Biophysical modeling of EEG signals from current dipoles in human cortex

Kamilla Sulebakk, Marcus Berget and Sebastian Amundsen

November 2021

ABSTRACT: This research paper presents an automated system for efficient localization of abnormalities in EEG signals using artificial neural networks. When studying the data set using a feed forward neural network we found that the ReLu and Tanh activation functions performed the best, given that the loss converged faster for these, than the Sigmoid function. We also found that a normal distributed noise with a standard deviation of $\sigma = 0.5$ was the maximum noise we could safely add to our data set while still managing to produce acceptable fits. Further, the Principle Component analysis suggest that the components of the EEG-signals are highly correlated. A dimensionality reduction to 10 principle components gives a noticeable lower test-data error compared to using all dimensions of the data. Overall, the trained neural networks managed to perform precise localization of the abnormalities located in the cortex of 1000 patients.

1 Introduction

According to the World Health Organization (WHO), nearly 50 million people suffer from epilepsy worldwide. Further there were more than 300 000 new cases of cancer related to brain tumors in 2020. The two mentioned diseases can arise at any age. They affect the cortical neural activity, and can be studied utilizing electroencephalography (EEG) [2].

EEG are among the most important techniques for studying cognition and certain stimuli in the human brain, without having to make any break in the skin [2]. Generally, information gained from EEG signals plays an important role for detecting and localizing abnormalities and diseases in the brain [10], such as epilepsy and tumors among many. By accurately locating the source that causes abnormal signals, EEG scans enable surgery for removal of diseases with minimal damage of healthy brain tissues.

EEG signals arise from cortical neural activity and are typically described in terms of current dipoles [2]. In this project we generate data by modeling current dipoles and EEG signals arising from abnormalities in the brain. Using a feed forward neural network we want to study the fluctuations in neural activity and train our model to localize the position of the disease causing unusual brain activity. While training our model we will explore different types of machine learning techniques, such as principal component analysis and cross-validation resampling method, in order to maximize the accuracy of the model.

2 Theory

Generating a EEG data set

Electroencephalography (EEG) is one of the most important techniques for studying cognition and disease in the brain non invasively [2]. An EEG is a test used to detect abnormalities in brain waves. In practise, this involves electrodes consisting of small metal disks connected to the surface of the scalp. The electrodes detect electrical charges that result from activity of the brain cells. In cases where certain areas of the brain come out as more active than others, it might indicate abnormalities, in which can be signs of disease. In other words, the EEG technique can be used to evaluate multiple types of brain disorders, such as lesions of the brain, Alzheimer's disease, epilepsy or brain tumors [3].

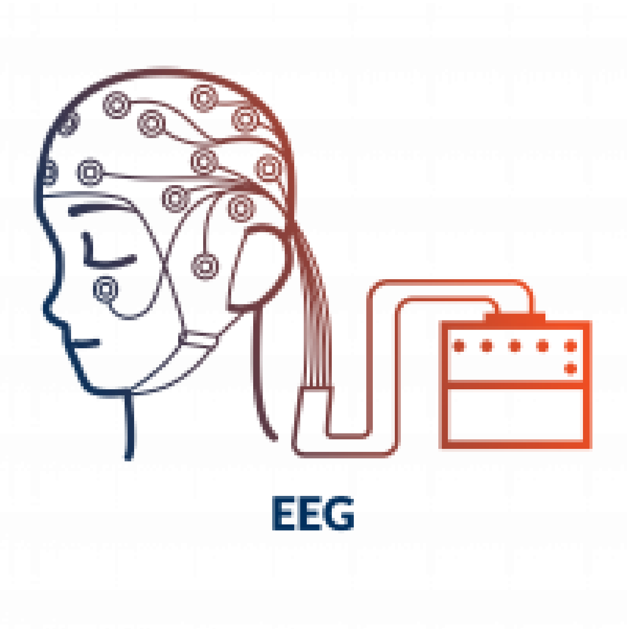

Figure 1: Illustration of the EEG method [4].

An illustration of the typical EEG measurement setup is depicted in Figure 1. Abnormality in EEG shows either as an increased amount of slow activity with a frequency below 8 Hz or as the appearance of abnormal waveforms, such as waves with a pointed shape and the occurrence of special patterns[11] . The main aim of this project is to train a model, to predict the localization of such abnormalities in the brain of patients. However, in lack of real data in consistent with our defined problem, we will be generating the data our self by utilizing work done by external developers. Credit goes to the people behind the article "Biophysically detailed forward modeling of the neural origin of EEG and MEG signals" [2], which laid the foundation for the generation of all our data for this project.

The signals received from EEG are known to originate from cortical neural activity, which are often described by using current dipoles [2]. It is therefor reasonable to implement current dipoles in the brain for when generating a biophysical modeling of EEG signals. The brain model which we will be using is called the New York Head model [3], and is based on high-resolution anatomical MRI-data from 152 adult heads. The model utilizes the software tool LFPy, which is a Python module for calculation of extracellular potentials from multicompartment neuron models [7]. This model takes into account that electrical potentials are effected by the geometries and conductivities of various parts of the head [2].

