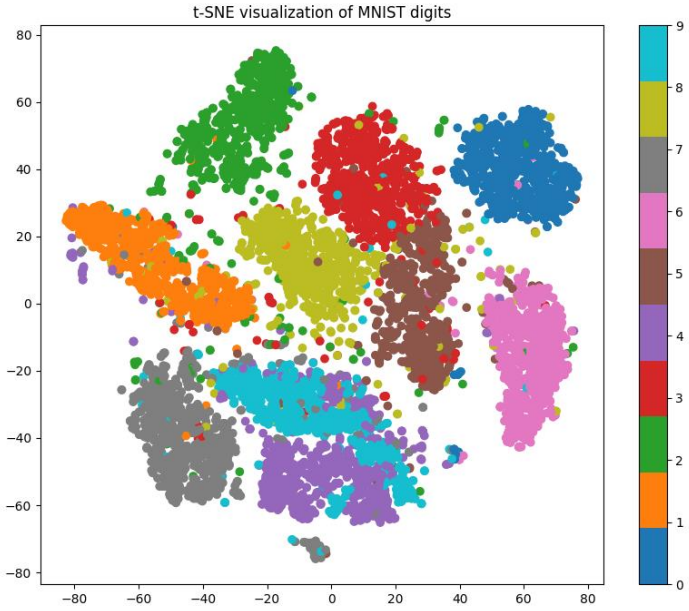


*Figure 12: t-SNE plot of class seperation for the MNIST data set, using a top performing CNN classification model.*

## 5.3 Image Generation

Using the trained EEG classifier / encoder to produce class conditioning labels and encoded latent space vectors I now have separated data to be able to use the ACGAN to create an image. The ACGAN is trained as explained in the section 4.4 above using the EEG data and the pretrained classifier to produce an input space vector and a class conditioning label predicted from the classifier. Fig 13 below shows an example of the resultant latent space encoding from the classification encoding model, overlaid with a random Gaussian noise vector. Despite the EEG encoding's superficial similarity to noise, the classifier's capacity to extract meaningful information from it, as evidenced by the distinct class probabilities, supports the hypothesis that the vector contains significant encoded features.

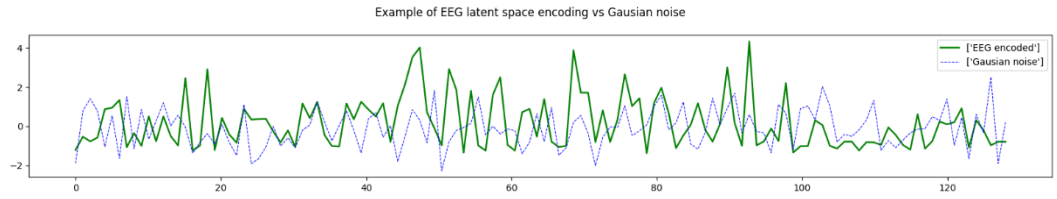


*Figure 13:* Example plot showing the distribution of the latent space vector from the EEG encoder compared with that of a random gausian generated distribution.

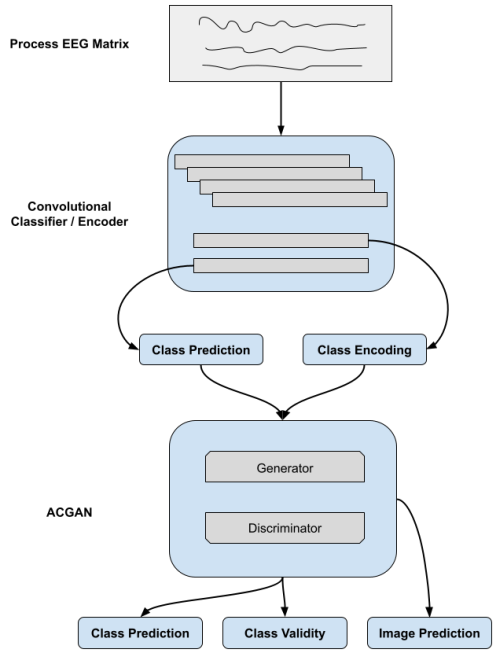


*Figure 14:* ACGAN training pipeline. Filtered, correlated and windowed EEG data as input. Class pridiction, validity and generated iamge as output.

I have used as a baseline the images generated from a base GAN using the MNIST image set, this GAN is trained using the MNIST training set and an input vector which is a gaussian noise vector. This is used to compare against the images generated from the EEG input vector. Fig 15 below shows a matrix of images generated by the GAN at the end of training epoch 2000. Variability in generated images across all classes can be seen. Fig 16 shows the images generated and validity scores from the compiled model after training, using the test data for a sample from each class. Table 8 gives the evaluation metrics for each class based on the test data set. The evaluation metrics used are explained in more detail below.

