BrainGAN

Generating Synthetic EEG Data

Chase Bosworth, Thomas Klimek, Dan Berman

1. Abstract

This paper describes the procedure behind developing a generative model for EEG data called BrainGAN. In the growing field of research for brain computer interfaces, EEG data is often measured and classified with various machine learning techniques in order to creative adaptive interfaces based around brain signals. One common issue is lack of data, as each classifier must be trained to a unique brain signal, and participants can only be screened for limited amounts of time. Researchers are often forced to work with small sets of time series data to develop new interfaces. This paper explores using a generative adversarial network (GAN) to create synthetic EEG data to be used for machine learning based EEG classification tasks. We propose an architecture for preprocessing EEG data and building a GAN. We then compare the results of a classification task, given a model trained on the original dataset and a dataset augmented with synthetic data from our GAN. Our experiments seek to increase classifier accuracy for motor imagery interaction tasks such as left hand v.s. right hand classification, as well as generate realistic EEG data to be used for further development of implicit brain computer interfaces.

2. Related Work

The application of GAN to EEG data is not unprecedented. Hartmann et al explored several GAN architectures in EEG-GAN: *Generative adversarial networks for electroencephalographic brain signals*. The GAN framework was first proposed in Goodfellow et al's *Generative Adversarial Networks*. Until recently, large, high quality datasets of EEG data related to motor imagery interaction tasks could be hard to come by, but as recent advances in machine learning have made the idea of brain computer interfaces seem more attainable, this area has become a growing area of research. In [4], Mishchenko, Yuriy, et al. demonstrated classification accuracies of over 90% for some subjects on a two state left hand - right hand motor imagery task. In 2018, in order to propel research in this area, Kaya, Murat, et al. published a dataset with over 60 hours of EEG data gathered from 13 participants completing BCI related tasks [5].

3. Data & Preprocessing

Our data comes from a large electroencephalographic motor imagery dataset for electroencephalographic brain computer interfaces [5]. From this dataset we will focus on the the motor imagery interaction paradigm, a task where users can be in two states: imagining movement with the left hand, or imagining movement with the right hand. This is a common paradigm in BCI research, and the subject's imagined action has been shown to be discernible through EEG data [4]. Our dataset contains 13 participants each having a total, across both conditions, of around 700 measurements of them completing this task. In our research, we focused on measurements coming from a single subject. Because this paper serves as a preliminary step and proof of concept of the use of GAN for EEG dataset augmentation, we decided to use only one of the 22 EEG channels available in our dataset. We analyzed measurements only from C3, which, located above the subject's motor cortex, has been shown to be the most salient to this task. A single instance in our data set comes from a one trial of the imagery task. The instance consists of 90 features, each a time series measurement of EEG data for the two tasks. Each datapoint is an ERP

value generally falling in the range of [-10, 10] corresponding to the EEG measurement at the given time stamp. Plotting our data, we can see in some cases left and right hand motor cognition task are visually discenerable such as figures [1] and [2]. However in other cases this difference is more nuanced such as figure [3], and figure [4]. In figure [5], which shows the mean of all ERP measurements collected from a given subject, we see that on average the two imagery tasks create distinct pattern.

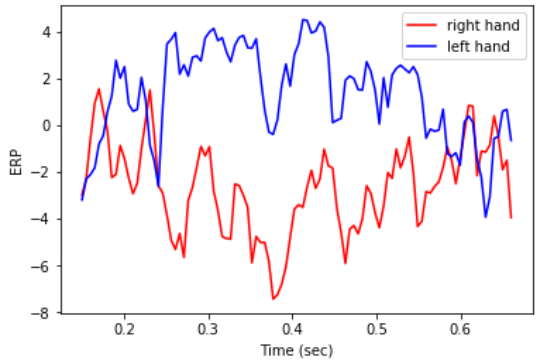

Figure [1]

