# Data Augmentation for multichannel BCI-EEG using GAN network

## Introduction

Brain-Computer Interface (BCI) techniques aim to use Electroencephalography (EEG) recordings from a subject to interact with an external device using only the brain signals. In recent years, the use of neural networks (NN) to extract features and process the EEG measurements has been increasing in popularity and has outperformed other methods in many applications. Nevertheless, one of the main drawbacks of NN is the vast amount of data needed for the training stage. This comes from the fact that NN has to learn to extract and to use the useful information from nothing else than the input data. Moreover, it has to generalize the concepts to be sure it could be applied to different scenarios besides the ones seen during the training phase. When it comes to obtaining more EEG samples from subjects this implies that the subject needs to be prepared, taught how to conduct the experiment, and finally do it. This normally can take several hours, making the acquisition time-consuming and therefore expensive.

Many authors suggest that having a larger dataset would most likely improve their results [1]. A possible reason for this is that the current lack of large datasets prevents them from using deeper architectures as they do in other fields. Data augmentation has been widely applied to overcome the shortage of data. The idea is to use available data and apply some transformations that simulate different realistic variations to create new samples. In the field of EEG recording, data augmentation has been done by adding random noise, eye blink artifacts, muscle activity, overlapping windows, different downsampling, between many others [1]. Since 2018, a few publications have come out using Generative Adversarial Networks (GANs) to try to augment the data. The idea behind GANs is to train two networks iteratively interacting with each other, one to produce fake data and the other to be able to distinguish real data from fake data. It can be seen as a minimax game in which two players take turns and choose the action to maximize their gain knowing the other player will try to minimize their loss.

The goal of this work is to probe that data augmentation of EEG signals using GAN networks can be done, and yield good results in a classification task.

## Deeper into GANs

GANs were proposed in 2014 producing groundbreaking results in the computer vision field but the training process was quite unstable. In the upcoming years, many modifications were introduced allowing the models to produce high-resolution fake-images that could deceive the human eye. Nevertheless, it is still complicated to train this type of architecture as the network in charge of detecting fake data (Discriminator) can collapse [2] into only detecting few and narrow modes of the input distribution as reals. The immediate consequence is that the other network (Generator) is not forced to generate a variety of fake data. One of the most promising modifications of GANs was the Wasserstein-GAN (W-GAN), introduced in 2017 [3], where the authors proposed changing the loss function. Originally, the loss was minimizing the Jensen-Shannon divergence between real and fake data distribution. But, this causes the gradient to vanish in the generator [3], so the W-GAN instead maximizes the Wasserstein distance of the distributions guaranteeing in this way to have a gradient at every point as far as the distribution is K-Lipschitz. To ensure the last requisite, the authors proposed both clipping the data or using a gradient penalty term, having each different advantages and disadvantages. In 2018, Hartmann et al. [4] proposed to use a dynamic gradient penalty term (see Eq. 1). The reason behind this is that the penalty plays a role in the trade-off between enforcing the Lipschitz continuity and overcoming the distance term, and the optimal value varies with the distance.

$$L_c = \underbrace{-\tilde{W}(\mathbb{P}_r, \mathbb{P}_\theta) +}_{\text{Wasserstein distance}} \underbrace{\max(0, \tilde{W}(\mathbb{P}_r, \mathbb{P}_\theta))}_{\text{Dynamic weighting}} \underbrace{\lambda \cdot E_{\hat{x} \sim \mathbb{P}_{\hat{x}}}[\max(0, ||\nabla_{\hat{x}} D(\hat{x})||_2 - 1)^2]}_{\text{Original gradient penalty term}})$$

Equation 1: GAN modified critic (discriminator) loss.

Another variation implemented here to improve the quality of fake samples is to train the network progressively [5]. This consists of increasing the dimension of the fake samples by steps until reaching the desired resolution.

