UNIVERSITY
OF OSLO

Master's thesis

Localization and identification of Neural Sources from simulated EEG Signals

Kamilla Ida Julie Sulebakk

Biological and Medical Physics
60 ECTS study points

Department of Physics
Faculty of Mathematics and Natural Sciences

Autumn 2023

Kamilla Ida Julie Sulebakk

Localization and identification of Neural Sources from simulated EEG Signals

Acknowledgements

Massive thank-yous to my supervisor Gaute Einevoll and my co-supervisor Torbjørn Ness.

Contents

Acknowledgements **i**

Introduction **v**

1 Introduction to Neuroscience **1**
1.1 The Neuron 1
1.2 Spike Trains and Action Potentials 3
1.3 Anatomy of the Cortex 4

2 Electroencephalograpy **5**
2.1 The Physiological basis of EEG signaling 5
2.2 The Inverse Problem and Source Localization 7
2.3 Head Models 8
2.3.1 The New York Head Model 8
2.4 The Current Dipole Approximation 10
2.5 The Effect of Dipole Orientation 13
2.6 Solving the EEG inverse problem 14

3 Creating EEG Data **17**
3.1 Simulation of EEG Signals from single dipoles 17
3.2 Noise 18
3.2.1 Final Dataset 19

4 DiLoc - A Neural Network Apporach for Source Localization **21**
4.1 Machine Learning and Neural Networks 21
4.1.1 Neural Networks 22
4.2 The creation of DiLoc 24
4.2.1 Architecture 24
4.2.2 Activation Functions 25
4.2.3 Initialization 28

5 Training the DiLoc Neural Network **31**
5.1 Training Methodology Overview 31
5.2 Data Preparation 32

5.2.1 Data Segmentation 32
5.2.2 Data Scaling 33
5.3 Cost Function 33
5.4 Back propagation algorithm 34
5.5 Optimization Algorithm 37
5.6 Regularization Techniques 39
5.7 Learning Rate Scheduling 39

6 Localizing Single Current Dipole Sources 41
6.1 Testing Methodology 41
6.2 Need a name 42
6.3 Performance Evaluation 44
6.3.1 Position Spesific Performance 44
6.3.2 Performance at Different Brain Structures 47

7 Convolutional Neural Network Approach for Localizing Single Dipole Sources 51
7.0.1 Data Set 52
7.0.2 Performance Evaluation 54

8 Extending the DiLoc Network 57
8.1 Method: Adjustments in Data Set and Architecture 57
8.1.1 Scaling of Target Values 58
8.1.2 Sigmoid as Last Layer Activation Function 58
8.1.3 Choosing an Optimal Cost Function 59
8.1.4 Overview on Arcitecture and Hyperparameters 63
8.1.5 Metrics of success 64
8.2 Predicting Single Dipole Sources with Varying Magnitudes . . 65
8.2.1 Adjusting Data Set and Hyperparameters 65
8.2.2 Performance Evaluation 66
8.3 Predicting Region of Active Correlated Current Dipoles with Amplitudes 71
8.3.1 Adjustments in Data Set and Architecture 71
8.3.2 Performance Evaluation 73
8.4 Localizing Multiple Dipole Sources 76
8.5 Previous work 76
8.5.1 Adjustments in Data Set and Architecture 77

Introduction

This is only from project description and needs to be rewritten. Electroencephalography (EEG) is a method for recording electric potentials stemming from neural activity at the surface of the human head, and it has important scientific and clinical applications. An important issue in EEG signal analysis is so-called source localization where the goal is to localize the source generators, that is, the neural populations that are generating specific EEG signal components. An important example is the localization of the seizure onset zone in EEG recordings from patients with epilepsy. A drawback of EEG signals is however that they tend to be difficult to link to the exact neural activity that is generating the signals.

Source localization from EEG signals has been extensively investigated during the last decades, and a large variety of different methods have been developed. Source localization is very technically challenging: because the number of EEG electrodes is far lower than the number of neural populations that can potentially be contributing to the EEG signal, the problem is mathematically under-constrained, and additional constraints on the number of neural populations and their locations must therefore be introduced to obtain a unique solution.

For the purpose of analyzing EEG signals, the neural sources are treated as equivalent current dipoles. This is because the electric potentials stemming from the neural activity of a population of neurons will tend to look like the potential from a current dipole when recorded at a sufficiently large distance, as in EEG recordings. Source localization is therefore typically considered completed when the location of the current dipoles has been obtained. However, an exciting possibility is to try to go one step further and identify the type of neural activity that caused a localized current dipole. For example, the type of synaptic input (excitatory or inhibitory) to a population of neurons, and the location of the synaptic input (apical or basal) will result in different current dipoles (Ness et al., 2022). It has also been speculated that dendritic calcium spikes can be detected from EEG signals, which could lead to exciting new possibilities for studying learning mechanisms in the human brain (Suzuki & Larkum, 2017). Identifying different types of neural activity from EEG signals would however require knowledge of how different types of neural activity are reflected in EEG signals. Tools

for calculating EEG signals from biophysically detailed neural simulations have however recently been developed, and are available through the software LFPy 2.0 (Hagen et al., 2018; Næss et al., 2021). This allows for simulations of different types of neural activity and the resulting EEG signals, opening up for a more thorough investigation of the link between EEG signals and the underlying neural activity.

The past decade has seen a rapid increase in the availability and sophistication of machine learning techniques based on artificial neural networks, like Convolutional Neural Networks (CNNs). These methods have also been applied to EEG source localization with promising results. However, it has not been investigated if CNNs can also identify the neural origin of EEG signals, in addition to localizing neural sources. In this Master's thesis, the aim will be to investigate the possibility of using CNNs to not only localize current dipoles but also identify the neural origin of different types of neural activity, based on simulated data of different types of neural activity and the ensuing EEG signal.

Chapter 1

Introduction to Neuroscience

Neuroscience is a multidisciplinary field focused on understanding the complexities of the human brain and nervous system. At its core, neuronal communication forms the foundation for brain function, where billions of neurons interact through electrical signals. Electroencephalography (EEG) plays a pivotal role in recording and analyzing the electrical communication in the brain. Most importantly EEG serves as a non-invasive tool to detect abnormal brain activity and identify neurological disorders such as epilepsy. By exploring electrical brain activity, neuroscience aim to advance our comprehension of the human brain and improve diagnostic and therapeutic approaches for various neurological conditions.

In this introductory chapter, we will delve into the foundational aspects of neuronal communication, which will be usefull in chapter 2, where we will explore the principles of EEG recordings. By familiarizing ourselves with the fundamental concepts of neurons, we hope to gain a deeper understanding of the applications of EEG in the dynamic field of neuroscience. The chapter is based on the books "Neuronal Dynamics" by Gerstner, Kistler, Naud, and Paninski [1] and "Principles of Computational Modelling in Neuroscience" by Sterratt, Graham, Gillies, and Willshaw [2].

1.1 The Neuron

Neurons are the fundamental units of the central nervous system, forming intricate networks with numerous interconnections. Similar to other cells, neurons have a voltage difference across their cell membrane known as the membrane potential. This membrane potential is a result of the selective permeability of the cell membrane to different ions, particularly sodium (Na+), calcium (Ca2+), and chloride (Cl-). At rest, the neuron maintains a relatively higher concentration of sodium ions outside the cell and a higher concentration of potassium ions inside the cell. This difference in ion concentrations, along with the presence of ion channels that regulate the flow

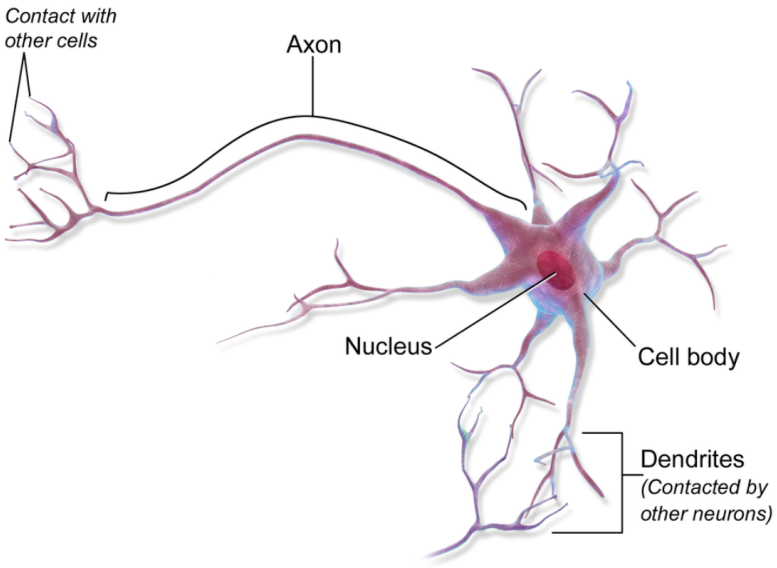

Figure 1.1: Illustation of single neuron with dendrites, soma and axon. The figure is adapted from ... Add proper citation and modify figure

of ions in and out of the cell, contributes to the resting membrane potential. Typically, the membrane potential of a neuron hovers around -65 mV, indicating that the interior of the cell is negatively charged compared to the external environment [2].