An optimal model of EEG signals would have consisted of multiple dipole moments. However, as such a model is complicated and computationally expensive, we will in this project only introduce one single dipole approximation $\mathbf{p}(t)$ for each multicompartmental neuron simulation. In this context, by multicompartmental modelling we refer to the widely used models within neuroscience, which acurately manufactures electical properties of single neurons. The sigle-dipole approximation might sound like a substantial simplification of the real biophysical properties, nevertheless it actually turns out to give a realistic modelling of EEG signals, when handling the single dipole moment, as an abnormality in the brain. We will be thinking of the abnormality as an epileptic seizures or a tumor in the brain, which among normal activity in the brain would have stuck out. The single-dipole approximation is implemented by summing up the multiple current dipole moments,

$$\mathbf{p}(t) = \sum_{k=1}^{M} \mathbf{p}_k(t) = \sum_{k=1}^{M} I_k^{\text{axial}}(t)\mathbf{d}_k,$$

where I^{axial} is the current flowing along the neurite with the distance vector $\mathbf{d}_k$ and M denotes the number of axial currents [2]. The data set we will be using in this project will consist of measures of different EEG signals at a given time from 1000 patients. This means that we for each patient pick a random location (at $t = 0$) for the single current dipole.

EEG signals are generated from synaptic inputs to cells in the cortex. Synaptic inputs are electrical (or chemical) signals that are being transmitted from one neuron to another, causing changes in the membrane potential of the neurons. In other words, neurons are specialized to pass signals, and synapses are the structures that make this transmission possible [12].
Imagining a small part of the cortex, all of these cells will have dendrites pointing upwards in the same direction (lets say the z-direction). Due to rotational symmetry around the z-axis, the contributions in the x- and y-direction will cancel. This is illustrated in Figure 2. What we see is that the extracellular potential is configured in all sorts of weird ways (A-C) when there is only one synaptic input, while the extracellular potential reminds more of a dipole when we have multiple synaptic inputs (D-F). We can there-

fore argue that the total contribution to the extracellular potential can be modelled as a dipole in the z -direction for the case where we have multiple synaptic inputs. Hence, for each dipole moment in our simulations, we pass multiple synaptic inputs, and make sure to rotate the positions of the dipoles such that it is orientated along the depth of the cortex.

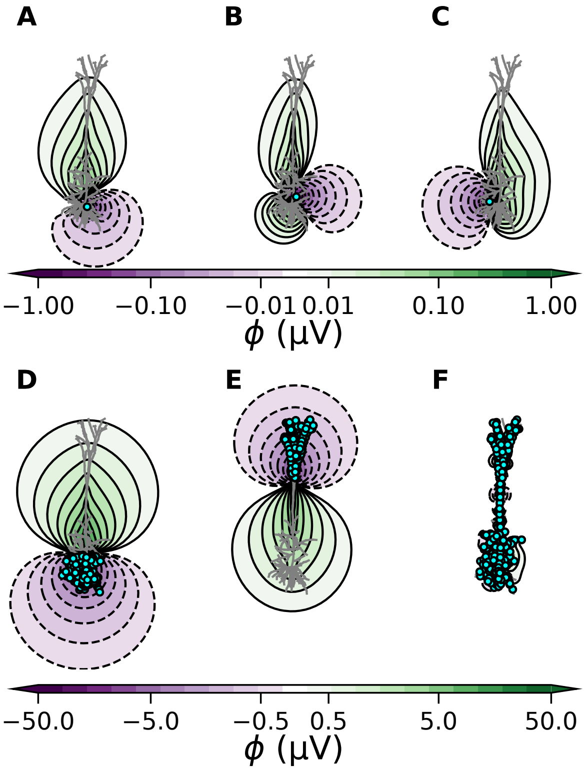

Figure 2: Extracellular potential from lonely synaptic input (A-C) and extracellular potential from multiple synaptic inputs (D-F).

The cortex matrix consists of 74382 points, which refer to the number of possible positions of the dipole moment in the cortex. When generating our data set, we will for each sample randomly pick the position of the dipole moment, such that one sample corresponds to one patient. In our head model we are considering 231 electrodes uniformly distributed across the cortex, meaning that each EEG sample will consist of this many signals for each time step. However, we are not interested in the time evolution of the signals as this does not affect nor say anything about the position of the dipole moment, and we therefor simply pick out the EEG signals for when $t = 0$ (note that the choice of time step could have been randomly picked). Our final design matrix will then consist of 1000 rows, corresponding to each patient, and 231 columns also refereed to as features, representing the signal of each electrode. The final output we are trying to predict is then the one dimensional vector with length 1000, where each element consists of the x-, y- and z- position of the dipole moment. An example of how the input EEG signals may look like is given in appendix A, where we also have marked the dipole moment with a yellow star.

Noise

All experimental data include some amount of noise, and data for EEG signals are no exception. Artifacts are signals recorded by EEG but not generated by brain. An artifact occurs when there is noise registered by the system that contamines the neural EEG data. Examples of physiological artifacts can be muscle activity, such as eye movements or blinks, perspiration and respiration. Technical artifacts however might be cable and body movements or AC electric and electromagnetic interferences, among other things. Some artifact may mimic true epileptiform abnormalities or brain tumors. Awareness of such noise is important in distinguishing artifact from brain waves [13]. Thus, in order to make our data set, and hence the generated EEG signals realistic, we add normally distributed noise, with mean mean $\mu = 0$, when performing the EEG simulations. Further, we will in this project study how different types and values of noise affects the performance of the fit.