## Previous work

As mentioned before, there have been publications on using GANs on EEG data. In the following table, there is a list of those works, with details on what they introduce to the fields, which type of data they used, and how they test their methods.

| Reference | Novelty | Data | Test |
|---|---|---|---|
| [4] | Improvement to WGAN-GP for EEG | Motor task (hand movement) | Visual inspection |
| [6] | Recurrent GAN for EEG augmentation | Motor movement<br>Imagery events | Classification |
| [7] | WGAN-GP, CC-GAN, multichannel | RSVP Dataset | Classification<br>ERP and P300 |
| [8] | Test GAN on EEG | Cue-based visual attention task | Visual inspection |
| [9] | GAN for SSVEP | SSVEP with dry-EEG | Classification |

Table 1: summary of the literature on augmenting EEG data using GAN.

All of the authors claim that GANs is a promising method to effectively reproduce EEG signals. Specifically, the three authors who tested the generated data on a classification task proved that the data is effectively representing meaningful features of the EEG that are used by other networks. The most comprehensive work on the effectiveness of GANs was done by [7] and [4], where the former used a class-condition GAN to generate different types of multi-channel signals and used them as augmented data for a classifier, and the latter analyzed and proposed improvements on the GAN's loss function. Both of them dive into the characteristics of the generated EEG and how those are affected by different network parameters such as convolutional filter size and upsampling and downsampling methods. Also, they discuss and propose different metrics to evaluate the quality of the generated samples, where basically they conclude that there is no unique metric to correctly describe all the aspects of the fake signals.

# Materials and Methods

## Dataset description

A public dataset collected by the BCI group at Colorado State University was used in this study [10]. It contains EEG data recorded at 256Hz with an 8-channel electrode system (g.GAMMAsys) from 9 healthy participants who were asked to perform a visual stimulus protocol designed to elicit P300 waves. It consisted of looking at a computer screen where nine letters were randomly displayed sequentially in a three-by-three table. Subjects were asked to count the number of times a given letter appeared in the center of the table.

The dataset was partitioned into two different datasets. One containing 750 ms around the targeted letter stimulus, and the other the same with the non-targeted letter. Afterward, the signals were filtered using a band-pass filter between 0.01 Hz and 50 Hz. As the signals were sampled at 256 Hz, the 750 ms segments contain 192 samples.

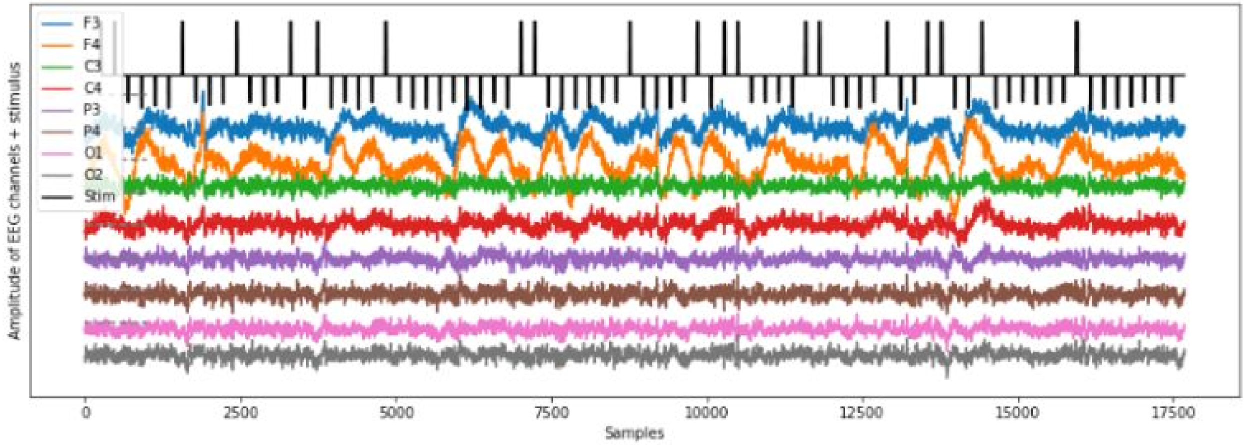

Figure 1: original dataset from one subject. In black, the stimulus signal, where the positive spikes indicate when the target letter was flashed.