A neuron consists of three distinct parts: the *dendrites*, the *soma*, and the *axon*. Dendrites, with their branching structure, play an important role in collecting signals from other neurons. These signals are transmitted to the soma, which acts as the central processing unit, performing essential non-linear processing. If the total input received by the soma reaches a specific threshold, an *action potential* is initiated. This signal generates an electrical current that travels along the axon. When the electrical current reaches the *synaptic cleft*, chemical messengers known as *neurotransmitters* are realised. These messengers play the key role in transmitting the signal to the next neuron. If *receptors* on the receiving neuron accept the neurotransmitters, a new electrical signal is generated. This transmission of signals between neurons the synapses is called *synaptic input* [1].

In Figure **??**, we have provided a basic illustration of a single neuron with dendrites, soma, and axon.

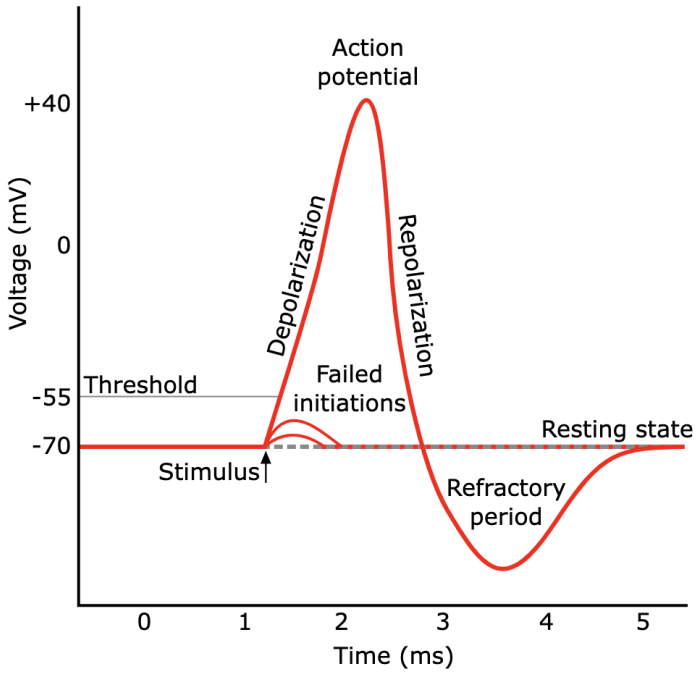

Figure 1.2: Illustation of a characteristic action potential. The figure is taken from ... Add proper figure text and citation

1.2 Spike Trains and Action Potentials

Fix illustation so that it does not contain depolarization, repolarization and refractory period?

When the electrical signals transmitted towards the soma reach the so-called threshold value, usually around -55 mV, the neuron fires. This initiation of an action potential, when observed in intracellular recordings, can be seen as a spike with an amplitude of about 100 mV and a duration of 1-2 ms [1]. In Figure 1.2, we have provided an illustration of the typical intracellular action potential.

The action potential is characterised by a swift and steep rise in the electrical potential, resulting in a rapid upward, positive spike, followed by a quick decline in the potential back to the resting state. This form of the action potentials remains relatively constant throughout the propagation along the axon. When collecting information out of these spikes, it is therefor not the shape of the spikes that is studied. Instead, the information lies in the number and timing of chains of action potentials emmitted by the neuron, also reffered to as *spike trains*.

Spike trains observed during epileptic seizures exhibit distinguishable characteristics compared to spike trains in regular neural activity. Epileptic

seizures result from bilateral synchronous firing of neurons and manifest as specific EEG patterns known as *spike-and-wave discharges*. These discharges exhibit repetitive and rhythmic patterns, typically occurring at a frequency of around 2.5 Hz, setting them apart from the irregular and asynchronous firing of action potentials seen in typical spike trains[1]. During epileptic events, the initiation of spike-and-wave discharges involves complex mechanisms, including the interplay of voltage-gated sodium and calcium channels and the role of inhibitory postsynaptic potentials [3]. Wikipedia citation. See if I can use Torbjørn book instead.

1.3 Anatomy of the Cortex

Fill in from sources, paper etc. https://www.cell.com/current-biology/pdf/S0960-9822(11)01198-5.pdf, https://www.ncbi.nlm.nih.gov/books/NBK2510/ Neurons in the brain are part of a vast network, interwoven with billions of other neurons and glial cells, creating the complex brain tissue. The brain is divided into various regions, and one essential area is the cortex, a thin but expansive sheet of neurons that folds over other brain structures. Different cortical areas have specific roles; some are specialized in processing sensory information, while others handle working memory or control motor functions [1].

The human cerebral cortex is typically divided into six layers. The oldest part of the cortex, known as the archipallium, has a more straightforward structure with three distinct neuronal layers. Within the archipallium, the hippocampus plays a major role in learning and memory functions. It is a crucial cortical structure implicated in the development of some common epilepsy syndromes [4].

Neurons communicate through synapses, where *presynaptic neurons* sends, and *postsynaptic cells* receives the electrical signals. In the animal brain, a single presynaptic neuron can connect to over 10,000 postsynaptic neurons. While many axonal branches end close to the neuron itself, some axons extend several centimeters to reach neurons in other brain regions [1].

Within the cortex, there are two primary classes of neurons. *Pyramidal neurons* send information to distant areas of the brain, and play a crucial role in long-distance communication. On the other hand, *interneurons* are considered local-circuit neurons, exerting their influence on nearby neurons. Most pyramidal neurons form excitatory synapses, meaning they stimulate post-synaptic neurons, while most interneurons form inhibitory synapses, meaning they suppress the activity of pyramidal cells or other inhibitory neurons [4].

Chapter 2

Electroencephalograpy

Electroencephalography is a prominent recording technique employed to capture the electrical activity of the cerebral cortex. This invaluable tool has significantly enriched our understanding of neuronal interactions and the organizational complexity of the brain. As one of the most widely utilized non-invasive methods in both neuroscience research and clinical practice, EEG has played a pivotal role in studying brain activity during diverse cognitive processes, facilitating disease diagnosis, and assessing functional connectivity.

In this chapter, our primary objectives are to explore the physiological basis of EEG signaling, shed light on the concept of the inverse problem in EEG, and introduce the use of head models to simulate realistic EEG measurements. Understanding the foundations of EEG and its methodologies will lay the groundwork when we further in this work will investigate the possibilities of using simulated EEG measurements to train an artificial *neural network* for the purpose of localizing the sources generating these signals. Remeber to mention difference between a neural network and an artificial one.

2.1 The Physiological basis of EEG signaling

EEG signaling? The roots of EEG trace back to the groundbreaking work of Hans Berger, who recorded the first human brainwave in 1924, marking the beginning of a new era in neuroscience research [3]. Since then, EEG has become an indispensable method, providing valuable insights into brain dynamics and functioning. EEG is a valuable tool that can be used to detect abnormalities in specific areas of the brain, aiding in the diagnosis of various brain disorders, including epilepsy, Alzheimer's disease, and brain tumors. By identifying distinct patterns of brain activity associated with these conditions, EEG has become an essential tool for early detection and treatment planning.

https://www.electrical4u.com/eeg-measurement/, include one more sentence about the amplifier bank. The EEG technique involves the use of small metal disks, known as *electrodes*, strategically placed on the scalp to simultaneously record electrical charges resulting from neuronal activity. Typically, each electrode can capture synchronous signals originating from an area of approximately 6 sq. cm. on the cortical surface [4]. These recording electrodes are then connected to individual wires, which, in turn, link to channel connectors leading to a differential amplifier bank. Figure 2.1 provides an illustration of the typical EEG measurement setup.

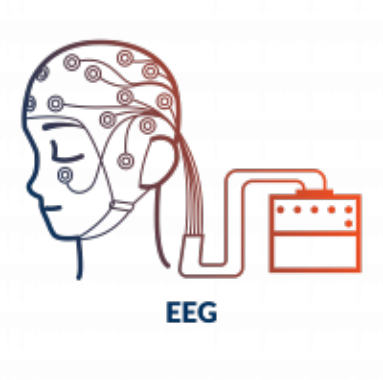

Figure 2.1: Illustration of the EEG method. Add citation.

Electrical potentials generated by individual neurons are far too small to be picked up by the recording electrodes. Therefore EEG measurements primarily reflect the summation of synchronous activity from thousands of pyramidal neurons with similar spatial orientation. The activity originating from neurons with different geometric alignments cannot be picked up because their individual electrical signals tend to cancel each other out, resulting in undetectable waves at the scalp electrodes [4].

The EEG is typically described in terms of rhythmic activity and transients, which are divided into frequency bands. Frequency bands are often extracted using spectral methods, and most of the cerebral signals observed in the scalp EEG fall within the range of 1–20 Hz. Abnormal activity can broadly be classified into *epileptiform* and *non-epileptiform* activity. Epileptiform activity typically arises in EEG recordings of patients with epilepsy and includes spikes and sharp waves, that can be seen synchronously throughout the entire brain. In this context, spikes refer to hypersynchronized bursts from a sufficient number of neurons, arising from high-frequency bursts of action potentials [4].

Detecting and localizing abnormal electrical patterns in EEG represents an important research pursuit. One of the fundamental aspects in this field is the *EEG inverse problem*, which aims to ascertain the spatial distribution of brain activity using potential measures acquired from scalp EEG recordings.