Methods

Feed Forward Neural Network

In this paper we will be using a feed forward neural network when fitting the generated EEG data. For a qualitative description of the set up of a feed forward neural network, please have a look at our previous paper [6]. As done in our previous projects, we split our data into training and testing sets in order to make the model compatible with multiple data sets. In addition to study the model when splitting the data randomly into random train and test subsets, we will be using K-fold cross-validation, in which we presented in our first project. Moreover, we will standardize the features by removing the mean and scaling to unit variance. The purpose of the neural network will be to locate the positions of the dipole moments in the cortex of the many patients. For the fitting, we will be studying different types of activation functions.

When training our model, we simply try to minimize the loss. In our neural network we will be using the quadratic cost function with Ridge regularization parameter (L_2-norm), given by:

$$\frac{1}{2}\sum_{i=1}^{n}(a_i - t_i)^2 + \lambda||w||_2^2, \tag{1}$$

where a_i is the output, t_i is the target data, λ is the Ridge regularization parameter and w is the weights. As discussed in our previous project, the quadratic loss function and the L_2- norm is often convenient to use given their versatility and simple derivatives. The Ridge regularisation term is also beneficial in case we have heavy outliers [5].

The purpose of an activation function is to transform the input to a single neuron in the network and cause a non-linear relation between input- and output- values. There are many types of activation functions, but in this paper we will be studying the performance of the neural network for the three following activation functions:

- The Sigmoid function: $f(z) = \frac{1}{1+e^{-z}}$
- Hyperbolic tangent: $f(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$
- ReLU: $f(z) = \max\{0, x\}$.

Comparing the deviations from the actual data and the predictions given each of the activation functions can be done by studying the accuracy and loss while training.

R2 Score

We will also be looking at the coefficient of determination R^2 of our activation functions as a supplementary metric for how our model performs. The R2 score works by measuring the amount of variance in the predictions our model makes. The R2 score is given by:

$$R^2(\hat{y}, \tilde{y}) = 1 - \frac{\sum_{i=0}^{n-1}(y_i - \tilde{y}_i)^2}{\sum_{i=0}^{n-1}(y_i - \bar{y})^2}, \tag{2}$$

Where the mean value of y_i is defined by $\bar{y}$:

$$\bar{y} = \frac{1}{n}\sum_{i=0}^{n-1} y_i.$$

In general the model is better for higher R^2 values, where the maximum value is 1 and the minimum value is 0. If the R^2 score is high the model will generally perform a better fit on unseen data. For this to be the case it is important that we calculate the R^2 score for the validation set and not the training data.

Principle Component Analysis

Further in this project we will perform a principle component analysis (PCA). PCA is a method for analyzing/transforming data in order to do dimensionality reduction, among other things. PCA can be broken into 6 steps:

1. Scale the data. This is important for two reasons. First, it's important that the vectors have the same dimensions and orders of magnitude. This way, the PCA won't be biased towards vectors with larger values. Secondly, centering the data will make the next steps computationally easier.
2. Compute the covariance-matrix for the features.
3. Compute the eigenvalues and eigenvectors for the covariance-matrix.
4. Sort the eigenvalues. The largest eigenvalue is referred to as the eigenvalue of PC1 (Principle component 1), where it's corresponding eigenvector is referred to as the eigenvector for PC1. Intuitively, PC1 is the line that intercepts the data such a way that the data points projected onto this line has the largest variance.
5. Create the matrix $\mathbf{W}$ which consists of the first k principle component eigenvectors.
6. Transform the design matrix X by the transformation $\mathbf{T} = \mathbf{XW}$. The matrix $\mathbf{T}$ then represents the original design matrix, but with a fewer number of features.

There are several advantages of doing dimensionality reduction. But for the purpose in this project, it is mainly to avoid over-fitting. In general, the greater the number of explanatory features, the higher the chance of over-fitting. In addition, by analyzing how many features it is possible to remove before the model starts learning poorly, will give us an insight into the data. For doing the PCA we will use the PCA-library provided by scikit-Learn [8].

Explained Variance Ratio

The explained variance ratio of the principle components is a metric that tells how much of the total variance, a principle component accounts for. In other words, the explained variance ratio tells how important a specific principle component is. It is common to plot the explained variance ratio for the principle components to get an idea of how important each PC is. These plots are commonly referred to as scree plots, as they often have the shape of a scree, or an elbow.

Variation of noise and parameter values

In our study we will be investigating how noisy our data can be before the neural network is unable to recognise any meaningfull pattern. We will therefore adjust the added noise to our generated data set and study how the loss changes as a function of added noise. We can decide whether a loss value is high or low, by comparing our value for loss to the scale of our data set. By this we mean to for example compare the loss to the uncertainties which are prevalent in our model.

The parameters used when training a neural network is far from trivial, so a big part of our task will also be to test which NN parameters that gives the lowest or fastest convergence. This will be done quite pragmatically, by finding the optimal learning rate, number of layers, sequentially. This assumes that the parameters are independent of each other, which isn't the case. But it will likely give a good parameter combination nevertheless.

3 Results

Feed Forward Neural Network

Table 1: Table of the optimal NN parameters for training on the EEG-data. η and λ refers to the learning rate and the ridge regularization parameter, respectively. The noice added to the data has a deviation of 0.1.