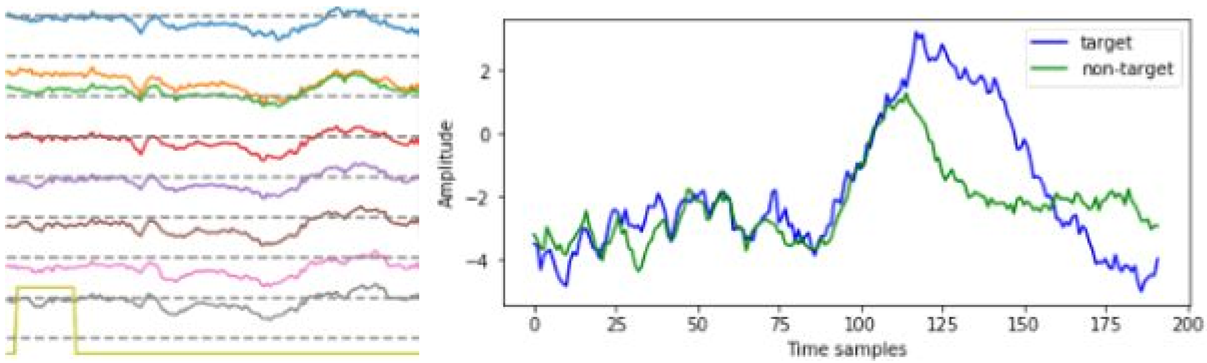

Figure 2: on the left the average over all the positive stimulus segments, showing 700 ms after a stimulus (in yellow). On the right, and average overall samples and channels after splitting the dataset into target letter (segments after a positive flank in the stimulus signal) and non-target letter (segments after a negative flank in the stimulus signal).

The training and validation set contains recordings from 5 subjects split 80% for training and 20% for validation. The testing set contains recordings from 4 other subjects. The same sets are used for both training the GAN and Classifier models, meaning that the test set is not seen either by the classifier or the GAN during any stage of the training.

| **Dataset** | **Real non-target letter** | **Real target letter** | **Fake non-target letter** | **Fake target letter** |
|---|---|---|---|---|
| Training/Validation | 900 x 192 x 8 | 300 x 192 x 8 | 1000 x 192 x 8 | 1000 x 192 x 8 |
| Testing | 720 x 192 x 8 | 240 x 192 x 8 | - | - |

Table 2: dimensions of each dataset (samples x time x electrodes)

# GAN model

The GAN model was heavily based on the implementation of [4] with some modifications to work with the dimensions of the dataset. It required to decrease the depth of the network by reducing the number of progressive blocks to four instead of six and to add an extra dimension to handle the 8-channels instead of the single-channel the authors used in their paper. Also, several bugs had to be corrected and many implementation details were modified to optimize the performance and parametrization.

| *Generator* | | |
|---|---|---|
| **Layer** | **Activation / Normalization** | **Output shape** |
| Input | | 200 x 1 |
| Fully Connected | WN / LreLu | 50 x 12 |
| Cubic Upsample | | 50 x 24 |
| Conv2D | WN / LreLu / PN | 50 x 24 |
| Conv2D | WN / LreLu / PN | 50 x 24 |
| Cubic Upsample | | 50 x 48 |
| Conv2D | WN / LreLu / PN | 50 x 48 |
| Conv2D | WN / LreLu / PN | 50 x 48 |
| Cubic Upsample | | 50 x 96 |
| Conv2D | WN / LreLu / PN | 50 x 96 |
| Conv2D | WN / LreLu / PN | 50 x 96 |
| Cubic Upsample | | 50 x 192 |
| Conv2D | WN / LreLu / PN | 50 x 192 |
| Conv2D | WN / LreLu / PN | 50 x 192 |
| Conv2D | WN | 1 x 192 |

| *Discriminator* | | |
|---|---|---|
| **Layer** | **Activation / Normalization** | **Output shape** |
| Input | | 1 x 192 |
| Conv2D | WN / LreLu | 50 x 192 |
| Conv2D | WN / LreLu | 50 x 192 |
| Conv2D | WN / LreLu | 50 x 192 |
| Conv2D Downsample | WN / LreLu | 50 x 96 |
| Conv2D | WN / LreLu | 50 x 96 |
| Conv2D | WN / LreLu | 50 x 96 |
| Conv2D Downsample | WN / LreLu | 50 x 48 |
| Conv2D | WN / LreLu | 50 x 48 |
| Conv2D | WN / LreLu | 50 x 48 |
| Conv2D Downsample | WN / LreLu | 50 x 24 |
| Conv2D | WN / LreLu | 50 x 24 |
| Conv2D | WN / LreLu | 50 x 24 |
| Conv2D Downsample | WN / LreLu | 50 x 12 |
| Fully Connected | WN | 1 x 192 |