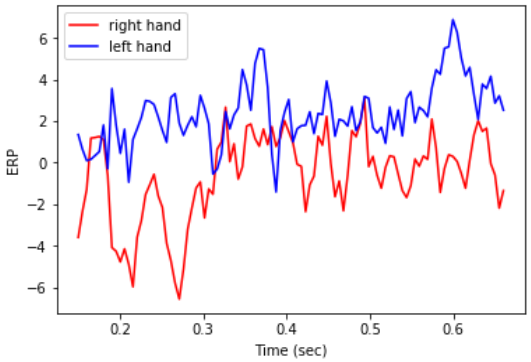

Figure [2]

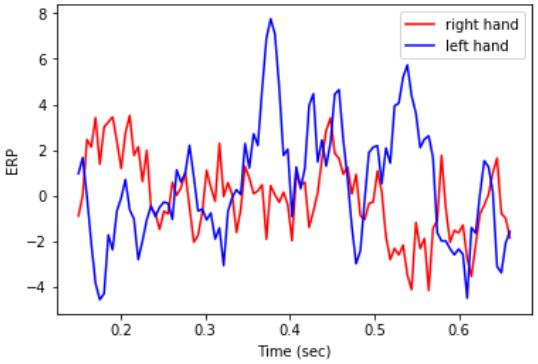

Figure [3]

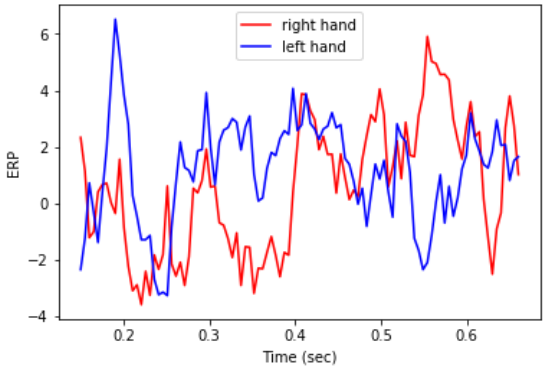

Figure [4]

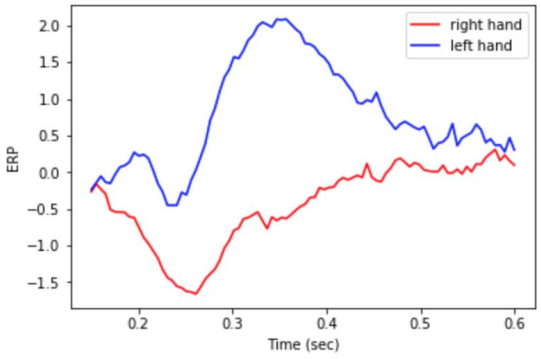

Figure [5]

We utilize Python libraries numpy, pandas, and scipy to load and store the data from .mat files. We follow basic techniques from the researchers who collected the data [4] to capture the most meaningful measurements from the data.

To identify an appropriate window over the sequence of measurements following the patient's viewing the prompt for the test condition, we attempted several offsets and window sizes until landing on

the one which yielded the highest classification accuracy with SVM. In the end, we trimmed the data such that each trial consisted of 90 measurements within the interval [.15, .6] (in seconds) timed from the display of the stimulus. We also experimented with methods of feature extraction and dimensionality reduction to simplify the learning task of the GAN. We implemented a Principle Component Analysis (PCA) as an exploratory data analysis tool to translate instances into a new coordinate system with fewer dimensions/features. While this reduction of features did provide a faster GAN training runtime, using PCA features in classification decreased overall accuracy and thus was not considered in our final experiments. Another downside to this approach is that it is not possible to reverse the data from the new feature space to its original feature space, as information is lost through the PCA transformation. Thus our GAN would be incapable of generating EEG data to be used for BCI development.

4. GAN Structure

Generative Adversarial Networks are state of the art generative models. Although they are well known for their generation of high quality synthetic images, they are able to generate samples from many different complex distributions. We propose a GAN framework to generate EEG response for left hand and right hand tasks. Our architecture consists of a generator for creating synthetic samples of EEG data, and a discriminator which works to distinguish between real and fake data. These two networks are trained in coordination. As the discriminator learns to identify real data from fake, the model provides feedback to the generator so that it can improve the quality of the synthetic data. This cycle continues throughout training, with the hope that each model strengthens the until the discriminator is unable to distinguish between real and fake data, and the generator is unable to improve upon its samples.