Further in this chapter, we will explore the concept of the EEG inverse problem in greater detail and examine its implications for source localization. But first we will dive into the subject of head models for the purpose of simulating neural networks and EEG recordings.

Should propably come after

2.2 The Inverse Problem and Source Localization

In the field of neuroscience, the inverse problem involves deducing the underlying neural activity responsible for a set of measured EEG data. In contrast to the *forward problem*, where known parameters are used to predict the resulting EEG potential, the inverse problem lacks a unique solution. This implies that different configurations of neural sources can produce the same EEG activity distribution on the scalp [5]. Is it then possible to reach a loss equal to 0? Needs to be explained that without restictions this problem lacks uniqe solution. multipole dipoles can sup up etc... but we make restirctions and places where dipoles can be detected.

The forward problem entails mathematically modeling of the relationship between neural current sources in the brain and the resulting EEG measurements on the scalp. This can be described as:

$$\Phi(t) = L \cdot p(t). \tag{2.1}$$

Here, Φ represents the vector of measured EEG signals at time t, $p(t)$ is the vector of a single cortical current dipole source at time t, and L is the *lead field matrix* that connects scalp electrode recordings with neural sources. While the full details of the lead field matrix will be explored later, for now, we envision it as a geometric arrangement linking the sensitivity of EEG measurements from diverse scalp locations to potential neural current sources within the brain.

Turning our attention to the inverse problem, its essence lies in estimating neural current sources in the brain using measured EEG data—essentially, the reverse of the forward problem. This relationship can be formulated as:

$$p(t) = L^{-1} \cdot \Phi(t), \tag{2.2}$$

where, L^{-1} denotes the inverse of the lead field matrix.

However, unlike the forward problem, the inverse problem lacks a unique solution due to its ill-posed nature. As a result, localizing the precise neural sources generating EEG signals becomes a demanding and statistically-driven endeavor. To address the complexities arising from numerous unknowns, methods such as neural networks are employed. Neural network can be designed to refine solutions to the inverse problem, providing robust and meaningful estimations of the neural sources responsible for the measured EEG data.

Before embarking on the utilization of artificial neural networks to solve the inverse problem, a substantial amount of appropriate EEG data is essential. This data can be obtained through simulated EEG data generated by *forward modeling*. Forward modeling deals with solving the forward problem, 2.1, and describes how the electrical activity in various regions of the human cortex gives rise to EEG signals recorded at the scalp electrodes. This process involves the use of *head models* that account for the conductivity of different tissues and the geometry of the head. These head models guide the simulation process, aiding in approximating real-world EEG recordings. head models do more than only account for the conductivity of different tissues and the geometry of the head.

2.3 Head Models

To accurately simulate EEG data and facilitate source localization, head models that precisely represent the conductivity distribution within the human head is essential. Head models serve as computational representations of the anatomical structure of the head, encompassing the brain, skull, cerebrospinal fluid, and scalp. They play a major role in the simulation of electrical signals originating from current dipoles, their propagation through various tissue compartments, and their impact on the recorded values at scalp electrodes.

Build up to the current dipole approximation? Next, Lead Fields?

Is this information needed? EEG signals are significantly influenced by the biophysical intricacies of the head. Notably, the cerebrospinal fluid exhibits a conductivity of approximately 1.7 S/m, while the skull and scalp possess conductivities of about 0.01 S/m and 0.5 S/m, respectively. These conductivity disparities underscore the necessity for comprehensive head models that consider such variations. Beyond conductivity, these models also account for the influence of tissue arrangement on EEG signals, such as whether a neuronal population resides within a *sulcus* or a *gyrus* [6]. By employing such a biophysically detailed head model, one can more accurately simulate the impact of various tissues on the distribution of extracellular potential. As a result, this model offers a heightened level of precision in representing EEG signals, leading to improved accuracy in EEG source localization solutions.

2.3.1 The New York Head Model

The New York Head model (NYHM), developed by the Biomedical Engineering Department at the City University of New York, is a highly detailed computer model tailored for simulating electrical brain activity, with an emphasis on EEG source localization. Grounded in high-resolution anatomical MRI data from 152 adult heads, this model allows the segmentation of six distinct tissue types within the head: scalp, skull, cerebrospinal fluid, gray

matter, white matter, and air cavities. Its high level of detail and accuracy makes it an excellent tool for simulating and comprehending brain activity in a realistic manner. Presenting a three-dimensional representation of the head and brain, coupled with precise information about tissue geometry and electrical properties, this head model will be utilized in our work within the context of EEG simulation to generate the data used for dipole source localization.

To precisely calculate EEG signals recorded at various scalp locations, the New York Head Model (NYHM) relies on a fundamental principle known as the *reciprocity theorem*. This theorem states that if you swap the positions of a dipole source and a measurement point, the measured value for the potential will remain the same. We calculate what is called the *lead field* by simulating a current dipole source at an electrode location and computing the resulting potential distribution at different points within the cortex. The lead field values are thus the EEG signals that would be measured at the electrode location when a source is positioned at one of the points in the cortex [6]. This turns out to be more efficient than simulating the EEG signals that would be measured at the electrode positions from the electrical signature of a single dipole within the cortex. The efficiency of this approach becomes evident due to the reduced computational demands when simulating EEG signals measured at recording electrodes. Specifically, simulating the propagation of the electrical field from a single dipole location, propagating through the densely distributed cortex and reaching the recording electrodes, requires tens of thousands of calculations, corresponding to the vast number of dipole locations within the cortex. In contrast, simulating the electrical field from the recording electrodes through the cortex requires calculations equal to the number of recording electrodes, which is significantly smaller, thus greatly simplifying the computational load.

This flipped approach is more efficient? Potential vs. electric field.

To construct the NYHM's lead field matrix (L), the researchers behind the model have performed these calculations for 231 electrode positions on the scalp and 70,000 possible dipole source locations within the cortex. This lead field matrix establishes the mathematical relationship between a current dipole moment within the brain cortex and the resulting EEG signals. Each row of the matrix corresponds to the Mathematically, the lead field matrix is defined as:

$$L_{ij} = \frac{E_j^{(i)}}{I}, \quad \text{for } j = x, y, z. \tag{2.3}$$

Here, I represents the injected current at the electrode locations, and $\mathbf{E}^{(i)} = (E_x^{(i)}, E_y^{(i)}, E_z^{(i)})$ is the resulting electric field in the brain at . The row index i spans from 1 to $231 \cdot 70,000$. This use of the reciprocity principle greatly simplifies computations because it means that solutions to the forward problem are precomputed for every current dipole location. Consequently, the

EEG signal Φ, which encapsulates the values measured at each of the recording electrodes resulting from a current dipole, can be calculated through a straightforward matrix multiplication:

$$\Phi = (\phi_0, \ldots, \phi_{70,000})\Phi = L\mathbf{p}, \tag{2.4}$$

where the current dipole moment $\mathbf{p} = (p_x, p_y, p_z)$ holds the dipole's magnitude in the x, y, z-directions respectively.

For further comprehensive details about the New York Head model, we refer readers to Huang, Parra, and Haufe (2016) [7].

2.4 The Current Dipole Approximation

By accurately simulating EEG data, non-linear optimization algorithms such as neural networks can be utilized for solving the EEG inverse problem. However, the simulation of intricate neuronal dynamics is a computationally expensive and complex task. Not correct. Head models are always based on dipole approximation, and so are lead field matrices. An approximation that address this challange and simplifies the simulation phase is the *current dipole approximation*. This approximation is rooted in the observation that the neuron's contribution to the extracellular potential V_e becomes increasingly more dipole-like with an increasing distance.

The key insight behind the current dipole approximation lies in the *multipole expansion*, a technique that comes to our aid when the recording point is situated at a significant distance from the source distribution. While electrical charges within neural tissue can give rise to the formation of current multipoles, the multipole expansion theorem offers a method to express the extracellular potential, denoted as $V_e(R)$, in terms of various pole contributions. In the context of EEG signals, employing this theorem reveals that $V_e(R)$ can be effectively approximated by a single dipole [8]. Wikipedia has also been used here. Check with book that everything is correct.