Table 3: architecture of GAN's generator and discriminator. WN:Weight Normalization, LRelu:Leaky Rectifier Linear Unit, PN: Pixel Normalization. For more details on the implementation refer to original implementation [4].

# Classifier model

The classifier is an implementation of the EEGNet [11] based on this implementation [12] with the necessary modifications to run with these datasets. No effort was done on improving the classifier as the only goal was to compare the performance based on using or not an augmented dataset.

| Layer | Output shape |
|---|---|
| Input | 1 x 8 x 192 |
| Conv2D | 8 x 8 x 192 |
| Batch Normalization | 8 x 8 x 192 |
| Depthwise Conv2D | 16 x 1 x 192 |
| Batch Normalization | 16 x 1 x 192 |
| Exponential Linear Unit Activation | 16 x 1 x 192 |
| Average Pooling | 16 x 1 x 48 |
| Dropout | 16 x 1 x 48 |
| Separable Conv2D | 16 x 1 x 48 |
| Batch Normalization | 16 x 1 x 48 |
| Exponential Linear Unit Activation | 16 x 1 x 48 |
| Average Pooling | 16 x 1 x 6 |
| Dropout | 16 x 1 x 6 |
| Flatten | 96 |
| Fully Connected | 2 |
| SoftMax Activation | 2 |

Table 4: architecture of EEGNet. For more details on the implementation refer to original implementation [12].

# Experiments

## Generating fake signal of target letter and non-targeted letter

The first part of the experiments done in this work consist of training two different GANs. One, to generate P300 waves like the ones produced by the brain when the expected letter appeared on the screen, and the other to generate the brain signal produced when another letter different from the expected was showing up. The latter is also a P300 wave as a stimulus occurs but with lower energy than in the former case.

To evaluate the generated samples, both a comparison in time and frequency is done between real and fake samples.

It is important to remark that a reduced amount of samples was used to train the GAN, as the final goal is to use that reduced dataset to train a classifier knowing that it could be further improved if more samples were available.

## Comparing classification results with and without data augmentation

Once the GANs are trained, they are used to generate fake samples and augment the original datasets. With these two datasets, the original and the expanded, containing both target and non-target letter stimulus a classifier is trained on both separately and the accuracy is compared.

For the no-augmentation case, the training set is balanced by leaving out non-target letter samples, so there are 240 samples of each in the training and 60 in the validation. For the augmentation case, the training set is the one used previously with the addition of 240 samples of each type, yielding 480 samples of each type in the training set. The validation remains exactly the same as the no-augmentation case, without any addition of fake samples.

To compare the results a statistical analysis over the validation and test data is done. For this, fifty trainings are computed with random initial states and the confusion matrix of the mean with the standard deviation is shown.

# Results

## Generating fake target letter and non-target letter signals

The first evaluation over the fake data is comparing the average over samples and how each channel resembles their real counterpart.