η	0.0001
λ	1e-5
Hidden layers	5
Hidden neurons	200
Batch size	40

Table 1 shows an overview of the values of parameters that gave the optimal results when training on the feed forward neural network.

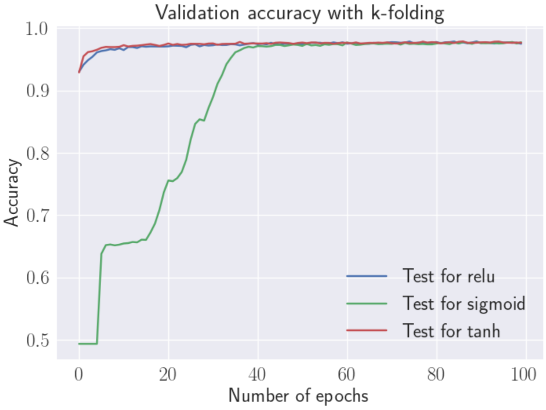

Figure 3: The validation accuracy for the neural network with k-folding for the three different activation functions: ReLu, Sigmoid and Tanh. The noice added to the data has a deviation of 0.1.

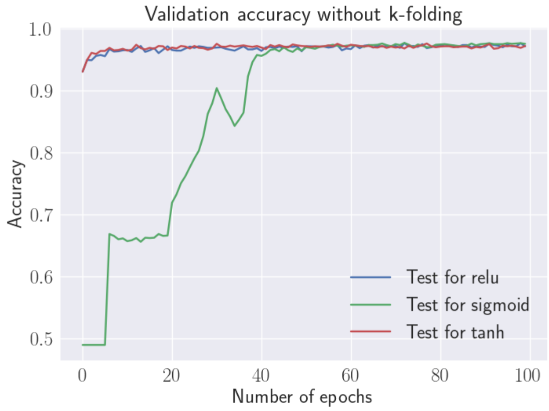

Figure 4: The validation accuracy for the neural network for the three different activation functions: ReLu, Sigmoid and Tanh. The noice added to the data has a deviation of 0.1.

In Figure 3 we have provided the validation accuracy for the different activation functions given by the k-fold cross validation method. The same plot can be seen without the use of the validation method in Figure 4.

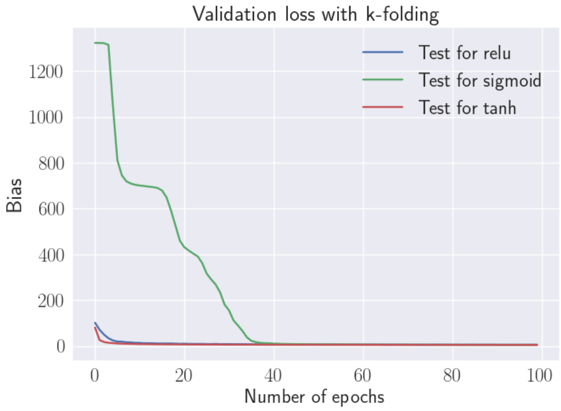

Figure 5: The validation loss for the neural network with the three different activation functions: ReLu, Sigmoid and Tanh. The noice added to the data has a deviation of 0.1.

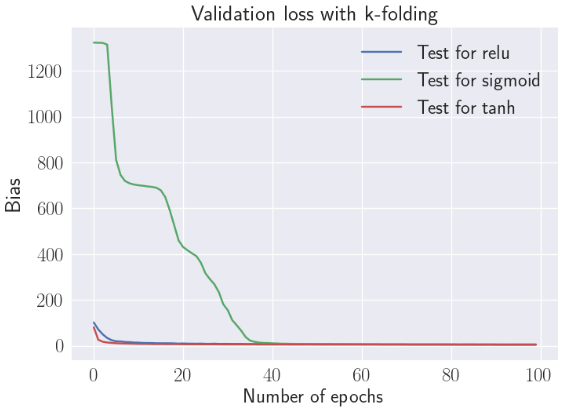

Figure 6: The validation loss for the neural network with k-folding for the three different activation functions: ReLu, Sigmoid and Tanh. The noice added to the data has a deviation of 0.1.

Figure 5 shows the validation loss given by the activation functions for k-fold cross validation. We have provided the same plot without use of the validation method in Figure 6.

We notice the same trend in from the results provided in Figure 3 and 4 as we did in Figure 5 and 6, where the Sigmoid requires more epochs before it performs ”up to par” with the two other methods. We can see a clear trend in the accuracy score and loss for the different activation functions in our neural network. Both ReLu and Tanh reaches acceptable accuracies and losses after around the same number of epochs, while the Sigmoid activation function requires many more epochs before it reaches a similar value.