Neuronal multipoles depend on the spatial arrangement and symmetry of the charge distribution and result from the interplay of current sources and sinks [9]. The expression for the extracellular potential associated with different multipole orders may take complicated forms, and be hard to interpreted. However, when the distance R from the center of the source to the recording point surpasses the extent of the source, the applicability of multipole expansion becomes evident [10] is this correct?. This expansion are often beneficial as usually only the first few terms are needed in order to provide an accurate approximation of the original funtion, as we will se now. Using the multipole expansion theorem, the representation of the extracellular potential $\phi(R)$ takes the form:

$$V(R) = \frac{C_{\text{monopole}}}{R} + \frac{C_{\text{dipole}}}{R^2} + \frac{C_{\text{quadrupole}}}{R^3} + \frac{C_{\text{octopole}}}{R^4} + ... \tag{2.5}$$

where the numerators represents the contributions to the extracellular potential. The terms denoted C_{monopole}, C_{dipole} and $C_{\text{quadrupole}}$ represents contributions to the extracellulat potential, V_e, and can in general be extremely complicated as they depend on the relationship between radial coordinates and symmetry of the current source and measurement electrode. We note that the contributions beyond the dipole term decay more rapidly with distance R. This means that in scenarios where we are considerably distant from the source distribution, the higher-order terms become negligible, leaving us primarily with the monopole and dipole contribution. This brings us to another interesting observation: The monopole contribution vanishes, stemming from the fact that the net sum of currents over a neuronal membrane is invariably zero. Consequently, we are left with an approximation of the extracellular potential, V_e, that relies solely on the dipole contribution:

$$V_e(\mathbf{r}) \approx \frac{C_{\text{dipole}}}{R^2} = \frac{1}{4\pi\sigma}\frac{|\mathbf{p}|\cos\theta}{|\mathbf{r}-\mathbf{r}_p|^2}. \tag{2.6}$$

Here we have substituded for C_{dipole} in terms of other properties. Here p symbolizes the current dipole moment within a medium of conductivity σ. The distance between the current dipole moment at $\mathbf{r}_p$ and the electrode location $\mathbf{r}$ is denoted as $R = |\mathbf{R}| = |\mathbf{r}-\mathbf{r}_p|$. Additionally, θ signifies the angle between $\mathbf{p}$ and $\mathbf{R}$. This equation is recognized as the dipole approximation and stands as a reliable method for calculating the extracellular potential, particularly when the distance R significantly surpasses the dipole length. This condition is inevitably met in EEG studies, as the dipole length cannot exceed the depth of the cortex, while EEG electrodes are typically positioned several centimeters away from the cortical surface on the scalp [6].

Consequently, we have connected the concept of multipole expansion with the validity of the dipole approximation. Through multipole expansion, we can comprehend the intricate extracellular potential in terms of various contributions, and by carefully considering their behavior, we arrive at the precise dipole approximation.

In Figure 2.2, we have provided a simulation of the extracellular potential generated by a neuron in response to a single synaptic input, where the spatial distribution of membrane current was explicitly taken into consideration. The Figure has been collected from work done by Torbjørn Ness and Gaute Einevoll. This simulation aligns with the dipole approximation, as it clearly visualizes the distribution of electric charge in the extracellular potential of the neuron's surroundings, revealing distinct dipole patterns when observed from a greater distance.

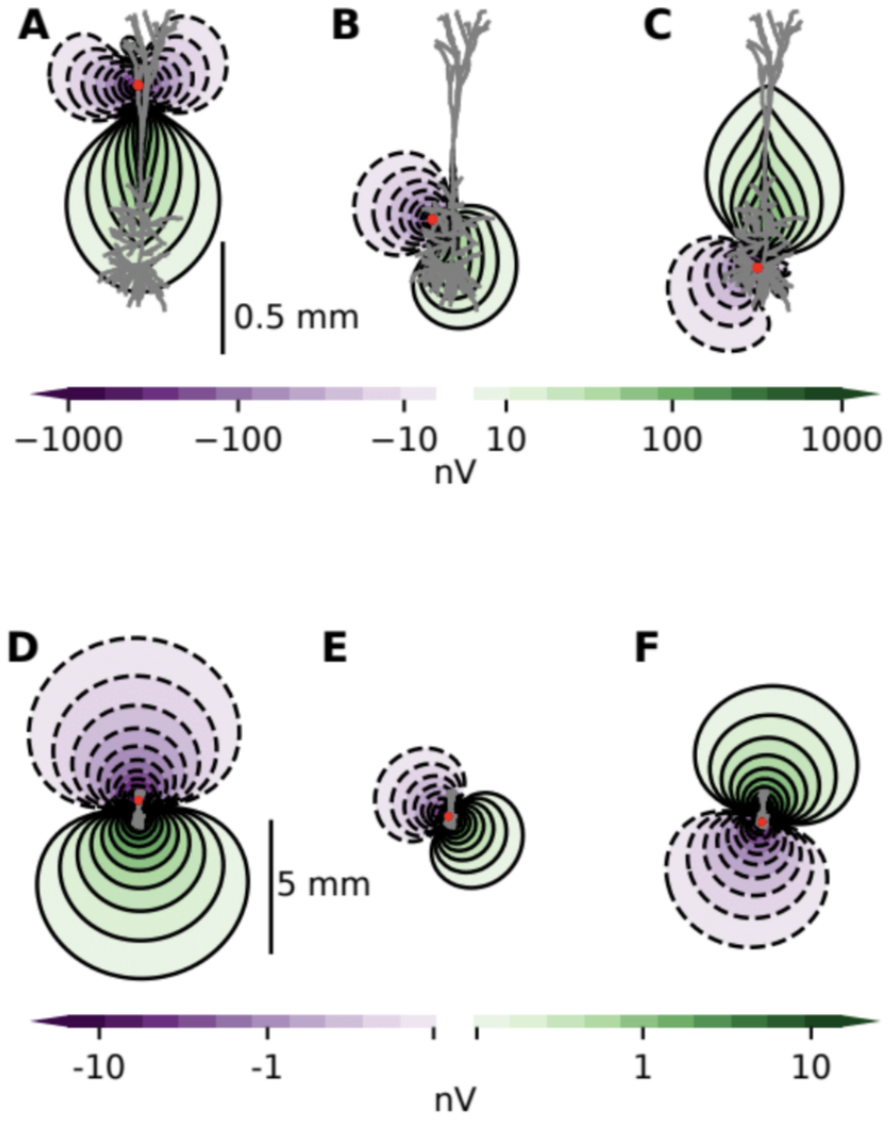

Figure 2.2: Simulation of extracellulat potential showing distinct dipole pattern. The figure has been provided from my supervisiors Torbjørn Ness and Gaute Einevoll.

2.5 The Effect of Dipole Orientation

Figure 2.3 and 2.4 are borrowed from work done by Tornjørn Ness and Gaute Einevoll, and illustrate the impact of dipole orientation on EEG outcomes. Figure 2.3 represent the EEG signals obtained from two manually selected dipole locations within the New York head model. These dipoles are situated in a gyrus and a sulcus, respectively, and exhibit distinct EEG patterns. In general, the contribution of an individual current dipole to the EEG signal is maximized when the dipole is perpendicularly situated within a gyrus, as depicted in Figure 2.3B. Contrastingly, when a dipole is placed in a sulcus with a perpendicular orientation, a significant EEG contribution may still be observed, however unlike the dipole in the gyrus, it exhibits a more dipolar pattern, as shown in Figure 2.3C.

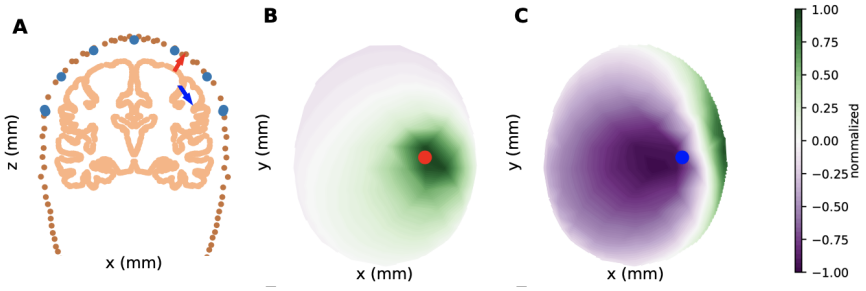

Figure 2.3: A: Two selected dipole locations in the New York head model: one in a gyrus (red) and one in a sulcus (blue). The head model is viewed from the side (x, z-plane). Close to the chosen cross-section plane, EEG electrode locations are marked in light blue. Available dipole locations near the cortical cross-section form an outline of the cortical sheet and are marked in pink. The current dipole moment for all cases was 10^7 nAμm. B: Interpolated color plot of EEG signal from the gyrus dipole, viewed from the top (x, y-plane). The plotted EEG signal is scaled, with a maximum value of 1.1 μV. C: Interpolated color plot of EEG signal from the sulcus dipole. The plotted EEG signal is scaled, with a maximum value of 0.7 μV. This Figure is borrowed from work done by Torbjørn Ness and Gaute Einevoll [6].

Further, Figure 2.4 depicts the EEG signals from identical dipoles positioned in various folding patterns of the cortical surface. These patterns align with the previous observations, as they showcases that the orientation of the current dipole moment significantly influences the EEG outcome. Firstly, Figure 2.4A and 2.4C provide an expanded illustration of the aforementioned scenarios, incorporating additional dipole moments located in a gyrus and a sulcus, respectively. In Figure 2.4B, where a collection of dipoles points randomly upwards and downwards, the EEG signal contribution appears to diminish significantly. Conversely, when the dipoles align in the depth dir-

ection of the cortex and are distributed across both gyrus and sulcus, we can expect an EEG contribution in between what we saw from Figure 2.4A and 2.4B, as depicted in Figure 2.4D. Lastly, Figure 2.4E demonstrates the minimal EEG contribution observed when the dipoles are divided between two opposing sulci.