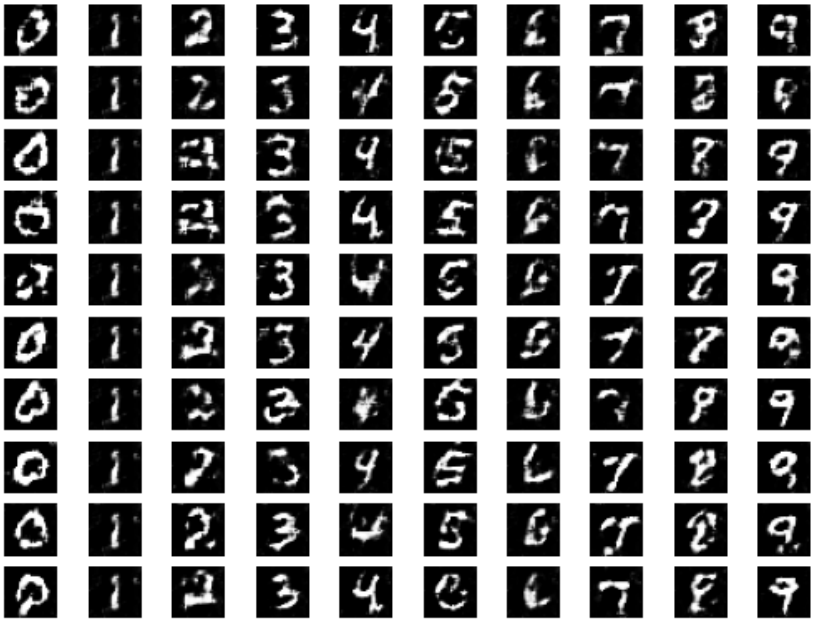

*Figure 15:* Base GAN image matrix after 2000 epochs of training, images generated from training data at the end of epoch.

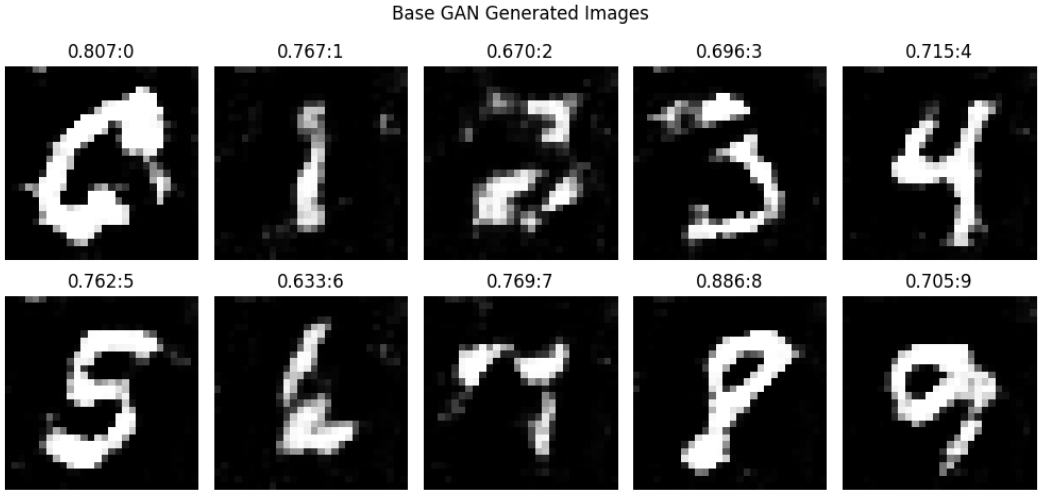


*Figure 16:* Base GAN sampled generated images.

*Table 8: Baseline GAN , validity confidence, Dice score and SSIM scores evaluated over all MNIST class images.*

| ***Class label*** | ***Validity score*** | ***Dice Score*** | ***SSIM score*** |
|---|---|---|---|
| 0 | 0.807 | 0.683 | 0.273 |
| 1 | 0.767 | 0.658 | 0.626 |
| 2 | 0.670 | 0.482 | 0.261 |
| 3 | 0.695 | 0.475 | 0.113 |
| 4 | 0.715 | 0.486 | 0.273 |
| 5 | 0.761 | 0.632 | 0.383 |
| 6 | 0.632 | 0.502 | 0.269 |
| 7 | 0.768 | 0.245 | 0.186 |
| 8 | 0.886 | 0.659 | 0.327 |
| 9 | 0.704 | 0.573 | 0.372 |
| | | | |
| mean | 0.740 | 0.540 | 0.31 |

During the training of the ACGAN on the EEG data, I explored three different network architectures see Fig 17. Two of these architectures combined the MoG layer and the conditional label embedding in a unique way. The three variations I evaluated were as follows...

1) No modulation layer.
2) multiplication of MoG layer with the Embedding.
    - Allows the embedding to scale or modulate the latent vector values.
    - Can be seen as a form of feature-wise attention.
    - Can allow for multiplicative interactions between latent and class features.
3) concatenating of MoG layer with the Embedding.
    - Can allow for multiplicative interactions between latent and class features.
    - Can allow for multiplicative interactions between latent and class features.
    - Can allow for multiplicative interactions between latent and class features.