In our implementation of the BrainGAN model, the discriminator takes the form of a fully connected 3-layer network (hidden size = 3) with utilizing leaky ReLU and dropout techniques (p=0.6) between layers. Our Generator takes a more complex form of a 5 layer fully connected network (hidden size = 4) which also uses leaky ReLU however dropout is not used for our generator, which is illustrated in the architecture diagram of our GAN in Figure [6]. The GAN was implemented using PyTorch, where the discriminator and generator are classes which inherit from nn.module, specify and construct the network architecture, and define the forward pass. A GAN class implements instances of the discriminator and generator, trains the two components, and returns generated data. Different hyperparameters for the model include learning rates for generator and discriminator, hidden layer size, and training epochs.

We used soft labeling in order to discourage the discriminator from making overconfident decisions and to introduce regularization to the system. We replaced the discriminator's target values of 0 or 1 in the classification of real and fake images with random numbers in the range [0,0.1] for fake images and [.9,1] for real images. This led to a substantial increase in the stability of our training. We also found that in order for our training to approach stability, the discriminator required substantial regularization. We implemented this regularization, through dropout and modifying its architecture to have fewer layers and a smaller hidden sizes than the generator. We attribute this need for regularization in the discriminator to the difficulties associated with generating time series data with a fully connected neural network. The fully connected neural network is imperfect for this task, as it estimates each value of the time series individually, and cannot interpret the direct dependence of x_t on x_{t-1}. This theory can explain why, given discriminator and generator of equal model complexity, the discriminator quickly learned to identify true

and fake samples, shrinking its loss before the generator could learn to generate higher quality data. This would result in instability and divergence during training. However, aggressive regularization of the discriminator addressed this problem. In Figure [7], we see the average of 300 synthetic samples from each class generated by a particularly successful convergence of our GAN. These feature averages resemble very closely those of the real dataset.

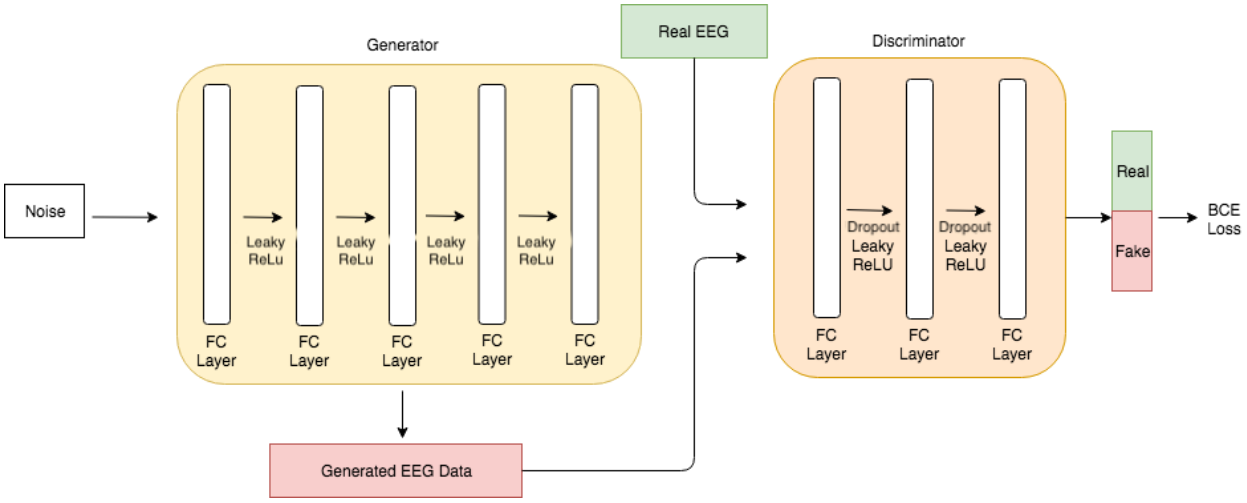

Figure [6]

5. Experiments & Results

The aim of this paper is to explore the effects of introducing synthetic EEG data to real datasets and how this affects classifier accuracy for a binary task. Our experiments seek to answer three questions as subproblems to support our hypothesis. Namely, how much data is required or sufficient for training a GAN, at what points of loss convergence does the GAN model produce generated data that most improved classifier accuracy, and what is the ideal amount of synthetic data to add to a real dataset to yield the best results of improvement in classification accuracy. These subproblems are all addressed in their respective experiments in which we treat different independent variables as hyperparameters, and tune the model accordingly to observe greatest increase in accuracy.