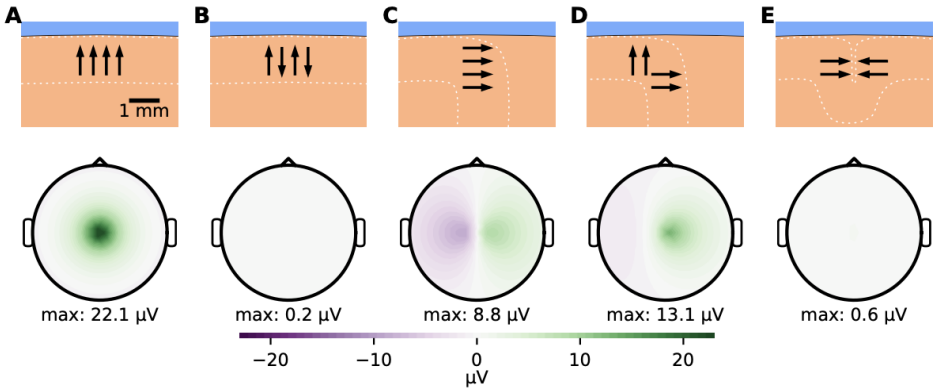

Figure 2.4: Different folding patterns of the cortical surface are represented by white dashed lines. EEG signals are calculated from four identical current dipoles with varying orientations. A: Dipoles aligned in the same direction within a gyrus. B: Dipoles pointing in opposite directions within a gyrus. C: Dipoles aligned in the same direction within a sulcus. D: Dipoles distributed between a gyrus and a sulcus, pointing towards the cortical surface. E: Dipoles divided between opposing sulci, pointing towards the cortical surface. Each panel features a dipole moment magnitude of 10 nAm, and the dipoles are positioned at the centers of the arrows in the top row. This Figure is borrowed from work done by Torbjørn Ness and Gaute Einevoll [6].

2.6 Solving the EEG inverse problem

In this chapter, we have explored the fundamental concepts that underpin the investigation of the inverse problem in EEG source localization. With this groundwork in place, we now turn our attention to the subsequent chapters, where we transition from theory to application.

In the next chapter, we delve into practical implementation by simulating EEG data. Utilizing the New York Head Model and current dipole approximation, we create synthetic EEG data that closely resembles real-world scenarios. This simulated data, generated using the principles discussed in this chapter, serves as a crucial foundation for the subsequent chapters. It brings us a step closer to addressing the EEG source localization challenge.

Chapter 4 explores the methods of machine learning and neural networks. Building upon the simulated EEG data, we construct a sophisticated neural

network designed to address the inverse problem efficiently. By the use of computational techniques, our aim is to bridge the gap between theoretical understanding and real-world applications.

Chapter 3

Creating EEG Data

In preparation for the application of neural networks to address the inverse problem, the acquisition of a substantial and appropriate EEG dataset is essential. This chapter focuses on utilizing the New York Head model in conjunction with the current dipole approximation to construct biophysically realistic EEG data.

3.1 Simulation of EEG Signals from single dipoles

Include range of x, y, z values and maybe eeg ? The New York Head model and its lead field matrix are integrated into the Python module LFPy. Within LFPy, we use the `NYHeadModel` class to calculate EEG signals originating from a current dipole moment p. The current dipole moments of the LFPy package are expressed in terms of nAμm, while the EEG signals derived from the NYHM are recomputed into units of mV. For more information about the LFPy module, we refer the reader to Hagen, Næss, Ness and Einevoll (2018) [11].

The lead field matrix of the NYHM comprises 74,382 discrete points, each corresponding to possible locations for dipole sources. In the context of simulating EEG measurements, the procedure commences with the random selection of positions from these points to serve as the locations for placing dipoles. Each simulated EEG sample entails a solitary dipole positioned at one of the randomly chosen locations. As the primary objective is to address the inverse problem, maintain uniform magnitudes for the dipole signals, as their variation is not of primary concern. By setting these magnitudes to 10^7 nA μm, the resulting EEG measurements span a range of approximately -1 to 1 μV.

Vise tilbake til forrige kapittel med dipolorientering? To ensure that the dipole orientations are predominantly aligned with the depth direction of the cortex, a rotation procedure is employed for each dipole moment, orienting it perpendicular to the cerebral cortex. Note that aligning the dipole

orientations perpendicularly to the cortex does not always lead to the dipole's normal vector pointing directly outward toward an EEG electrode. This variability arises due to the complex folding patterns found in the human cortex. In certain cases, when a dipole is located within a sulcus, the electrical signal it generates may follow the contour of the sulcus and enter deeper into the cortex. Consequently, the EEG signal must traverse a more extended path before ultimately reaching an EEG electrode [6].

The NYHM generates EEG signals as time series data, reflecting the structure of real-world measurements. As a result, the inherent format of EEG data deviates from a one-dimensional representation, instead adopting a matrix configuration of 231 times 1601 dimensions. In this arrangement, 231 values represent measurements from scalp recording electrodes, with 1601 time steps marking the temporal progression. However, as mentioned in the preceding chapter, EEG analysis often centers on specific frequency components within each temporal instance of measurement. This practice effectively reduces the multidimensional EEG data and eliminates less relevant data points.

In our analysis, we have adopted a one-dimensional data approach to pinpoint sources of neural activity. This discrepancy from conventional practices, which involve extracting diverse frequency spectra and analyzing time series to identify anomalies, offers simplification and computational efficiency. Additionally, this transition is well-suited to our simulated data, which lacks confusing signals from brain activity or unclear background noise, simplifying the data preparation before being fed into a neural network. Rather than conducting a time series analysis with frequency extraction, we focus on data from a static dipole representing a chosen time point. This approach results in a one-dimensional EEG signal that effectively encapsulates insights into the spatial distribution of EEG patterns and their relationship with specific dipole source locations within the cerebral cortex. Add plot of NYHM timeseries data?

3.2 Noise

Experimental EEG recordings inevitably contain noise, which can interfere with the accurate analysis of brain activity. *Artifacts*, which are signals recorded by EEG but originating from sources other than neuronal communication, pose a particular challenge in real-world data. Some artifacts can mimic genuine epileptiform abnormalities or seizures, underscoring the importance of identifying and distinguishing them from true brain waves [12].

Artifacts can be classified into two categories based on their origin. *Physiological artifacts* arise from the patient's own physiological processes, including ocular activity, muscle activity, cardiac activity, perspiration, and

respiration. *Technical artifacts*, on the other hand, originate from external factors such as cable and body movements or electromagnetic interferences [13].

Filtering techniques are commonly employed to remove artifacts from EEG recordings prior to analysis. However, in the case of simulated EEG data, the need for artifact removal is eliminated as the data inherently do not contain noise. Simulated EEG data can be considered as pre-filtered and preprocessed, ensuring a high signal-to-noise ratio (SNR) [14]. Nevertheless, to ensure the data aligns with real-world scenarios and accounts for other technical considerations which we will come back to later, it is necessary to introduce noise to the data before feeding it into the neural network.

In our approach, we recognize that the introduction of noise to the simulated EEG data is an essential step to enhance the robustness of the trained neural network and ensure its ability to handle real EEG recordings effectively. Since the specific characteristics and quantity of noise have not been the primary focus of our study, we have opted for a straightforward approach. Our final dataset incorporates normally distributed noise with a mean of 0 and a standard deviation equal to 10% of the standard deviation observed in the simulated EEG recordings. By introducing this noise, we introduce random variations around each data point while preserving the overall normalization properties of the dataset.

3.2.1 Final Dataset

The final dataset comprises 70, 000 rows, where each row corresponds to a single sample or fictive patient. Within the dataset, there are 231 columns representing the features, which denote the EEG measurements recorded at each electrode - a configuration directly derived from the NYHM. In practice, the data consists of two separete files holding pairs of EEG data and corresponding target data, where x-, y- and z coordinates of different dipole sources are the answer keys.

Figure 3.1 presents an example of the input EEG data for a single sample, with 10% noise added. The EEG result obtained from the specific sample contains a solitary current dipole source positioned randomly within the cerebral cortex. The prominent dipolar pattern in the figure indicates that the dipole is located within a sulcus. In the figure the EEG measure is visualized from multiple perspectives, including the x-z plane, y-z plane, and the x-y plane. The electrode locations are represented by filled circles, with the color of the fill indicating the magnitude of the measured signal at each electrode. The position of the current dipole moment is denoted by a yellow star. As indicated by the colorbar in the figure, the EEG signal for the specific sample ranges from -1 to 1 μV, which is the range that the simulated EEG data for all samples will fall within.

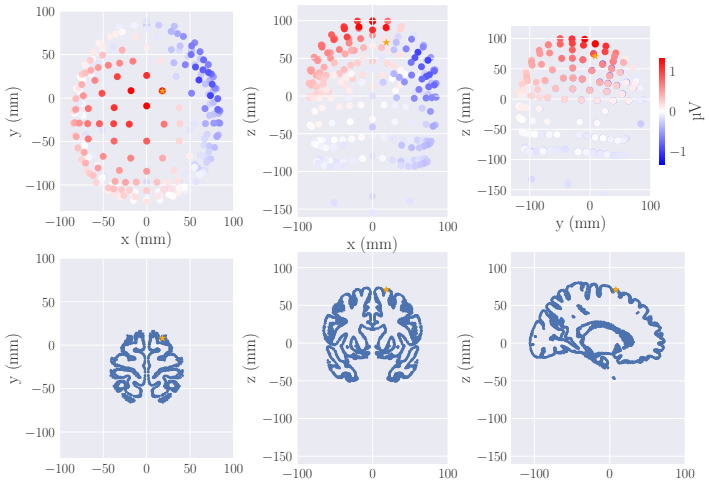

Figure 3.1: EEG for a sample containing one single current dipole source at a random position within the celebral cortex. As for all samples within the data set, 10 percent of normally distributed noise has been added to the original signal. The EEG measure is seen from both sides (x-, z-plane and y-, z-plane) and above (the x-, y-plane). EEG electrode locations are presented as filled circels, where the color of the fill represents the magnitue of the measured signal for the given electrode. The position of the current dipole moment is marked with a yellow star.