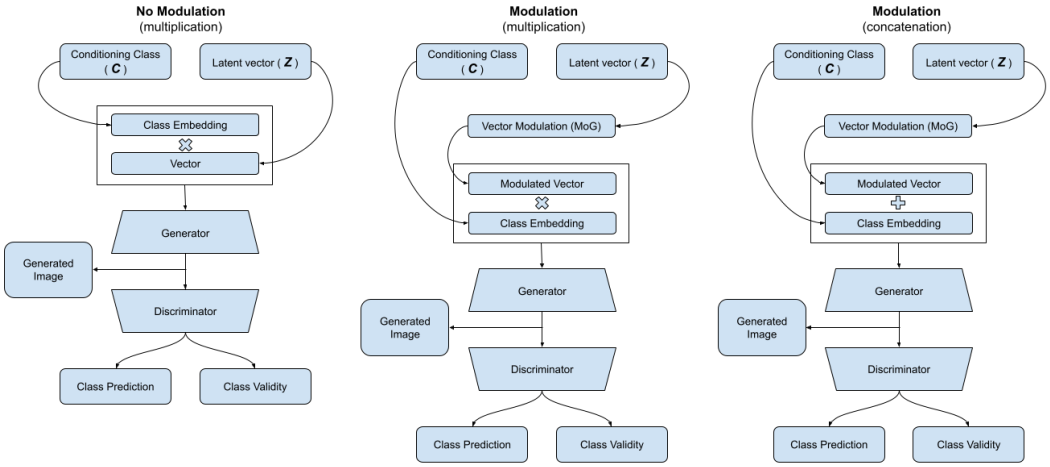


*Figure 17:* ACGAN model variations.

There are two metrics used to evaluate the predicted image generated by the ACGAN generator along with classification and confidence from the discriminator. As the discriminator will produce both a classification confidence and a predicted class label. The prediction will be evaluated against the ground truth label, the classification label, and the predicted confidence. For the image generation I will use two measures to assess the image against a selected MNIST image as ground truth. Calculating a Dice score, also known as the Dice Coefficient or F1 Score. This metric used to evaluate the similarity between two sets, often employed in image segmentation tasks to compare a predicted segmentation against a ground truth image. This measure is analogous to the F1 score used to evaluate the classification model, however in the case of image comparison it measures the overlap between the predicted and true regions, providing a value between 0 and 1, where 1 indicates perfect overlap (i.e., identical segmentation) and 0 indicates no overlap. Mathematically, it is defined as twice the intersection of the predicted and ground truth regions divided by the sum of the pixels in both regions.

$$Dice\ score = \frac{2 * |\ prediction \cap ground\ truth\ |}{|\ prediction\ | + |\ ground\ truth\ |} \qquad (15)$$

The measure balances precision and recall, penalizing both false positives and false negatives. The Dice Score is sensitive to slight variations in overlap, making it an effective measure for evaluating segmentation accuracy in images. I also use the Structural Similarity Index[3] (SSIM). SSIM is a perceptual metric used to evaluate the quality of generated images by comparing them directly to ground truth images. Unlike traditional measures like Mean Squared Error (MSE) that only assess pixel-by-pixel differences, SSIM considers human visual perception, focusing on structural information, luminance, and contrast to assess image similarity. It produces a score between -1 and 1, where 1 indicates perfect similarity. SSIM is calculated by comparing local patterns of pixel intensities normalized for luminance and contrast, providing a more holistic view of image quality. It is particularly useful for evaluating image generation tasks, such as super-resolution, denoising, and inpainting, where maintaining structural fidelity and visual quality is crucial. By accounting for variations in structure, texture, and lighting, SSIM offers a robust measure of how closely generated images resemble their corresponding ground truth, making it an effective tool for evaluating generative models.

$$SSIM(x,y) = [l(x,y)]^{\alpha} * [c(x,y)]^{\beta} * [s(x,y)]^{\gamma} \quad (16)$$

Where…

$l(x,y) = \frac{(2\mu x * \mu y + C1)}{(\mu x^2 + \mu y^2 + C1)}$ μ mean of pixel value (17)

$c(x,y) = \frac{(2\sigma x * \sigma y + C2)}{(\sigma x^2 + \sigma y^2 + C2)}$ σ std dev of pixel value (18)

$s(x,y) = \frac{(\sigma xy + C3)}{(\sigma x * \sigma y + C3)}$ σxy covariance of pixel value (19)

[3] Structural similarity index measure, https://en.wikipedia.org/wiki/Structural_similarity_index_measure ( accessed 2024 )

- C1, C2, and C3 are constants to avoid division by zero.

Typically, α = β = γ = 1 and C3 = C2/2, which simplifies the SSIM formula to:

$$SSIM(x,y) = \frac{((2\mu x * \mu y + C1)(2\sigma xy + C2))}{((\mu x^2 + \mu y^2 + C1)(\sigma x^2 + \sigma y^2 + C2))} \quad (20)$$

The SSIM scores are calculated using the scikit-image[4] functions.

All results below use the final trained Classifier model, along with the Final ACGAN model trained for each option. Evaluation is done using the test data set which has not been used in the training process for ether of the two models.

Results from option 1: ACGAN, no modulation layer.