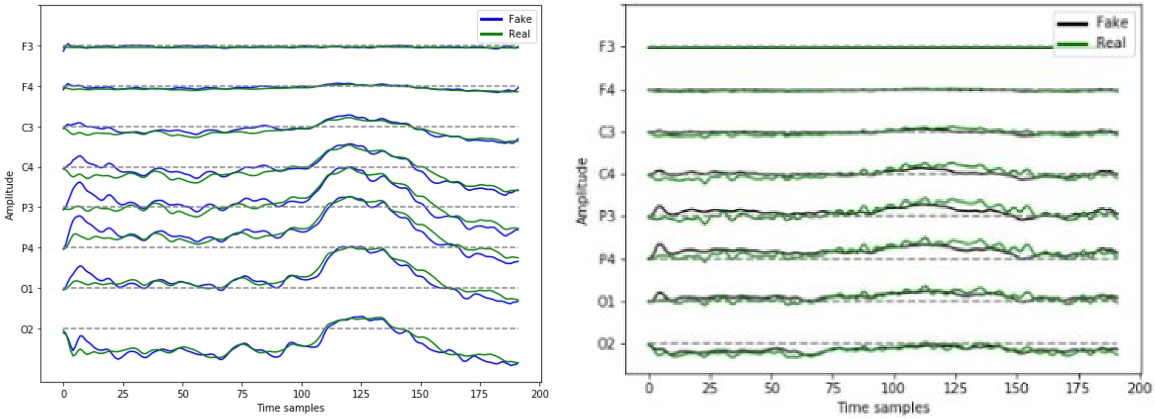

Figure 3: average over samples of real and fake data. P300 waves on the left and non-P300 on the right.

We can observe that the first two and the last two channels are very similar both in shape and magnitude. In the non-P300 case, we can also notice a higher frequency range in the real signals that were not reproduced by the fake signals.

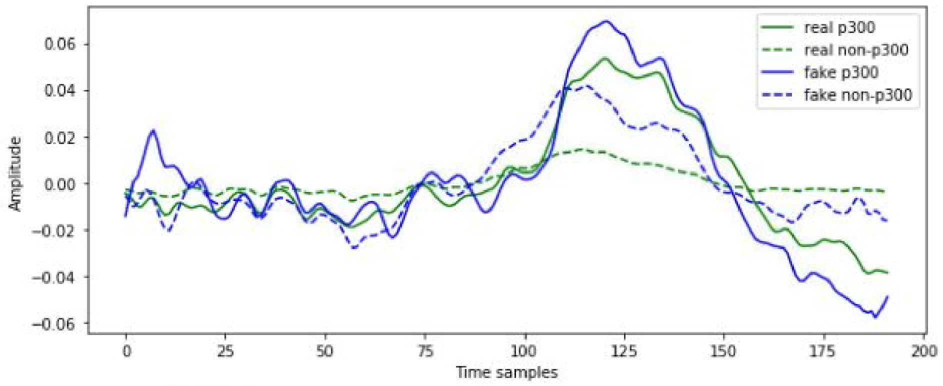

Figure 4: average over samples and channels of both fake and real samples of P300 and non-P300 signals.

As seen in Figure 4, both types of fake signals present higher amplitudes on average than their real counterparts after normalization, but they maintain a relative amplitude difference between them.

To analyze furthermore in a qualitative way the generated signals the frequency spectrum is shown for each channel.

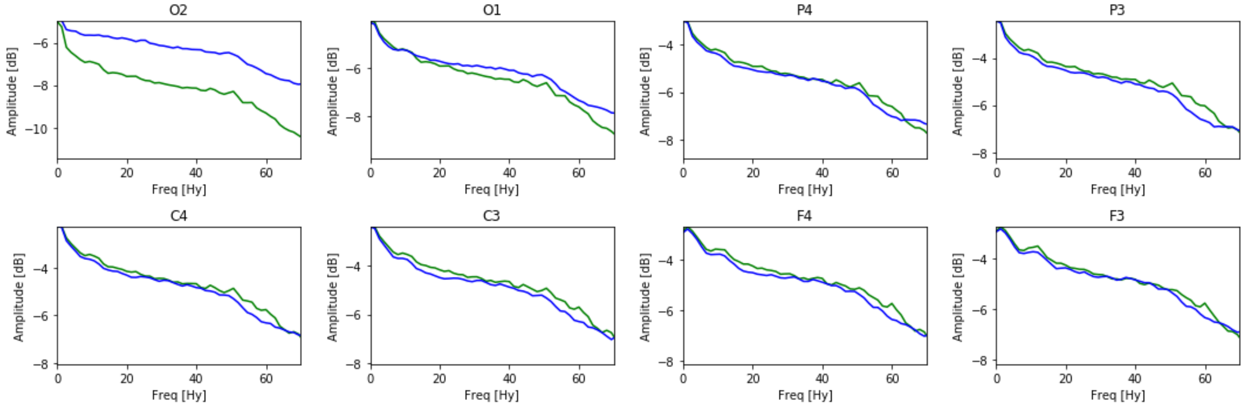

Figure 5: power spectral density of each EEG channel for the P300 signals. In each subplot, it is compared the real (green) and fake (blue) samples.