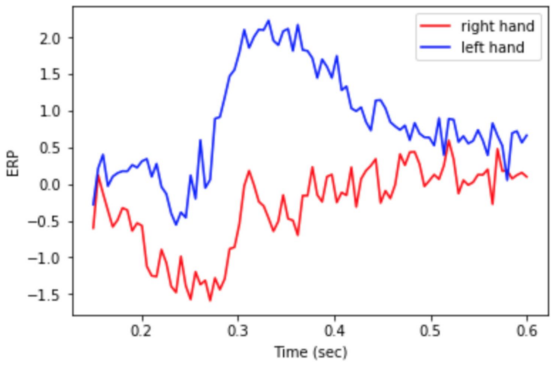

Figure [7]

5.1 Experiment 1

In this experiment we explore changing the GAN's training size as a parameter to observe how much data is needed for our model to learn the distribution. This is useful information for BCI researchers to understand the limits of GAN data generation and how much data is needed to successfully utilize this technique. For each training set size used to train our GAN, we augment a dataset of EEG data with 100

synthetic samples of left hand and right hand data. Using a held out test set composed of real EEG data, we compare the classification accuracy of a logistic regression classifier trained on only real data to that of an logisitc regression classifier trained on the augmented datasets. We run model selection for both models at every training size. We have three trials and plot the average of the results over each trial. The total size of the training data is 513 samples, or training instances.

Training Size	Real data classification accuracy			Augmented data classification accuracy		
	Trial 1	Trial 2	Trial 3	Trial 1	Trial 2	Trial 3
90%	0.661	0.707	**0.753**	0.707	**0.753**	**0.753**
80%	0.736	0.759	**0.821**	0.728	**0.821**	0.806
70%	0.823	0.756	0.797	**0.829**	0.725	0.803
60%	0.770	0.762	**0.789**	0.762	0.754	0.782
50%	**0.747**	0.741	0.661	0.719	0.722	0.710

Figure [8]

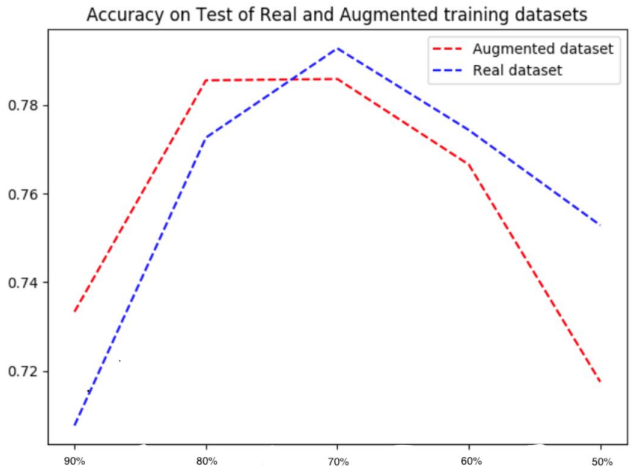

Figure [9]

Based on the charts from Figure [8] and Figure [9] we see that the GAN performs better when it has access to more data, and is no longer able to reliably increase classification accuracy with less than 60% of the training dataset, which is 307 samples of data. We also observe that augmenting the dataset with synthetic data does allow for increases in classifier accuracy when the GAN is able to see more samples. We observe lower general accuracies for training with 90% of the data, as the train-test split results in a very small test dataset. With such a small test dataset, one misclassification can skew the results. We can likely attribute the lower overall accuracies for both real and augmented datasets when trained on more data to this large impact of each misclassification on the final metric.