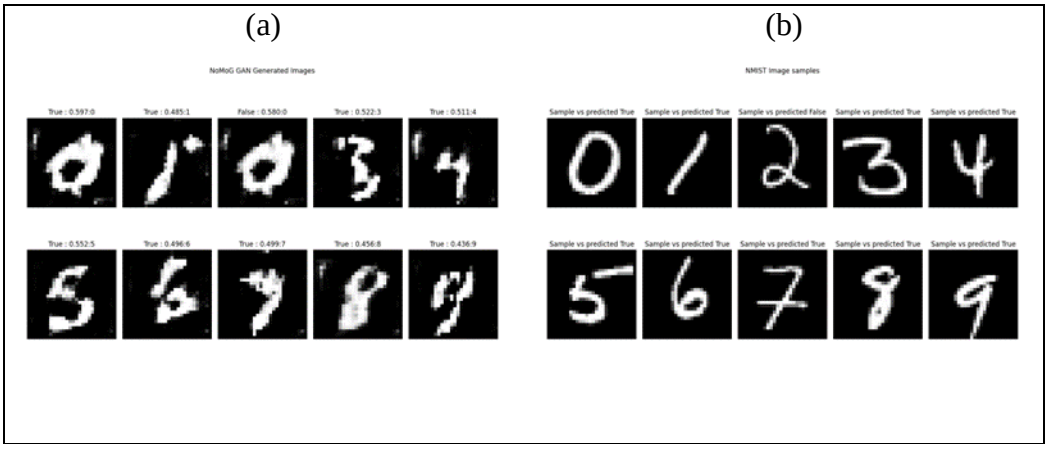


*Figure 18:* ACGAN generated images vs the sampled images from MNIST (ground truth). (a) images generated from a random selection from each class using the test data. The classification vs the ground truth, the confidence score and the predicted class label from the discriminator, (b) Random selected image from corresponind class from the MNIST test data set for comparison.

From Fig 18, the classifier predicted the incorrect class for class 2. In Table 9 whilst the validity score for the for the false prediction is comparable with the correctly predicted class 0, the SSIM score is suitably low as the predicted image of a 0 has a large variance from the sample image of a 2.

[4] https://scikit-image.org/docs/stable/auto_examples/transform/plot_ssim.html ( accessed 2024 )

*Table 9:* Evaluation matrix for generated images vs the sampled images from MNIST (ground truth).

| *Class label* | *Prediction* | *Validity score* | *Dice Score* | *SSIM score* |
|---|---|---|---|---|
| 0 | True | 0.597 | 0.581 | 0.229 |
| 1 | True | 0.484 | 0.408 | 0.449 |
| 2 | False | 0.58 | 0.285 | 0.098 |
| 3 | True | 0.521 | 0.437 | 0.293 |
| 4 | True | 0.511 | 0.408 | 0.380 |
| 5 | True | 0.552 | 0.467 | 0.316 |
| 6 | True | 0.495 | 0.567 | 0.401 |
| 7 | True | 0.499 | 0.198 | 0.139 |
| 8 | True | 0.456 | 0.568 | 0.271 |
| 9 | True | 0.436 | 0.454 | 0.334 |

Results from option 2: ACGAN with Modulation layer, multiplication with embedding.

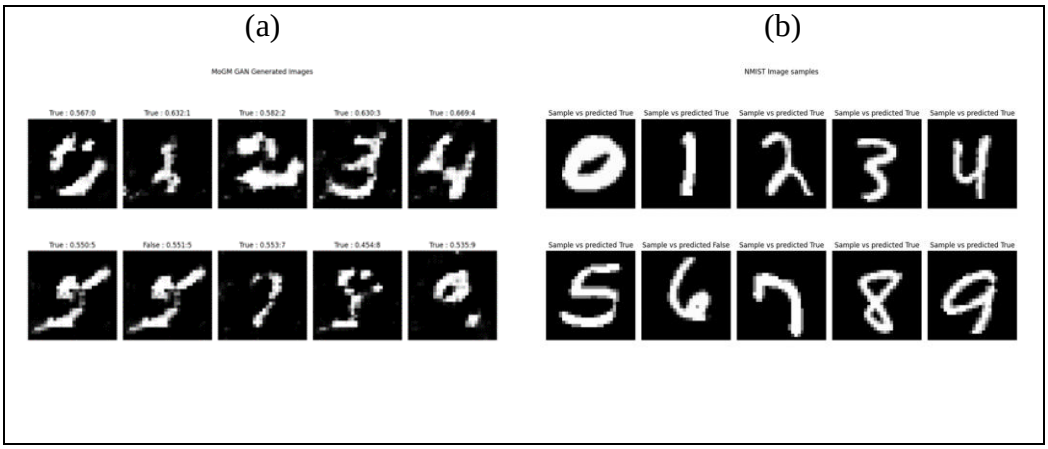


*Figure 19:* ACGAN MoGM generated images vs the sampled images from MNIST (ground truth). (a) images generated from a random selection from each class using the test data. The classification vs the ground truth, the confidence score and the predicted class label from the discriminator, (b) Random selected image from corresponind class from the MNIST test data set for comparison.

From Fig 19, the classifier predicted the incorrect class for class 6. In Table 10 whilst the validity score for the for the false prediction is comparable with the correctly predicted class 6, the SSIM score is also similar this is because the 5 & 6 look similar and occupy similar pixel spaces. The Dice score for class 7 is low, but the MNIST sample is an outlier variant from the rest of the 7 class, compare back to Fig 18 (b) class 7.