Chapter 4

DiLoc - A Neural Network Apporach for Source Localization

When having enough and suitable data for a problem to solve, one can start building tailorsuited neural networks to address the problem. We aim to build a neural network that maps measured EEG signals to the corresponding location of current dipoles. In this chapter, we will provide a comprehensive overview of the feed forward neural network that has been build to solve the EEG inverse problem, reffering to it as *DiLoc*. We will be discussing its architecture and parameters, and moreover present an alternative approach using a convolutional neural network for the same purpose of source localization. Convolutional neural network is not included yet. Unsure if it should be in this chapter or chapter with CCN method and results.

scaling

4.1 Machine Learning and Neural Networks

Machine learning (ML) is a field concerned with constructing computer programs that learn from experience, where the utialization of data improves computer performance across various tasks. Within this broad scope, one application could lie in the identification of sources generating abnormal electrical brain signals, as we will be performing. By employing specific machine learning algorithms, EEG data can be processed and analyzed to accurately localize the sources responsible for the recorded signals. These algorithms should be able to learn from input data and uncover patterns that associate the signals with corresponding sources, effectively solving the EEG inverse problem.

According to Mehta et. al. a definition of machine learning could be "...a subfield of artificial intelligence with the goal of developing algorithms

capable of learning from data automatically" [15]. The typical ML problems are addressed using the same three elements. The first element is the dataset $\mathcal{D} = (\mathbf{X}, \mathbf{y})$ where $\mathbf{X}$ commonly is refered to as the *design matrix*, and consists of independent variables, and $\mathbf{y}$ is a vector consisting of dependent variables. Next, we have the model itself, $f(\mathbf{x}; \boldsymbol{\theta})$. The ML model can be seen as a function used to predict an output from a vector of input variables, i.e. $f : \mathbf{x} \rightarrow y$ of the parameters $\boldsymbol{\theta}$. Finally, the third element, allows us to evaluate how well the model performs on the obervations $\mathbf{y}$. This element is known as the cost funtion $\mathcal{C}(\mathbf{y}, f(\mathbf{X}); \boldsymbol{\theta})$.

4.1.1 Neural Networks

In order to solve the inverse problem we will be building the DiLoc Neural Network. Neural networks are a distinct class of so-called *nonlinear machine learning models* capable of learning tasks by observing examples, without requiring explicit task-specific rules [16]. The models mimics the way bilogical neurons trasmit signals, with interconnected *nodes* that communicate through mathematical functions across layers. The layers in neural networks contain an optional number of nodes, where each connection is represented by a *weight* variable. In the context of neural networks, these weight values can be understood as parameters representing the strenght of communication between nodes. Changing these weight values determines how strongly or weak a nodes's output influences another noded's input. During training, the neural network learns to adjust these weights so to capture the underlying patterns and relationships in the data. The network is able to do so by using past experiences known as *training examples*. These patterns are further updated by the usage of appropriate non-linear functions, known as *activation function*, and finally presented as the output [17]. A neural network consits of many nodes stacked into *layers*, with the output of one layer serving as the input for the next. Typically, the neural networks are built up of an input layer, an output layer and layers in between, called *hidden layers*. In figure 4.1 we have provided the basic architecture of neural networks. Here nodes are depiced as circular shapes, while arrows indicate connections between the nodes.

The behaviour of the human brain has inspired the following simple mathematical model for an artificial neuron, or node:

$$a = f\left(\Sigma_{i=1}^{n} w_i x_i)\right) = f(z), \tag{4.1}$$

where a is the output of the node, and is the value of the nodes activation function f which has as input a weighted sum of signals $x_i, x_{i+1}, ..., x_n$ recieved by n other nodes, multiplied with the weights $w_i, w_{i+1}, ..., w_n$ and added with biases $b_i, b_{i+1}, ..., b_n$. Biases, represented by b_i for each node, provide an additional adjustment to the weighted sum, allowing the network to model biases in the data and play a crucial role in the overall flexibility

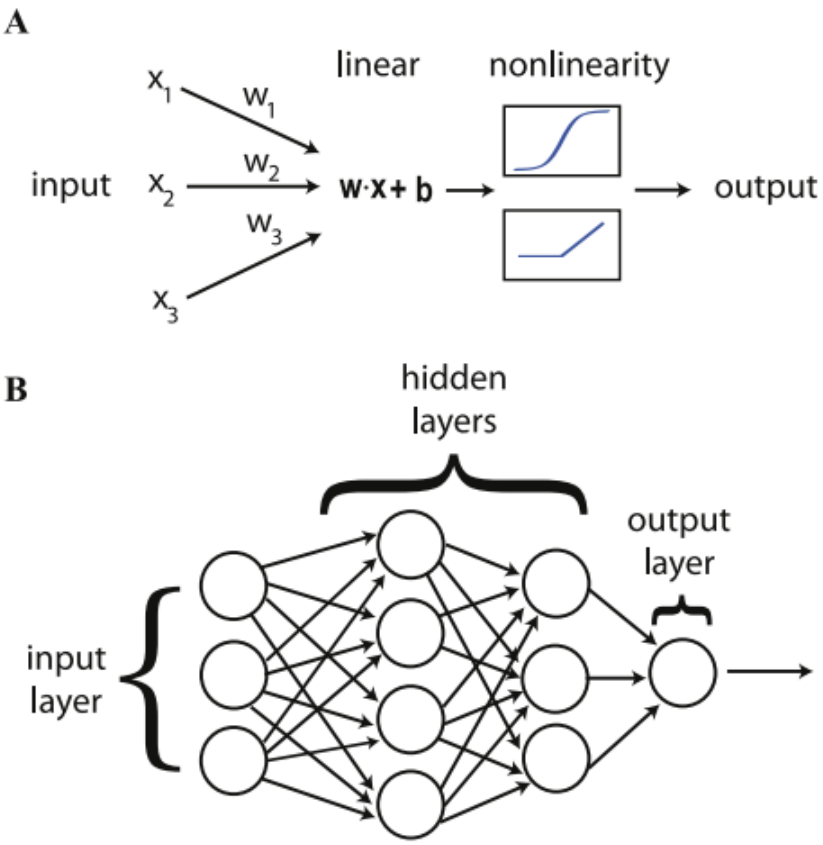

Figure 4.1: **(A)** The fundamental structure of neural networks comprises simplified nodes units that perform a linear operation to assign different weights to inputs, followed by a non-linear activation function. **(B)** These nodes units are organized into layers, where the output of one layer serves as the input to the subsequent layer, forming a hierarchical arrangement. Add proper citation.

of artificial neurons. The exact expression of a varies depending on the type of non-linearity that exists in the activation function applied to the input of each node. However, in almost all cases a can be decomposed into a linear operation that weights the relative importance of the various inputs, and a non-linear transformation $f(z)$. As seen in equation 4.1, the linear tranformation commonly takes the form of a dot product with a set of node-specific weights followed by re-centering with a node-specific bias. A more convenient notation for the linear transformation z^i then goes as follows:

$$z^i = \boldsymbol{w}^{(i)} \cdot \boldsymbol{x} + b^{(i)} = \mathbf{x}^T \cdot \mathbf{w}^{(i)}, \tag{4.2}$$

where $\mathbf{x} = (1, \boldsymbol{x})$ and $\mathbf{w}^i = (b^{(i)}), \boldsymbol{w}^{(i)})$. The full input-output function can be expressed by incorporating this into the non-linear activation function f_i, as expressed below:

$$a_i(\mathbf{x}) = f_i(z^{(i)}). \tag{4.3}$$

Not sure if all this information is necessary.

4.2 The creation of DiLoc

The development of DiLoc started with a deliberate and cautious approach, focusing on simplicity without compromising on accuracy in tackling the inverse problem. As a natural starting point, we adopted a fully connected, feed-forward neural network architecture, which eventually proved to be the most suitable framework for our purposes. The feedforward neural network (FFNN) was one of the first artificial neural network to be adopted and is yet today an important algorithm used in machine learning. The feed forward neural network is the simplest form of neural network, as information is only processed forward, from the input nodes, through the hidden nodes and to the output nodes.

4.2.1 Architecture

The determination of the optimal number of hidden layers and neurons was meticulously executed through an iterative trial-and-error procedure. Various network configurations, including small, medium, and large architectures, were systematically examined. Ultimately, we settled on the medium-sized network configuration. This selection, combined with considerations of additional network attributes, which will be elaborated upon later in this chapter, yielded the most promising results in terms of prediction accuracies.

The input layer is designed with 231 neurons, corresponding to the number of features in our dataset, i.e. the number of recording electrodes for each sample. Subsequently, the network consists of five hidden layers, comprising 512, 256, 128, 62, and 32 neurons, respectively. Finally, the output

layer holds an x-, y- and z-coordinate representing the predicted position of the desired dipole source. Figure 4.2 visualizes the construction of the fully connected neural network.