Except for the first two channels, the rest have a similar power spectrum density. Nevertheless, it is important to remark that electrodes F3 and F4, as seen in Figure 3, do not present a P300 component as the others do. and also their overall amplitudes are significantly lower. Therefore, in theory, it should not interfere with the classification. Overall, at higher frequencies than 60 Hz the spectrums start to be more different and that is because the parameters of the convolutional layers (filter size and strides) are tuned to reproduce lower frequencies better.

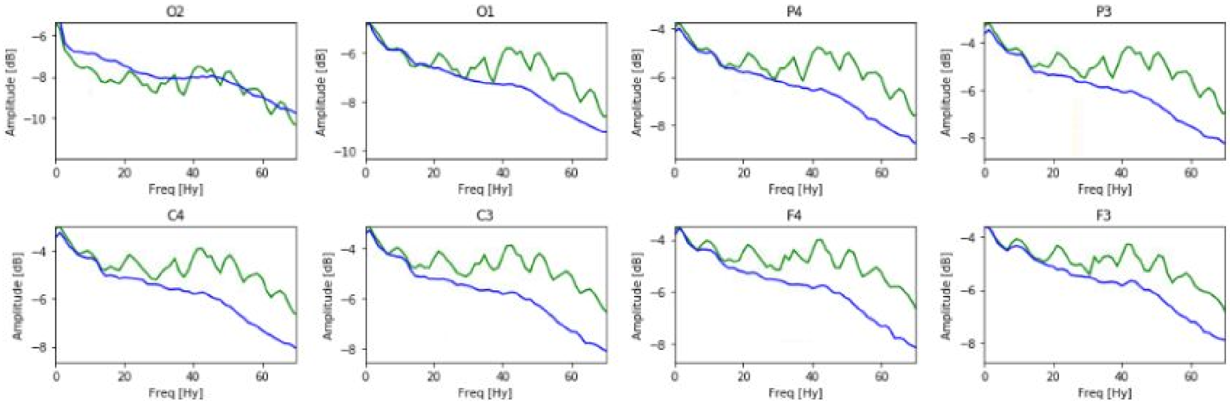

Figure 6: power spectral density of each EEG channel for the non-P300 signals. In each subplot, it is compared the real (green) and fake (blue) samples.

For the non-P300 signals, the generated ones did not match the real samples as it happened with the P300 signals. Again, at lower frequencies, we can observe a better similarity, which was the main priority. A reason that the rest of the spectrum is significantly different might be because in the P300 case, the alpha (8-15 Hz) and the theta (4-7 Hz) waves, which were the targeted frequency range, have more energy leaving higher frequencies in a different order of magnitude and perceptually looking more similar than in the non-P300 case.

In order to understand how the GAN was trained the losses are displayed below. It is still under discussion how these losses should look like and evolve over the iterations, as they normally are really noisy and can lead to good results even if they do not behave as theoretically they should.