*Table 10:* Evaluation matrix for ACGAN MoGM generated images vs the sampled images from MNIST (ground truth).

| *Class label* | *Prediction* | *Validity score* | *Dice Score* | *SSIM score* |
|---|---|---|---|---|
| 0 | True | 0.567 | 0.538 | 0.128 |
| 1 | True | 0.632 | 0.467 | 0.474 |
| 2 | True | 0.582 | 0.476 | 0.184 |
| 3 | True | 0.629 | 0.446 | 0.218 |
| 4 | True | 0.668 | 0.416 | 0.193 |
| 5 | True | 0.550 | 0.370 | 0.115 |
| 6 | False | 0.550 | 0.234 | 0.118 |
| 7 | True | 0.553 | 0.159 | 0.164 |
| 8 | True | 0.453 | 0.415 | 0.281 |
| 9 | True | 0.535 | 0.382 | 0.199 |

Results from option 3: ACGAN with Modulation layer, Concatenation with embedding.

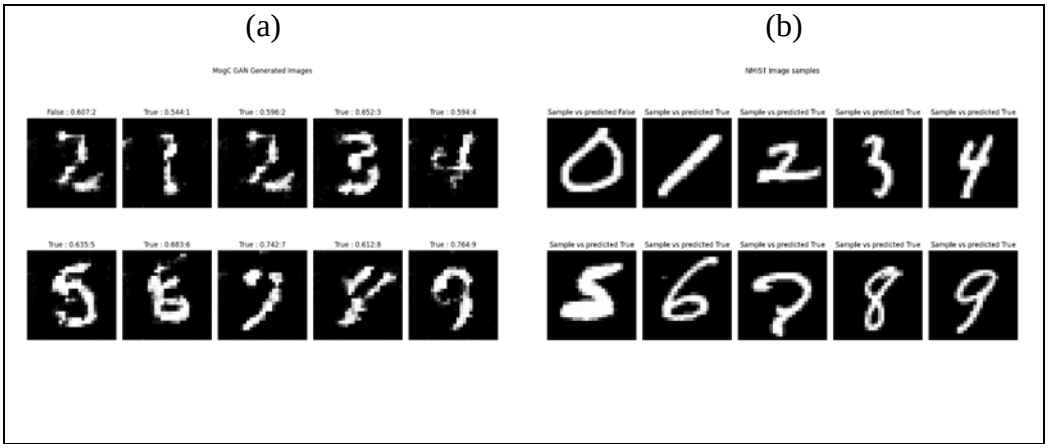


*Figure 20:* ACGAN MogC generated images vs the sampled images from MNIST (ground truth). (a) images generated from a random selection from each class using the test data. The classification vs the ground truth, the confidence score and the predicted class label from the discriminator, (b) Random selected image from corresponind class from the MNIST test data set for comparison.

From Fig 20, the classifier predicted the incorrect class for class 0. In Table 11 whilst the validity score for the for the false prediction is comparable with the correctly predicted class 0, the SSIM and Dice score are low due to the generated image of 2 hiving a high variance when compared with the expected class 0 image.

*Table 11: Evaluation matrix for ACGAN MogC generated images vs the sampled images from MNIST (ground truth).*

| *Class label* | *Prediction* | *Validity score* | *Dice Score* | *SSIM score* |
|---|---|---|---|---|
| 0 | False | 0.607 | 0.155 | 0.064 |
| 1 | True | 0.544 | 0.130 | 0.151 |
| 2 | True | 0.596 | 0.321 | 0.205 |
| 3 | True | 0.652 | 0.473 | 0.323 |
| 4 | True | 0.594 | 0.446 | 0.393 |
| 5 | True | 0.635 | 0.466 | 0.225 |
| 6 | True | 0.682 | 0.421 | 0.195 |
| 7 | True | 0.742 | 0.379 | 0.291 |
| 8 | True | 0.612 | 0.482 | 0.409 |
| 9 | True | 0.763 | 0.296 | 0.255 |

Table 12 below shows the combined average class evaluation metrices. For each class, a corresponding number of MNIST sample are taken for evaluation.

*Table 12:* Evaluation matirx of for all data in each class vs MNIST ground truth samples. The ACGAN MoGC model preforms best using SSIM as the key metric. Wt mean is the weitghed mean taking account of class distribution.

| | | | *MoGM* | | | *MoGC* | | | *NoMoG* | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| *Class* | *Support* | *Acc %* | *DS* | *SSIM* | *Val* | *DS* | *SSIM* | *Val* | *DS* | *SSIM* | *Val* |
| 0 | 181 | 88.4 | 0.47 | 0.23 | 0.57 | 0.37 | 0.20 | 0.66 | 0.52 | 0.18 | 0.58 |
| 1 | 285 | 89.8 | 0.43 | 0.42 | 0.62 | 0.45 | 0.46 | 0.56 | 0.45 | 0.41 | 0.48 |
| 2 | 190 | 87.4 | 0.42 | 0.16 | 0.58 | 0.33 | 0.23 | 0.61 | 0.38 | 0.15 | 0.53 |
| 3 | 244 | 93.9 | 0.44 | 0.22 | 0.63 | 0.53 | 0.35 | 0.65 | 0.42 | 0.24 | 0.52 |
| 4 | 184 | 93.5 | 0.42 | 0.19 | 0.66 | 0.33 | 0.27 | 0.60 | 0.38 | 0.24 | 0.52 |
| 5 | 207 | 91.3 | 0.38 | 0.25 | 0.55 | 0.36 | 0.19 | 0.64 | 0.44 | 0.25 | 0.55 |
| 6 | 224 | 88.8 | 0.33 | 0.21 | 0.50 | 0.45 | 0.26 | 0.68 | 0.49 | 0.27 | 0.49 |
| 7 | 213 | 96.2 | 0.40 | 0.37 | 0.55 | 0.45 | 0.37 | 0.74 | 0.55 | 0.36 | 0.49 |
| 8 | 248 | 94.8 | 0.36 | 0.22 | 0.46 | 0.38 | 0.27 | 0.62 | 0.52 | 0.23 | 0.49 |
| 9 | 213 | 93.0 | 0.46 | 0.31 | 0.54 | 0.46 | 0.36 | 0.75 | 0.48 | 0.35 | 0.44 |
| | 2189 | | | | | | | | | | |
| mean | | 91.71 | 0.411 | 0.258 | 0.566 | 0.411 | 0.296 | 0.651 | 0.463 | 0.268 | 0.509 |
| wt. mean | | 91.8 | | | | | | | | | |