5.2 Experiment 2

In this experiment, we evaluated the GAN at different points of convergence throughout its training. The motivation of this experiment is the fact that the optimal solution of the GAN is reached at the Nash equilibrium, where both generator and discriminator have equal losses and cannot improve upon their respective loss. We limited the number of "snapshots" of the GAN weights to five and only select these from after the first 1000 epochs. For this experiment, we fixed the training set to 80% of the full dataset. For every point of convergence, the GAN generates 100 samples, which are added to the real training data set. Subsequently, we train a classifier on the augmented data set and test on the held out test set. Additionally, we perform model selection for every augmented data set, 5 in total.

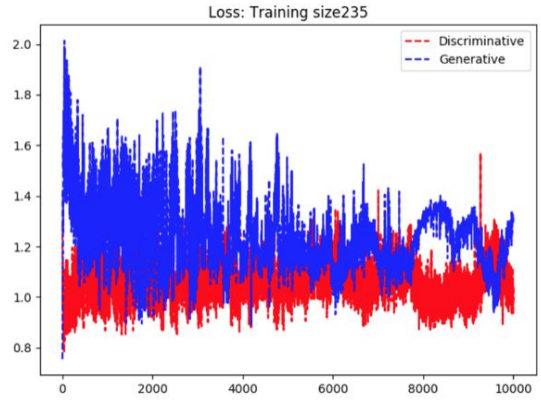

Figure [10]

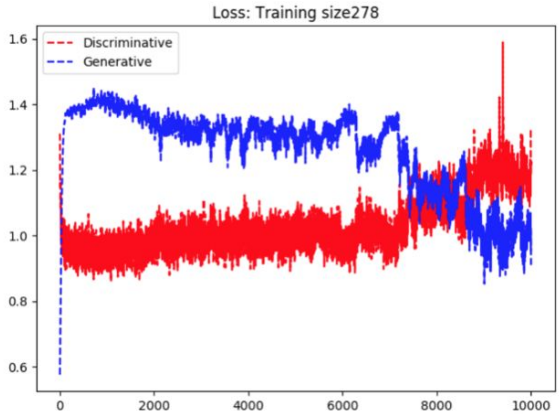

Figure [11]

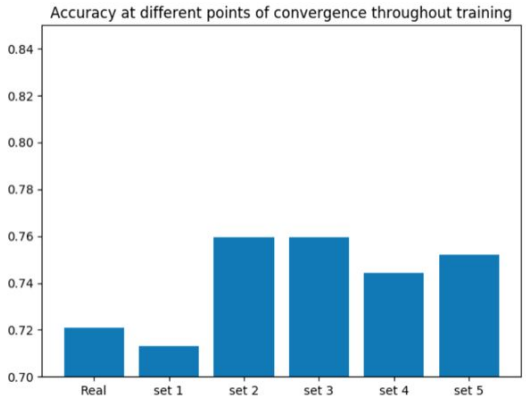

Figure [12]

We observed that datasets produced from the GAN at all convergence outperform the real data set, except for the first point of convergence. We hypothesize that data generated from the GAN at the first point of convergence is generated when the GAN has not truly learned the distribution of the data.

5.3 Experiment 3

In this experiment, we measured the effect of the ratio of synthetic data to real data in a model's training set on it's classification performance. We fixed the training size for the GAN to 513 trials (80% of the

data), with 129 (20%) reserved for the test set. We created 4 augmented training datasets, with 200, 150, 100, or 50 synthetic samples. We trained a logistic regression classifier on each of the 4 augmented datasets and recorded each model's classification accuracy on the test set. Lastly, we trained a logistic regression classifier on a training set containing only real data, and recorded its test set accuracy. Every classifier underwent model selection.

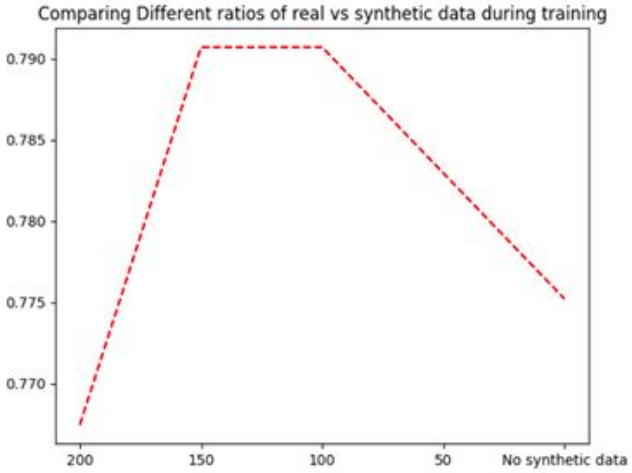

Figure [13]