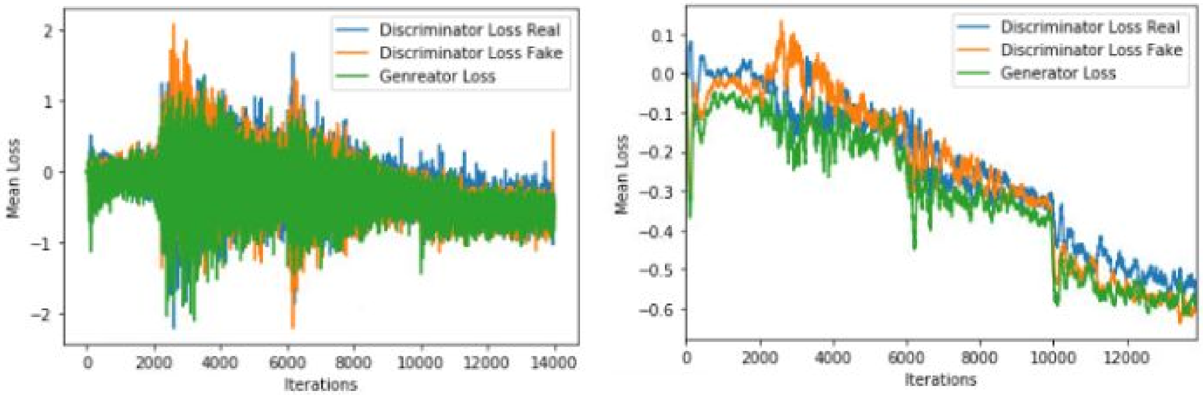

Figure 7: GAN losses: on the left the raw losses and on the right a smoothed version using a moving average window for the P300 case.

We can observe that the losses present a sort of pulses and that is due to the progressive nature of the GAN used. As mentioned before, the GAN starts with a seed vector of low resolution and by steps, it increases the resolution until reaching the desired output dimensions. So, every time the resolution is increased (or decreased, if talking from the Discriminator point of view) there is a major change in the losses. Then, within each of these constant resolution periods, the losses seem to converge to the same value, which theoretically is an equilibrium point in this minimax game the Generator and Discriminator are playing. Finally, the tendency of the losses to increase negatively, most likely is because of the penalty term in the losses that increases as the distance between the fake and real distributions grow apart in the Generator.

## Comparing classification results with and without data augmentation

To validate the effectiveness of the GAN generating fake EEG signals, a classifier was trained both with the original dataset and an augmented dataset. The following figures show the confusion matrices over the validation set and a testing set.

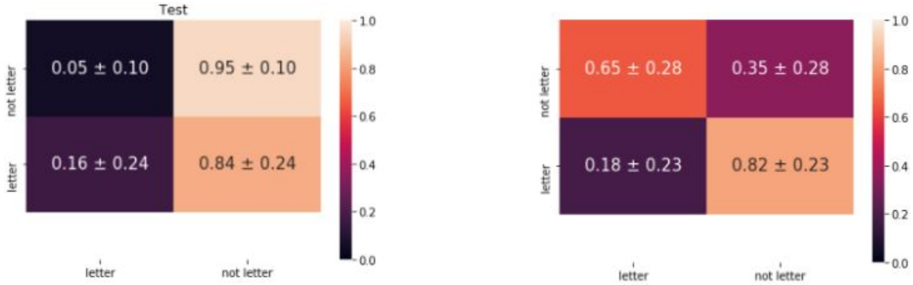

Figure 8: confusion matrix of the classifier using only real samples. On the left the validation set and on the right the test set.

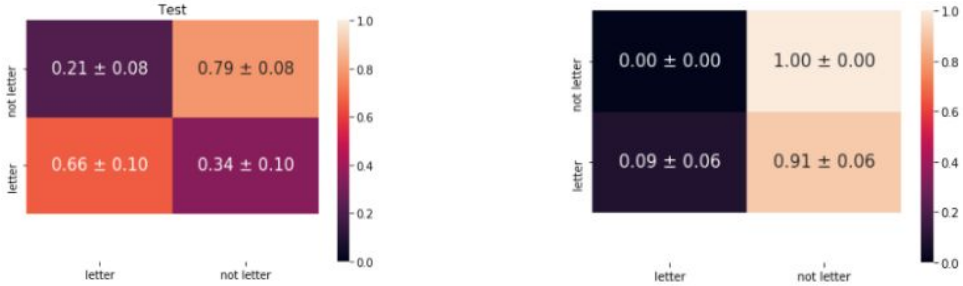

Figure 9: confusion matrix of the classifier using an augmented dataset. On the left the validation set and on the right the test set.

The classification improves on the validation set but not on the test set. To objectively probe that there is an improvement a t-test for each quadrant of the confusion matrices on the validation set was done yielding p-values lower than 0.001. Despite the non-target letters on the non-augmented case is 95%, it is at the expense of classifying also 84% of the target letters as non-target letters. Whereas using fake samples for the training evened the accuracy for both classes.