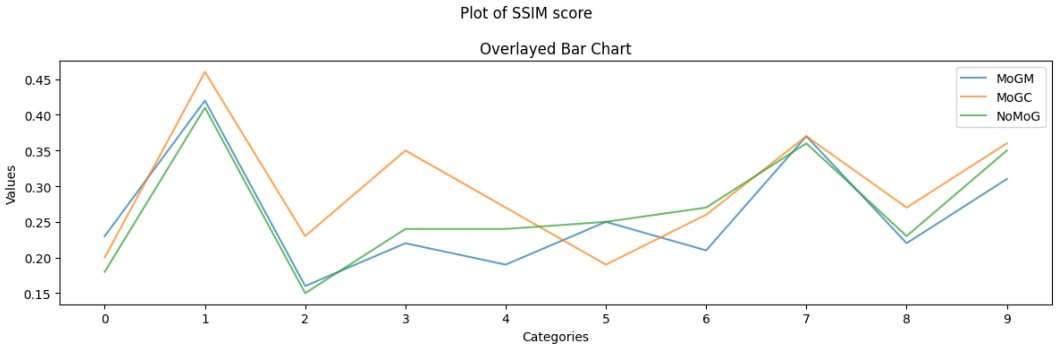


*Figure 21:* plot of averaged SSIM for all test data per class.

**Chapter 6**

# Conclusion

The results of this study support the hypothesis that an image can be generated from an EEG signal by leveraging the latent space vector and a predicted guiding code. When comparing the SSIM scores of the MNIST dataset and the ACGAN MoGC model, I found no significant difference to reject this hypothesis (p-value = 0.779). This aligns with the research objectives outlined in Section 1. The study findings demonstrate that raw EEG data can be effectively processed to extract meaningful features, which, when fed into a trained classification encoder and a subsequent generative network, can generate representative images of the perceived visual stimulus.

In this study, I have explored various processing and classification techniques commonly used in related fields to analyse EEG signals. However, it is important to note that EEG data collection can be prone to noise and variability. Careful handling of this stage is crucial to avoid introducing additional confounding factors that could hinder classification accuracy. Since data collection was not part of this study, I relied on the existing data collection methods used in the public MBD dataset.

The post processing of the data in the study indicates that there is potential for a lot of data to be discarded which may ultimately be useful in the process. The reduction of data from the initial raw set to the final data used for training and testing is from an initial ~5000 samples to ~200 samples, which is only 4% of the total data set. However, given the time scale for the study it was only possible to extract EEG signal data which were relatively easy to classify. This is a short coming in this study and could well form the basis for further work in this area.

While the research literature and expert discussions have provided valuable insights, a crucial question remains unanswered: Do the EEG signals in this study accurately reflect visual stimulus activity, or could they be influenced by subject-specific micro-expressions or gestures? Given the single-subject nature of this study, it is unclear whether the results would generalize to a larger population. While this is beyond the scope of the current study, it could be a fruitful area for future investigation.

Taking the data available and the hypothesis that sufficient features are encoded in the EEG signals recorded. This study uses a multi model approach in two stages, classification, and generation. The first conclusion from this study is the importance of the classification result. Whilst ~92% accuracy achieved from the classification model is a reasonable result; it is less than that of the top-ranking accuracy achieved on the classification of the MNIST data set for example where best of class models can attain ~99% accuracy. Despite the ACGAN model's ability to generate conditioned images, it is unable to rectify misclassifications based solely on the encoded EEG latent vector and conditioning label. As a result, when the model generates an image that is incorrectly classified, the resulting image exhibits a low SSIM score, indicating poor similarity to the ground truth. However, paradoxically, this misclassified image may still receive a relatively high validity score, suggesting a potential discrepancy between the model's perception of the image and its actual fidelity to the ground truth. A weakness in the study is the combination of the conditioning label data and the EEG latent vector. A solution could be improved classification and class separation in the classification model, along with deeper insight into the combination of the condition data and the encoded vector. Upon analysing the average SSIM scores for each class, the study achieved approximately 0.3, which represents a significant level of similarity given that -1 indicates no similarity and 1 indicates a perfect match. This corresponds to roughly 65% similarity, surpassing chance-level performance. Notably, the ACGAN SSIM score is comparable to that of the baseline GAN model trained solely on MNIST data. When examining the MNIST

dataset, it is evident that there exists a degree of variation among images within each class. Consequently, it is unrealistic to expect the model to consistently produce scores closer to the maximum, even with a near-perfect classification model.