We observe that the addition of 200 samples to the training dataset performs the worst, followed by the baseline condition. We hypothesize that the quality of the samples generated by the GAN while not optimal, produce a degree of noise, as might a noisy but still informative prior, yielding a regularizing effect. We conclude that the ratio of synthetic to real data in the training dataset should be treated as a hyperparameter which, when optimally tuned, improves test accuracy.

5.3 Conclusion

The inherent limitations of the GAN, namely the need for large amounts of data for training, proved to restrict the quality of the data generated by the GAN. Measurements of the difference between the mean real data and the mean synthetic data were larger than we would have expected. Moreover, the complexity of the time series data proved to be a challenge in preprocessing. Therefore, we restricted our scope to a single channel, rather than the full 22 channels. Even if we had been able to incorporate all 22 channels, the small data set would have made learning a model over so many features difficult, if even possible. In order to incorporate information from all the channels, more work may have to be done in the preprocessing step to reduce dimensionality. Nevertheless, we did observe small, but significant improvements in test accuracy, given the right hyperparameters (ratio of real data to synthetic, a significant enough training size, and a well tuned GAN architecture).

6. Future Work

We are interested in experimenting with other generative models in the future, as we believe that given the right model and a more inclusive approach to data-preprocessing, it is possible to see even better results than those we observed in this pilot study. Namely, we are interested in investigating the variational autoencoder framework as another possible generative model which may be better suited for this task. Additionally, we would also like to implement a similar GAN for use with FNIRS data to supplement the Tufts BCI Lab research and classification process. GAN generated data can provide utility not only in increasing classification accuracy, but also to various steps in the development of brain computer interfaces such as simulating conditions of brain activity useful for the interface. As the field of brain computer interfaces continues, we expect to see more implementations of various machine learning techniques to process brain data. BrainGAN serves as a pilot study to demonstrate the effectiveness and a possible use case of generative models in the field of BCI.

7. Sources

[1] Hartmann, K.G., Schirrmeister, R.T., & Ball, T. (2018). EEG-GAN: Generative adversarial networks for electroencephalograhic (EEG) brain signals. *CoRR, abs/1806.01875*.
[2] Esteban, Cristóbal, Stephanie L. Hyland, and Gunnar Rätsch. "Real-valued (medical) time series generation with recurrent conditional gans." *arXiv preprint arXiv:1706.02633* (2017).
[3] Goodfellow, Ian, et al. "Generative adversarial nets." *Advances in neural information processing systems*. 2014.
[4] Mishchenko, Yuriy, et al. “Developing a 3- to 6-State EEG-Based Brain-Computer Interface for a Virtual Robotic Manipulator Control.” *IEEE Transactions on Biomedical Engineering*, 2018, pp. 1–1., doi:10.1109/tbme.2018.2865941.
[5] Kaya, Murat, et al. “A Large Electroencephalographic Motor Imagery Dataset for Electroencephalographic Brain Computer Interfaces.” *Scientific Data*, vol. 5, 2018, p. 180211., doi:10.1038/sdata.2018.211.

Git Repository commit history:
https://github.com/dberma02/GAN_for_Synthetic_EEG_Data/commits/master

Division of labor
It should be mentioned that there was a lot of overlap between contributors. Our team utilized pair programming to create new modules, experiments, and debug current problems. This approach helped us understand various models and experiments via the communication between partners.

Dan Berman	Data preprocessing and preparation, attempted feature extraction with fourier amplitudes, baseline experiment, GAN architecture and parameter tuning, writing paper

Thomas Klimek	Experiment with PCA for dimensionality reduction, Neural Network with feature freezing, GAN architecture turning, additional data preprocessing, experimental design, writing paper
Chase Bosworth	GAN class (implementation of vanilla GAN + class method functions), experiments 1 (training sizes), experiment 2 (convergence testing), experiment 3 (real-synth data ratio), writing paper