In both cases, the classifier performed poorly in the testing set. This may suggest that data from a new subject cannot be classified correctly (with this classifier at least) unless maybe a small training is performed on a few samples of the new subject.

# Conclusion and discussion

In this work, a Generative Adversarial Network was designed to generate fake P300 signals from the brain, and those samples were later used to augment a dataset to train a classifier to distinguish the P300 waves from non-P300 brain signals. The GAN successfully imitated the signal in all 8 channels located at different positions on the subjects' head, based both on a qualitative way, assessing the appearance of the signals in the time and frequency domain, and in a quantitative way, improving the accuracy of a classifier when the fake samples were used to augment the training set.

The Progressive GP-WGAN with the modified dynamic penalty term was stable on every training. Nevertheless, one of its major limitations is that the architecture needs to be tuned for different types of signals. The network used here, for example, works for a low-frequency EEG signal but not for an audio signal containing much higher frequencies.

As a possible future research path, it could be contrasted the improvement on a classifier using augmented data produced by GANs against other augmentation techniques.

# Bibliography

*[1] Roy, Y., Banville, H., Albuquerque, I., Gramfort, A., Falk, T. H., & Faubert, J. (2019). Deep learning-based electroencephalography analysis: A systematic review. Journal of Neural Engineering, 16(5). https://doi.org/10.1088/1741-2552/ab260c*

*[2] Kodali, N., Abernethy, J., Hays, J., & Kira, Z. (2017). On Convergence and Stability of GANs. http://arxiv.org/abs/1705.07215*

*[3] Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein GAN. http://arxiv.org/abs/1701.07875*

*[4] Hartmann, K. G., Schirrmeister, R. T., & Ball, T. (2018). EEG-GAN: Generative adversarial networks for electroencephalograhic (EEG) brain signals. http://arxiv.org/abs/1806.01875*

*[5] Karras, T., Aila, T., Laine, S., & Lehtinen, J. (2018, October 27). Progressive growing of GANs for improved quality, stability, and variation. 6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings.*

*[6] Abdelfattah, S. M., Abdelrahman, G. M., & Wang, M. (2018). Augmenting the Size of EEG datasets Using Generative Adversarial Networks. Proceedings of the International Joint Conference on Neural Networks, 2018-July(January), 1–6. https://doi.org/10.1109/IJCNN.2018.8489727*

[7] *Panwar, S., Rad, P., Jung, T.-P., & Huang, Y. (2019). Modeling EEG data distribution with a Wasserstein Generative Adversarial Network to predict RSVP Events. http://arxiv.org/abs/1911.04379*

[8] *Fahimi, F., Zhang, Z., Goh, W. B., Ang, K. K., & Guan, C. (2019). Towards EEG generation using gans for bci applications. 2019 IEEE EMBS International Conference on Biomedical and Health Informatics, BHI 2019 - Proceedings. https://doi.org/10.1109/BHI.2019.8834503*

[9] *Nik Aznan, N. K., Atapour-Abarghouei, A., Bonner, S., Connolly, J. D., Al Moubayed, N., & Breckon, T. P. (2019). Simulating Brain Signals: Creating Synthetic EEG Data via Neural-Based Generative Models for Improved SSVEP Classification. Proceedings of the International Joint Conference on Neural Networks, 2019-July. https://doi.org/10.1109/IJCNN.2019.8852227*

[10] https://www.cs.colostate.edu/eeg/main/data/2011-12_BCI_at_CSU

[11] *Lawhern, V. J., Solon, A. J., Waytowich, N. R., Gordon, S. M., Hung, C. P., & Lance, B. J. (2016). EEGNet: A Compact Convolutional Network for EEG-based Brain-Computer Interfaces. https://doi.org/10.1088/1741-2552/aace8c*

[12] github.com/vlawhern/arl-eegmodels

# Glossary

SSVEP: Steady-State Visual Evoked Potential

RSVP: Rapid Serial Visual Presentation

ERP: Event-related Potential

BCI: Brain-Computer Interface

EEG: Electroencephalography

NN: Neural Networks

GAN: Generative Adversarial Network

CNN: Convolutional Neural Network