This variance per class in the MNIST data also leads to a degree of variance in the final predicted image. The ACGAN does a reasonable job of generating a representative image based on the training time allowed in this study. Earlier difficulties with training GAN networks have been identified and changes in the design of the ACGAN introduced to mitigate these difficulties, such as the introduction of the modulation layer and embedding prove to be beneficial. The finally selected ACGAN with Modulation and concatenation performs the best from all three options. This study has also validated that the random noise vector is not required and can be replaced directly with the encoded vector.

A further extension to this study would include using a reinforcement learning agent applied to the class label prediction. This agent would penalise the incorrect class label. In the current design the weakness is that the predicted class conditioning label has a higher influence on the predicted image over the encoded latent vector. Which in cases where the classification is incorrect the agent would provide unsupervised learning adjustments which may help to correct the ACGAN prediction and validity to indicate an uncertain score.

# References


1. Abbasian, M., Rajabzadeh, T., Moradipari, A., Aqajari, S.A.H., Lu, H. and Rahmani, A. (2023) Controlling the Latent Space of GANs through Reinforcement Learning: A Case Study on Task-based Image-to-Image Translation. https://arxiv.org/abs/2307.13978

2. Adam, A., Ibrahim, Z., Mokhtar, N., Shapiai, M., Cumming, P., Mubin, M. (2016) Evaluation of different time domain peak models using extreme learning machine-based peak detection for EEG signal. *Springer Plus* , 5(1):1036. Springer Science and Business Media LLC.

3. Aggarwal, A., Mittal, M. Battineni, G. (2021) Generative adversarial network: An overview of theory and applications. *International Journal of Information Management Data Insights*, 1(1): 100004. Elsevier. http://doi.org/10.1016/j.jjimei.2020.100004

4. Badcock, N., Mousikou, P., M, Y., De lissa, P., Thie, J., Mcarthur, G. (2013) Validation of the Emotiv EPOC EEG system for measure research quality auditory ERPs. *Psychiatry and Psychology, PeerJ.* 1(38). https://doi.org/10.7717/peerj.38

5. Chaddad, A., Wu, Y., Kateb, R., Bouridane, A. (2023) Electroencephalography Signal Processing: A Comprehensive Review and Analysis of Methods and Techniques. *MDPI, Sensors* 23(14):6434. https://doi.org/10.3390/s23146434

6. Chollet, François / others. (2015) Keras. https://keras.io (accessed 2024)

7. Falciglia, S., Betello, F., Russo, S., Napoli, C. (2024) Learning visual stimulus-evoked EEG manifold for neural image classification. *Neurocomputing.* Elsevier BV 588:127654

8. Fulcher, B.D., Little, M.A. and Jones, N.S. (2013) Highly comparative time-series analysis: the empirical structure of time series and their methods. *Journal of the Royal Society Interface*. The Royal Society. 10(83), 20130048. https://royalsocietypublishing.org/doi/10.1098/rsif.2013.0048

9. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y. (2014) Generative Adversarial Nets, *Information Processing Systems*. 27

10. Gramfort, A., Luessi, M., / Larson, E., Engemann, D., Strohmeier, D., Brodbeck, C., Goj, R., Jas, M., Brooks, T., Parkkonen, L., Hämäläinen, M. (2013) MEG and EEG data analysis with MNE-Python. *Frontiers in Neuroscience*. Frontiers Media SA. 7(267):1-13

11. Gurumurthy, S., Sarvadevabhatla, R., Babu, R. (2017) DeLiGAN: Generative Adversarial Networks for Diverse and Limited Data. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 166-174. https://doi.org/10.1109/cvpr.2017.525

12. Homan, R W., Herman, J., Purdy, P. (1987) Cerebral location of international 10-20 system electrode placement. *Electroencephalography and Clinical Neurophysiology.* 66(4):376-382. https://doi.org/10.1016/0013-4694(87)90206-9

13. Jasper, H. (1957) To standardise the method of EEG placement. *Brussels IV International EEG congress*.

14. Koide-Majima, N., Nishimoto, S., Majima, K. (2024) Mental image reconstruction from human brain activity: Neural decoding of mental imagery via deep neural network-based Bayesian estimation. *Neural Networks*, Elsevier. 170:349-363

15. Kumar, P., Saini, R., Roy, P., Sahu, P., Dogra, D. (2018) Envisioned speech recognition using EEG sensors. *Personal and Ubiquitous Computing*. 22(1):185-199. Springer Science and Business Media LLC. http://doi.org/10.1007/s00779-017-1083-4

16. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P. (1998) Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, Institute of Electrical and Electronics Engineers (IEEE). 86(11):2278-2324

17. LeCun, Y., Cortes, C., Burges C J., (1998) THE MNIST DATABASE of handwritten digits. http://yann.lecun.com/exdb/mnist/. (accessed 2024)

18. Lubba, C., Sethi, S., Knaute, P., Schultz, S., Fulcher, B. and Jones, N. (2019) catch22: Canonical time-series characteristics. *Data Mining and Knowledge Discovery*. Springer Science and Business Media LLC, 33(6), 1821-1852