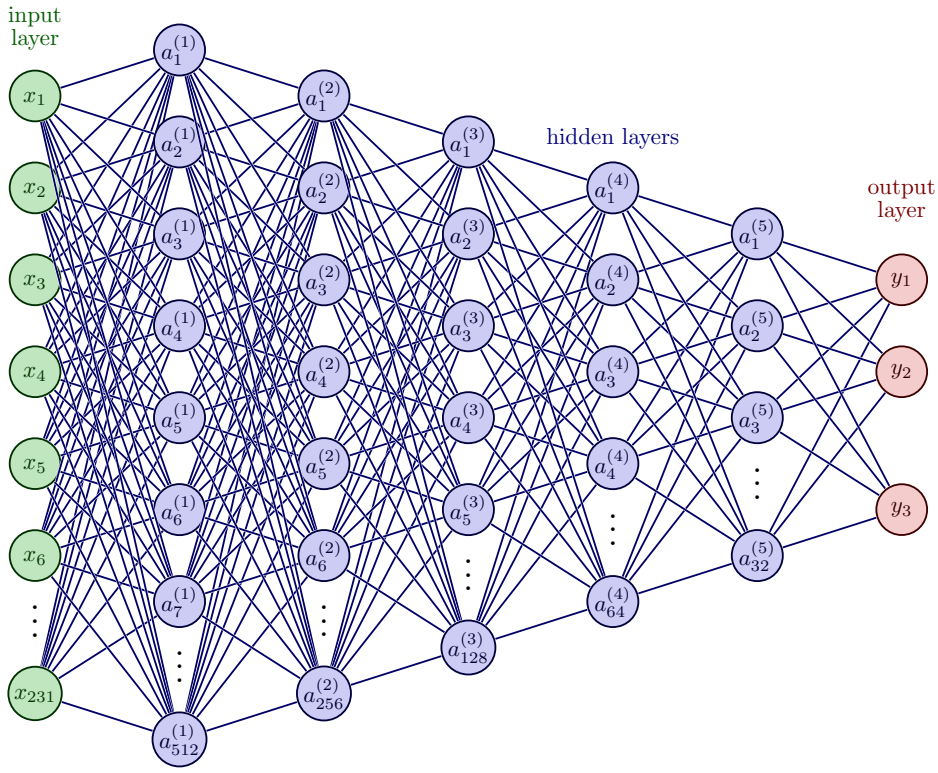

Figure 4.2: Architecture.

4.2.2 Activation Functions

Activation functions are fundamental components within the architecture of neural networks, serving to transform input signals into meaningful outputs. Essentially, these functions introduce nonlinearity into the network's computations, thereby enabling the network to capture and model complex, nonlinear relationships within data [18]. This property is essential because many real-world phenomena and data patterns exhibit inherent nonlinearity. Without activation functions, neural networks would essentially be linear models, capable only of representing linear relationships between inputs and outputs. While linear transformations occur within individual nodes through the weighted sum of inputs, the introduction of non-linear activation functions allows neural networks to capture complex relationships and patterns. These functions are applied at every artificial neuron in the hidden layers and in the output layer [19]. In the context of solving the EEG inverse prob-

lem, where the EEG data contains intricate, nonlinear patterns, activation functions empower DiLoc to effectively model and learn from the EEG data structures.

Drawing inspiration from the behavior of biological neurons, activation functions can be understood as decision-makers within the network, determining which information should be relayed to the next artificial neuron. This process is analogous to biophysics, where the axon of one cell takes the output signal from the preceding cell and converts it into a format suitable for input to the next cell. Some activation functions can also be directly associated with biological phenomena like action potentials and spikes within neurons. Similar to how real neurons respond to incoming electrical signals, activation functions decide whether a node in a neural network should be activated or not based on the strength of the input it receives. If the input exceeds a certain threshold, the artificial neuron "fires"; otherwise, it remains inactive [20].

Rectified Linear Unit

Within the input layer of the DiLoc network, nodes utilize the *Rectified Linear Units* (ReLU) activation function. ReLU closely resembles the behavior of biological neurons. Specifically, it echoes the concept of action potential: if a threshold is reached, the neuron fires, otherwise, the neuron remains inactive. For the ReLU function, this means positive input values remain unchanged, while negative input values are suppressed by setting them to zero.

Mathematically, the ReLU function is defined as:

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases} \tag{4.4}$$

The widespread adoption of ReLU in neural networks is attributed to its computational speed, performance, and generalization capabilities [21]. In contrast to the Hyperbolic Tangent activation functions, to which we will return shortly, ReLU offers a more straightforward mathematical representation. As seen in Figure 4.3, the ReLU function maintains the input value when positive and outputs zero for negative inputs. This behavior promotes computational efficiency, as at any instance, only a subset of neurons activate.

By using ReLU in our first layer we introduces non-linearity into the model and enables it to capture complex relationships and patterns within the EEG data effectively.

Here I need to explain pros with ReLU in fist layer. Include the problem of dead neurons.

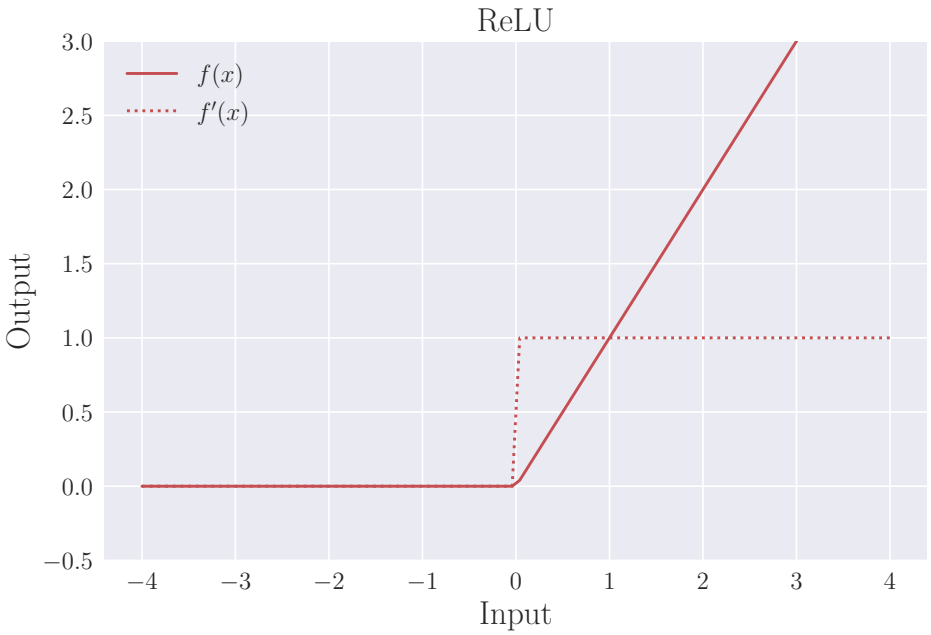

Figure 4.3: ReLU tangent activation function.

Hyperbolic Tangent

For the hidden layers, we employed another actication function, known as the *Hyperbolic Tangent* (tanh). This decision was driven by its ability to compress input values into a range between -1 and 1, which helps prevent activations from becoming extremely large and potentially unstable during training. By constraining the activations within this range, we aim for DiLoc to achieve a stable and effective learning process.

Tanh is continuous and differentiable at all points, that for reasons we will come back to is making it a suitable activation function for neural networks Where do I come back to this?. Its mathematical representation is as follows:

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{4.5}$$

As seen in Figure 4.4, the tanh function smoothly squashes input values into the desired range of -1 to 1, providing a more gentle activation compared to ReLU. While ReLU is known for its computational speed and simplicity, Tanh's bounded output ensures that activations remain within a controlled range.

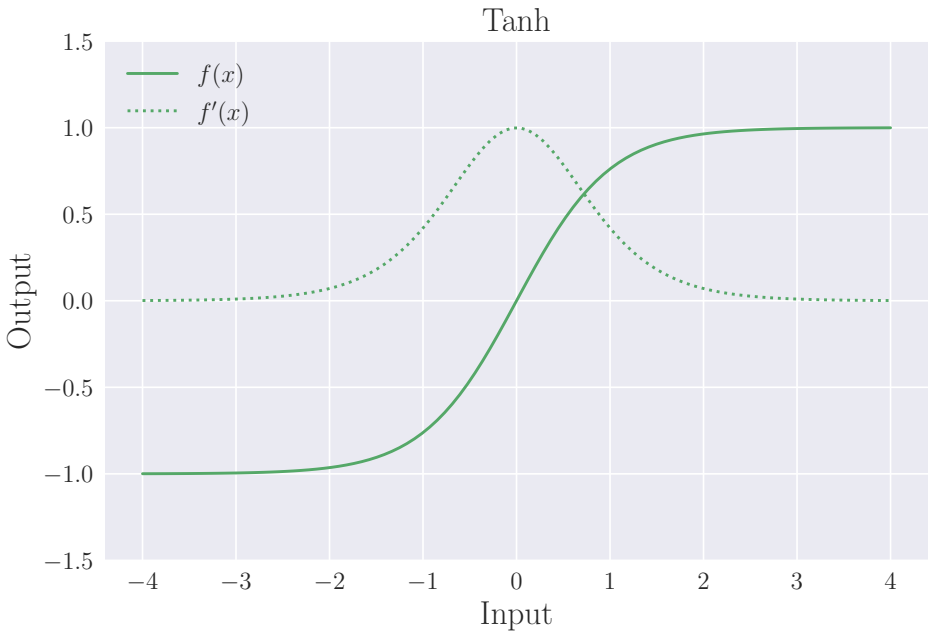

Figure 4.4: Hyperbolig tangent activation function.