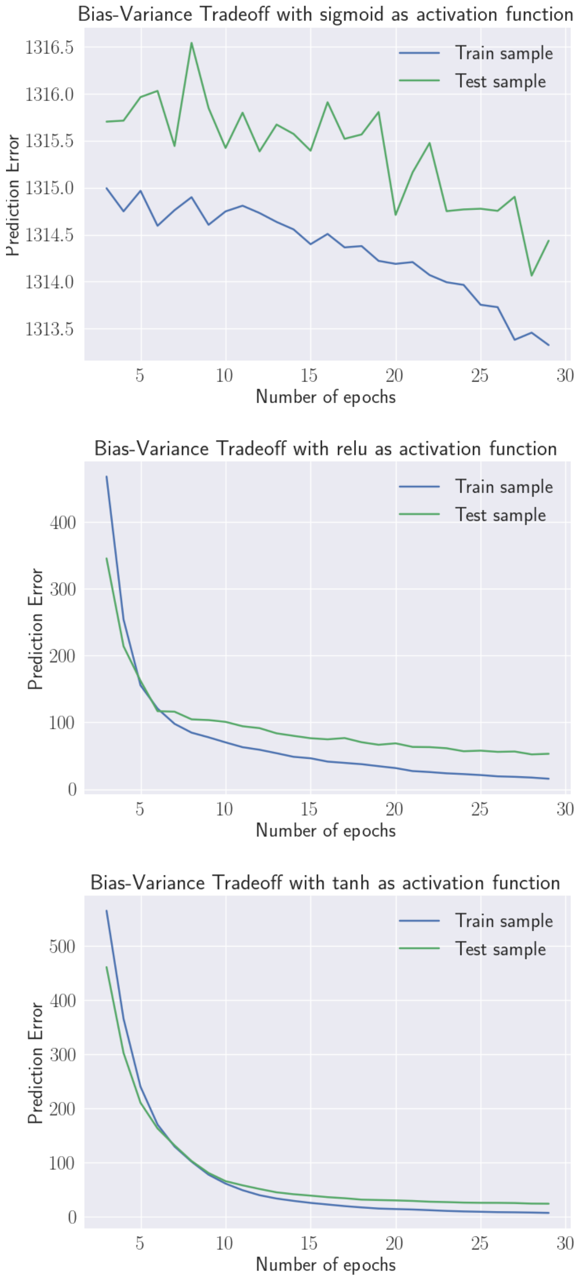

Figure 7: The figure provides three plots for the training and test sample errors as function of epochs, for the different activation functions. The noice added has a deviation of 0.1.

In Figure 7 we have provided three plots of the bias-variance trade-off for when using Sigmoid, ReLu and Tanh as activation function, respectively. The noise added has a deviation of 0.1 form the mean. First of all we notice that the Sigmoid curve has a funny shape, which does not seem to stabilize. The result shows that even after 30 epochs, the value of the error for neither of the samples is close to what is hoped for. This indicates that both the variance and bias is high for the plottet values of epochs. ReLu and tan, on the other side, is clearly approaching 0, and we notice that the variance is increasing with an increasing number of epochs.

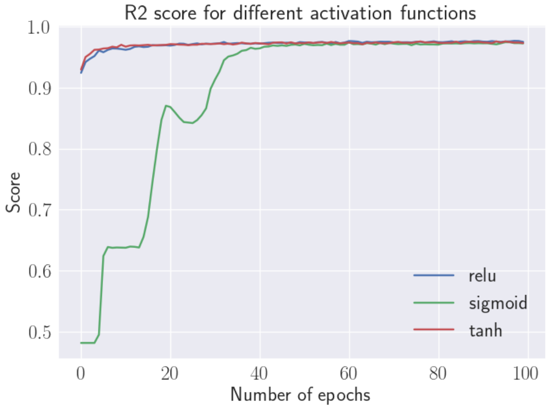

Figure 8: The R2 score for the activation functions: ReLu, Sigmoid and Tanh. The noice added to the data has a deviation of 0.1.

Figure 8 shows the R2 score for the different activation functions given by the k-fold cross validation method. We see the same pattern for the Sigmoid activation function, as for when plotting the accuracies and losses. When using Sigmoid as acivation function we see that it takes many more epochs for the R2 score to reach an acceptable value. However, when using ReLu or Tanh, the R2 score reaches a value close to 1, already after 5 epochs.

Table 2: The loss and accuracy given different values for added noise.

Noise deviation	Loss	Accuracy
0.001	3.2109	0.9815
0.005	3.2801	0.9850
0.01	3.6257	0.9825
0.05	4.0202	0.9815
0.1	5.6107	0.9785
0.5	52.2475	0.9305
1.0	152.1261	0.8620
5.0	1278.3146	0.8620
10	1343.1093	0.4935

We have the validation loss and accuracy in In Table 2 we have presented the validation loss and accuracy for different values for the noise added to the data set. We can see that after reached a noise deviation of around 0.5 the loss becomes a lot larger and the accuracy seems to diminish. This is probably around where the threshold lies for how much noise we can add to our data.

PCA

In the following plots the EEG-signals are generated by adding a normal distributed noise where σ denotes the standard deviation of the noise, with a mean $\mu = 0$.

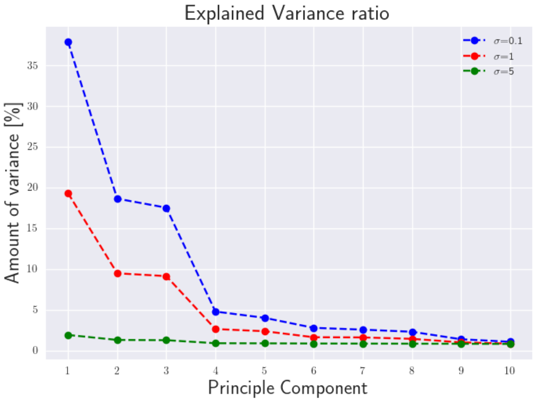

Figure 9: Plot of the explained variance ratio of the first 10 principle components, where the PCA is done on data with varying noise.