19. Ludwig KA, Miriani RM, Langhals NB, Joseph MD, Anderson DJ, Kipke DR. (2009) Using a common average reference to improve cortical neuron recordings from microelectrode arrays. *Journal of Neurophysiology*. American Physiology Society 101(3):1679-1689. https://doi.org/10.1152/jn.90989.2008

20. Mishra, R., Sharma, K., Bhavsar, A. (2021) *Visual Brain Decoding for Short Duration EEG Signals*. 29th European Signal Processing Conference (EUSIPCO). IEEE. 1226-1230. https://doi.org/10.23919/eusipco54536.2021.9616192

21. Mishra, R., Sharma, K., Jha, R. R. and Bhavsar, A. (2023) NeuroGAN: image reconstruction from EEG signals via an attention-based GAN. *Neural Computing and Applications.* 35(12), 9181-9192. https://doi.org/10.1007/s00521-022-08178-1

22. Odena, A., Olah, C., Shlens, J. (2017) Conditional Image Synthesis with Auxiliary Classifier GANs. *Proceedings of the 34th International Conference on Machine Learning*. 70:2642-2651

23. Prerau, M., Brown, R., Bianchi, M., Ellenbogen, J., Purdon, P. (2017) Sleep Neurophysiological Dynamics Through the Lens of Multi-taper Spectral Analysis. *Physiology, American Physiological Society*. 32(1):60-92

24. Radford, A., Metz, L. Chintala, S. (2016) Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *arXiv preprint*. https://doi.org/10.48550/arXiv.1511.06434

25. Radüntz, T. (2018) Signal Quality Evaluation of Emerging EEG Devices. *Frontiers in Physiology* , Vol. 9. https://doi.org/10.3389/fphys.2018.00098

26. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M. (2015) Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252.

27. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X. (2016) Improved techniques for training gans. *Advances in neural information processing systems*. 29.

28. Sanei, S., Chambers, J.A., (2013). EEG signal processing. *John Wiley & Sons*.

29. Seeliger, K., Güçlü, U., Ambrogioni, L., Güçlütürk, Y., Van G., Marcel A. J. (2018) Generative adversarial networks for reconstructing natural images from brain activity. *NeuroImage,* Elsevier. 181:775-785

30. Shen, G., Horikawa, T., Majima, K., Kamitani, Y. (2019) Deep image reconstruction from human brain activity. *PLoS computational biology*. Cold Spring Harbor Laboratory. 15(1):e1006633. https://doi.org/10.1101/240317

31. Spampinato, C., Palazzo, S., Kavasidis, I., Giordano, D., Shah, M. and Souly, N. (2019) Deep Learning Human Mind for Automated Visual Classification. *arXiv preprint*. https://arxiv.org/abs/1609.00344

32. Tibdewal, M N., Mahadevappa, M., Ray, A K., / Malokar, M., Dey, H R. (2016) Power line and ocular artifact denoising from EEG using notch filter and wavelet transform. *3rd International Conference on Computing for Sustainable Global Development* INDIACom 1654-1659

33. Tirupattur, P., Rawat, Y. S., Spampinato, C. and Shah, Mubarak. (2018) ThoughtViz: Visualizing Human Thoughts Using Generative Adversarial Network. *Proceedings of the 26th ACM International Conference on Multimedia*. 950–958. https://doi.org/10.1145/3240508.3240641

34. Vallat, R. and Walker, M. P. (2021) An open-source, high-performance tool for automated sleep staging. *eLife Sciences Publications, Ltd*. 10,e70092. https://doi.org/10.7554/eLife.70092

35. Vivancos, D. and Cuesta, F. (2022) MindBigData 2022 A Large Dataset of Brain Signals. https://arxiv.org/pdf/2212.14746

36. Zhang, X., Ma, Z., Zheng, H., Li, T., Chen, K., Wang, X., Liu, C., Xu, L., Wu, X., Lin, D. and Lin, H. (2020) The combination of brain-computer interfaces and artificial intelligence: applications and challenges. *Annals of Translational Medicine*. 8(11)

# Appendix

Python packages installed in environment, python 3.8, MSC v.1929 64 bit.

ale-py==0.8.1
atari-py==0.2.9
AutoROM==0.4.2
AutoROM.accept-rom-license==0.6.0
cvlib==0.2.7
deepface==0.0.79
einops==0.6.0
entrypoints==0.4
ffmpeg-python==0.2.0
fqdn==1.5.1
gensim==4.3.1
graphviz==0.20.1
gym-notices==0.0.8
gym-super-mario-bros==7.4.0
ipympl==0.9.3
isoduration==20.11.0
jsonpointer==2.3
keras-contrib==2.0.8
keras-tuner==1.3.5
mediapipe==0.10.8
mlxtend==0.22.0
pillow-heif==0.14.0
pip==23.0.1
pipdeptree==2.6.0
pycatch22==0.4.5
pydot==1.4.2
pymc==5.6.1
PySocks==1.7.1
python-docx==0.8.11
python-version==0.0.2
qqdm==0.0.7
scikit-image==0.21.0
sentence-transformers==2.2.2
shapely==2.0.2
ssqueezepy==0.6.4
statsmodels==0.14.1
tensorflow-gpu==2.10.0
tensorflow-intel==2.10.1
tf-models-official==2.10.0
torchaudio==2.0.1+cu118
torchsummary==1.5.1
uri-template==1.2.0
webcolors==1.13
wget==3.2
yasa==0.6.5