When doing an principle component analysis on the data, we observe that explained variance ratio has the characteristic shape of a scree (or elbow), where most of the total variance is spread along the first principle components (at least for $\sigma = 0.1$ and $\sigma = 1$). We also observe that as the noise is increased, the less explained variance each of the first principle components account for. For $\sigma = 5$ it seems as if each principle component has approximately the same explained variance.

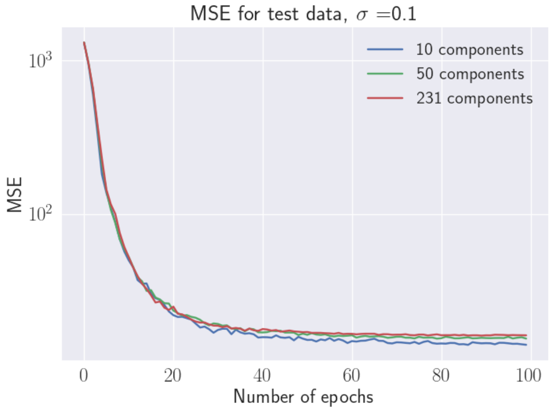

Figure 10: Plot of the MSE of the test data as function of the number epochs, when training the NN on a varying number of principle components.

Above, in figure 10 we see that the test MSE converges towards a lower value, when reducing the number of principle components.

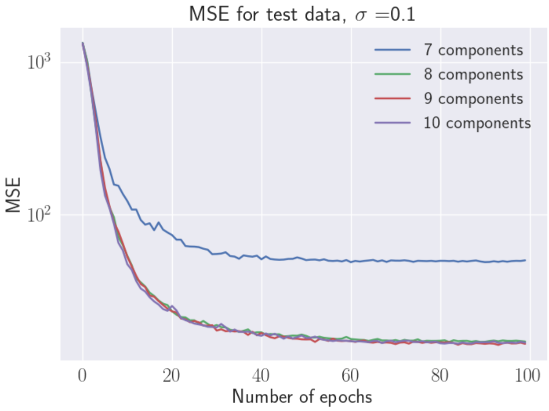

Figure 11: Plot of the MSE of the test data as function of the number epochs, when training the NN on a varying number of principle components.

Figure 11 is similar to Figure 10, but here we see when the reduction of principle components give an increasing return of MSE, which seems to happen at an usage of 7 principle components. The total amount of the explained variance ratio when using the respective PC's are given in the table below:

Table 3: The explained variance summed over the number of components.

Number of PC's	7	8	9	10
Total explained variance [%]	86	89	91	92

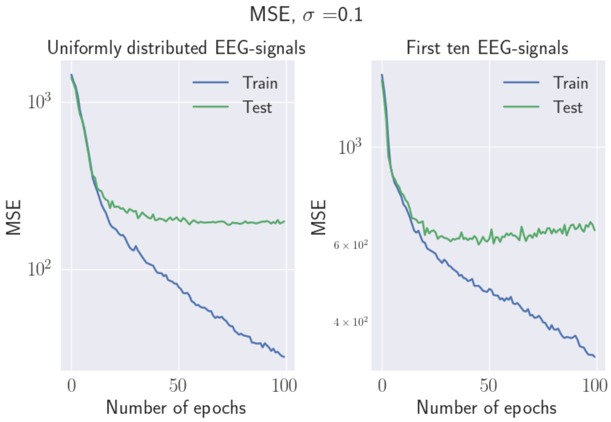

Figure 12: Here we compare training the network using 10 signals picked out uniformly from the total EEG-signals (left), compared to using the first 10 signals (right).

In the figure above we see that using 10 features extracted uniformly from the EEG-signals give a significantly better result, compared to when using the first 10 features of the EEG-signals.

4 Discussion

Feed Forward Neural Network

We found that the neural network managed to find the position of the dipole moment quite accurately. We can therefore conclude that our neural network works on this specific set of data.

Even though ReLu and Tanh behave quite similarly, Tanh has a slightly faster convergence towards our desired properties (high accuracy, low bias). This, in addition to Tanh being less sensitive to parameter initializations [6], is why we decided to use this activation function in the rest of our simulations for the neural network.

The slow convergence rate when using Sigmoid, probably stems from a vanishing gradient, which is the main disadvantage of using Sigmoid [6]. This further means that Sigmoid likely gives more expensive computations, due to the computational handling of very small numbers. On the other hand, Tanh is known to be more computationally efficient which allows for quicker run-times [9], which we also observed in our previous machine-learning project [6].

It is interesting to note that the ”trajectory” of the Sigmoid function seems to become a lot smoother as we implement the k-fold cross validation method. This could come from the fact that when we use k-folding we resample the data, which in practise could mean that different deviations between samples ”cancel each other out”, leading to a more smooth convergence for both bias and accuracy. Other than these smoother convergences for the Sigmoid function there isn’t any huge differences in loss and validation with or without k-folding.

It is important to note that the R2 score could vary due to random selection of training and validation sets. We do however assume that both the ReLu and Tanh will fit new data well after around 20 epochs by qualitatively looking at the plot for the R2 score in Figure 8 and the bias-variance tradeoff in Figure 7. This is not the case for the Sigmoid, as this function need more epochs before it reaches an equivalent R2 score and error as the two other activation functions. This is the same trend we saw in the accuracy plot for the Sigmoid function.

We found that a normal distribution of added noise with a deviation of 0.5 was the highest amount of acceptable noise we could add, when studying the normal distribution noise model. One interesting point to investigate in the future could be to add different noise models and see how our model would behave. We could for example have added periodic noises of some kind. It would be interesting to look at where the ”value treshold” for these noise models would lie.

PCA

As mentioned, the scree plot (Figure 9) from the principle component analysis suggest that most of the total variance is spread along the first principle components. Combined with the results from Figure 10, this demonstrates that a PCA dimensionality reduction indeed works for our data.

But what exactly does this tell us about our data? The answer to this can be found by looking at step 2 of the PCA. The covariance-matrix from which the eigenvalues - and vector are computed, contains the information about how much each feature varies together, or how much they are correlated. So by only looking at the the results mentioned above, all we can say is that the features, which essentially is the spatial information, are correlated. So in the case of the EEG-signals we are analyzing, this probably means that if you look at the strength of two neighbouring EEG-signals they are likely to be pretty similar. This is also suggested in Figure 12 where we see that picking out 10 EEG-signals uniformly distributed gives a significantly lower test error, compared to when using the first 10 features of the EEG-signals. If the neighbouring EEG-signals are highly correlated, you will naturally have to use EEG-signals distributed from all regions to account for as much variance as possible.

So we have seen that reducing the number of features can reduce the test error, but in Figure 11 we see that at when reducing the number of dimensions down to 7, the test error suddenly shoots up. By again looking at Figure 9, there is nothing there that suggest that we should have such a sudden jump in the error at 7 components. What makes such a huge difference by using PC’s accounting for 86% v.s. PC’s accounting for 89% of the total variance (table 3)? The answer to that is that we’re not quite sure. But nevertheless, we observe that the total amount of explained variance has to be above a certain threshold of about 89%. This goes to show that one has to be careful when using the explained variance to predict how many PC’s are appropriate using - reduce them enough to avoid significant overfitting, but don’t get too greedy.

5 Conclusion

In this we have generated EEG signals and trained a neural network to localize the positions of abnormalities, modelled as single dipoles, in the cortex of the brain for 1000 patients. Our model managed to find the position of the dipole moment, obtaining a faster convergence when using the ReLu and Tanh activation functions. We also explored how much noise we could add to our generated data set before our model was incapable of producing acceptable fits, which was at a normal distributed noise of $\mu = 0$ and $\sigma = 0.5$. Further we performed a principle component analysis that showed that the components of an EEG-signal are highly correlated. By doing a principle component transformation of the data ($\sigma = 0.1$), the first 10 PC’s account for approximately 92 % of the total explained variance ratio. Training the network by using the first 10 PC’s gives a noticeable lower test error, compared to using all 231 principle components. The amount of noise added to the data has a significant effect on the total variance explained by each principle component, and we find that by adding a noise of $\sigma = 5$ the explained variance of each PC is more or less the same, indicating that the correlation of the features have vanished.

Appendix A

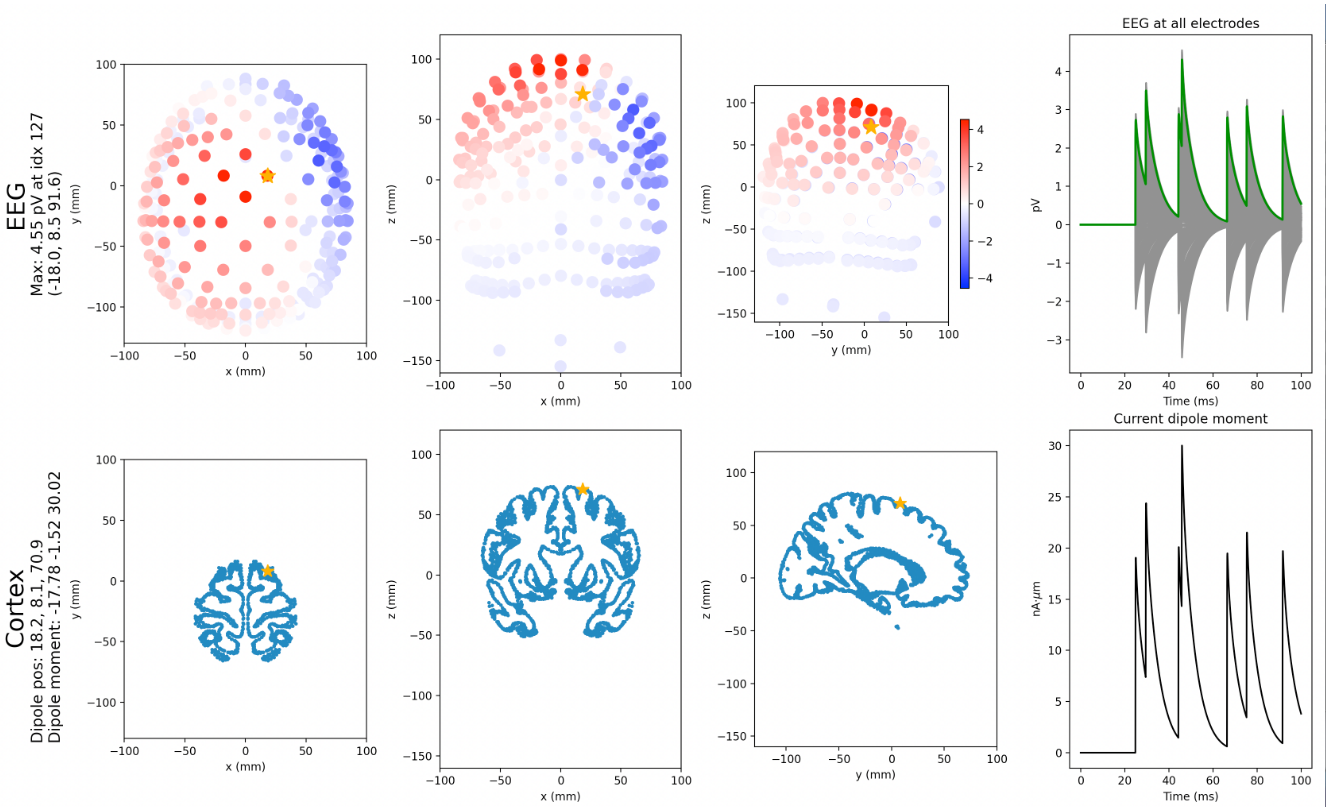

Figure 13: Plot of the EEG signal when the dipole moment was at its largest.

Appendix B - Links to existing literature

To the best of our knowledge, this is the first EEG study to employ deep learning algorithm for the automated regression of the localization of abnormal activity (dipole moments) in the cortex of the brain. However, there have been multiple studies for the use of artificial neural networks to handle EEG-signals. Nevertheless, in line with the data sets provided, these algorithms have most commonly been used for detection of abnormalities in the cortex of the brain and for classification of whether the EEG-signals belongs to normal or abnormal activity.

In the article ”Deep convolutional neural network for the automated detection and diagnosis of seizure using EEG signals” the authors have utilized convolutional neural networks with 13 layers in order to detect normal, preictal, and seicure classes of EEG-signals. The input data used in the algorithm was obtained from five patients and contained a total of 100 EEG-signals in each class. Similarly to part of our research the authors used leaky ReLu as activation function for their neural network. However as the authors were studying a classification problem SoftMax was used as last layer activation function in order to predict which class the EEG-signal belonged to. The accuracy, specificity (true negative rate) and sensitivity (true positive rate) achieved were 88.67%, 90.00% and

95.00%, respectively.

In the article ”Automated Detection of Brain Tumor in EEG Signals Using Artificial Neural Networks” the researchers have developed an automated system for efficient detection of brain tumors in EEG signals using feed forward backpropagation neural network. The trained neural network when fed with test EEG signals, effectively classified the presence of brain tumor in the EEG signal. An important part of the research in general, and the analysis of the EEG signals in particular were to employ Fast Fourier transform in order to eliminate the artifacts present in the EEG signals. By organizing the data into training and testing sets, the neural networks managed to detect the presence of brain tumor in EEG-signals. The article tells nothing about the number of features used in the network or values for other types of parameters, however, it is safe to believe that by trial and error the values giving the most precise results have been used. The network was evaluated with 325 samples of EEG data, whereas 163 of these correspond to the presence of brain tumor and the other 162 samples were EEG data without brain tumor. The accuracy of the evaluation were 94.48% and 98.77% for the normal and abnormal EEG signals, respectively.

It is not an easy task to compare our results to the findings in the two articles provided, especially because we have studied a regression problem rather than a classification problem. Moreover, we have used other machine learning algorithms than the ones utilized in the studies given in the articles. However, it is interesting to see how machine learning can be used to study different aspects of EEG data sets. If we had more time, it would definitely have been interesting to generate EEG signals that we could have used to perform classification studies, as the authors did in the discussed articles.

References

[1] Hastie, T, Tibshirani, R., Friedman, J., 2016, The Elements of Statistical Learning

[2] Næss, S., Halnes, G., Hagen, E., Hagler, D,J., Dale, A, M., Einevoll, G, T., Ness, T,V., 2020, Biophysically detailed forward modeling of the neural origin of EEG and MEG signals

[3] Johns Hopkins medicine.,2021, Electroencephalogram

[4] Brain vision., 2021, EEG

[5] Sulebakk, K., Berget, M., Amundsen, S., 2021, Regression models and resampling methods

[6] Sulebakk, K., Berget, M., Amundsen, S., 2021, Classification and Regression, from linear and logistic re- gression to neural networks

[7] `https://lfpy.readthedocs.io/en/latest/readme.html#summary`

[8] `https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html`

[9] Gharat, S., 2019, What, Why and Which?? Activation Functions

[10] M. Murugesan, R. Sukanesh, 2009, Automated Detection of Brain Tumor in EEG Signals Using Artificial Neural Networks

[11] `https://tidsskriftet.no/2013/01/tema-klinisk-nevrofysiologi/eeg-nar-og-hvordan`

[12] Sterratt, D., Graham, B., Gilles, A., Willshaw, D., 2011, Principles of computational modelling in neuroscience

[13] `https://www.bitbrain.com/blog/eeg-artifacts`