Linear Activation

Last Layer Linear Avtivation? DiLoc aims to provide direct and unconstrained predictions of the x-, y-, and z-positions for a desired dipole source. In our specific application, these target positions exhibit a range of values, with x ranging from -70 to 70 mm, y from -58 to 78 mm, and z from -69 to 59 mm. This wide range of potential values underscores the need for a regression approach that allows for unbounded and continuous predictions. When dealing with regression problems, linear activations, or rather, no activation function at all, are commonly utialized.

Therefore, in contrast to ReLU and tanh activations used in earlier layers, we adopted linear transformations without the use of an activation function for the nodes within the last layer of DiLoc. This decision aligns with the requirements of our specific task, ensuring that the output of the network is not constrained within a specific range.

Include info about exploding and vanishing gradient ?

4.2.3 Initialization

All inner parameters within a neural network are adjustable. *Initialization* of the parameters of weights and biases is an important process used within neural networks. The choice of these initial values can significantly impact how quickly the model converges during training and how well the model

generalizes to unseen data.

There are several techniques for initialization in neural networks. In DiLoc, we have utilized the Xavier initialization, as this initialization commonly is paired with the tanh activation function. For every layer l, the weights $W^{[l]}$ and biases $b^{[l]}$ are sampled as follows:

$$W^{[l]} \sim \mathcal{N}(\mu = 0, \sigma^2 = \frac{1}{n^{[l-1]}}) \tag{4.6}$$

$$b^{[l]} = 0 \tag{4.7}$$

In this approach, the weights of layer l are drawn from a normal distribution with a mean (μ) of 0 and a variance (σ^2) of $\frac{1}{n^{[l-1]}}$, where $n^{[l-1]}$ signifies the number of inputs to the neuron. Additionally, biases are initialized to zero. This practice aligns with the idea that biases represent the initial influence of each neuron before any data is processed, making zero initialization a reasonable starting point.

The use of activation functions like tanh can introduce challenges related to vanishing and exploding gradient issues. Xavier initialization addresses these concerns by ensuring that the variance across layers is proportional to $\frac{1}{n^{[l-1]}}$, thereby contributing to the stability of the learning process. Essentially, Xavier initialization strikes a balance in parameter initialization, preserves variance, and mitigates the risk of gradients vanishing during training. This makes it a well-suited choice for DiLoc, where both stability and efficient training are paramount, particularly when dealing with intricate EEG data patterns.

Chapter 5

Training the DiLoc Neural Network

This chapter will explore the training process of the DiLoc neural network, outlining the decisions made during its training. Carefully choosing and adjusting elements of the training process ensures DiLoc's ability to make accurate predictions, establishing it as a valuable tool for solving the inverse EEG problem. Subsequent chapters will delve into the results and outcomes of this training, providing insights into the model's effectiveness.

5.1 Training Methodology Overview

Having constructed the DiLoc neural network architecture, we now embark on the phase of training the model to effectively localize equivalent current dipoles from EEG signals. This chapter delves into the process of training DiLocc and addresses various crucial aspects that influence the performance of the network. Training a neural network requires careful consideration and tuning of several key factors to achieve optimal results. In the process of training DiLoc, we will explore the following essential elements:

Data Preparation: We begin by preparing the data, which serves as the essential foundation for our machine learning model. This involves tasks such as data segmentation and standardization, ensuring the dataset is well-structured and representative.

Cost Function: Moving forward, we explore the significance of the cost function in guiding the network towards convergence. Understanding how to choose the appropriate cost function tailored to the problem is essential.

Back Propagation: The backpropagation algorithm plays an essential role in propagating error gradients through the network, refining the model's

weights and biases. We explain the algortihm.

Optimization Algorithm: The choice of optimization algorithm plays an important role in training efficiency and convergence speed. We delve into the tequniqe of gradient decent and its implications in the context of DiLoc.

Regularization Techniques: Overfitting is a common challenge in neural network training. We examine regularization techniques like dropout and weight decay to mitigate this issue.

Learning Rate Scheduling: Finally we present the method of adaptive learning rates. We discuss how learning rates significantly influence training dynamics, and how the method for scheduling learning rates during training ensure steady convergence.

5.2 Data Preparation

5.2.1 Data Segmentation

Standardization assumes that your observations fit a Gaussian distribution (bell curve) with a well behaved mean and standard deviation. You can still standardize your data if this expectation is not met, but you may not get reliable results.

The first step in prepering to *fit* a machine learning model, is to perform a data splkit, segregating the data set $\mathcal{D}$ into distinct sets for training, validation, and testing. This partitioning is done in order to make a model robust and compatible with multiple data sets. The size of each set commonly dependent on the size of the data set avaible, however a general guideline is that the majority of the data are allocated into the trainng set with the remainder going into the test set [15].

In the case of DiLoc's input data, we deal with 70 000 samples of EEG signals. Out of these, 50 000 samples are designated for the train and validation data. To ensure a representative and unbiased allocation, 80 percent of these 50 000 samples are randomly assigned to the training set. This training set serves as the core data that the network utilizes during the training process. The remaining 20 percent of the 50 000 samples form the validation set. This will play the role in preventing *overfitting*, the phenomenon where the network becomes excessively attuned to the training data and consequently performs poorly on new data. By independently evaluating the model's performance on the validation set throughout training, we have fine-tuned the network's parameters to achieve better generalization to unseen data. Upon completion of the training process, the test set comes into play. Comprising 20 000 samples, this set serves as the ultimate benchmark for evaluating

the model's capacity to generalize and make accurate predictions on new instances of data. By adhering to this train-validation-test data partitioning, we ensure a robust evaluation of DiLoc's performance and its capacity to effectively handle real-world scenarios with previously unseen data.

5.2.2 Data Scaling

Need some modification. Scaling is not done correctly in code. Scaling shoyld be done featurewise. Fix in code and write more detailed here. Having addressed the data segmentation, we proceed to the task of *data scaling*, which is a highly recommended procedure within machine learning. *Data standarizartion* is one out of two types of data scaling that involves centering the data around the mean μ and scaling it to achieve unit variance σ:

$$\mathcal{D}' = \frac{\mathcal{D} - \mu}{\sigma} \tag{5.1}$$

After standarization, the mean of the data set $\mathcal{D}'$ is 0 and the standard deviation is 1. By standarising the EEG inputs, we ensure that the distribution of the data becomes more uniform in all directions within the feature space, contributiong to more effective EEG signal analysis.

5.3 Cost Function

In the field of machine learning, *cost functions* play a crucial role in evaluating how well models make predictions. These mathematical functions evaluate how well models make predictions by measuring the discrepancy between predicted outcomes and actual target values. This evaluation results in a quantifiable metric known as *loss*. Higher loss values indicate poorer model performance, while lower values reflect more accurate predictions. When using the expression *fitting a model* one commonly refer to the prosess of finding optimal parameters $\boldsymbol{\theta}$ that minimize a chosen cost function. This iterative process utilizes training data to fine-tune the model's internal representations, improving its predictive accuracy.

this need to be rewritten... Selecting the appropriate cost function is important when addressing machine learning challenges. In our regression task, where the objective is to predict x, y, and z coordinates of single current dipoles, we employ the commonly used Mean Squared Error (MSE) cost function, well known for its continuous measure of model performance during training. However, given that our aim is to minimize the Euclidean distance between predicted dipole localizations (x_i, y_i, z_i) and true values $(\tilde{x}_i, \tilde{y}_i, \tilde{z}_i)$, we aptly refer to MSE as the Mean Euclidean Distance (MED). In essence, when predicting Euclidean coordinates, MSE and MED are conceptually equivalent, as they both aim to minimize the squared differences

between predicted and true values, but MED explicitly emphasizes the Euclidean nature of the coordinates, providing a more accurate reflection of our optimization goal:

$$\mathrm{MED}(\boldsymbol{\theta}) = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}((x_i - \tilde{x}_i)^2 + (y_i - \tilde{y}_i)^2 + (z_i - \tilde{z}_i)^2)} \tag{5.2}$$

Here $\boldsymbol{\theta} = \theta_0, \theta_1, ..., \theta_n$ represents the model parametes, $x_i, y_i and z_i$ stand for the predicted values and $\tilde{x}_i, \tilde{y}_i, \tilde{z}_i$ are the corresponding true values.

Due to the squaring step in the function, which penalizes larger errors more severely, the trained model is less likely to generate outlier predictions with substantial errors. However, in situations where the model produces a single, highly inaccurate prediction, the squaring effect can significantly amplify the error. Nevertheless, in many practical scenarios, the primary focus is not on these individual outliers, but rather on achieving a well-balanced model that delivers satisfactory performance across the majority of predictions [22].

In our simulation-driven context, devised to accommodate a spectrum of diverse EEG variations, it is noteworthy that the presence of apparent outliers is not rooted in sampling errors, but rather random fluctuations and indicative of distinct scenarios. As delineated in statistical literature [23], an outlier is characterized as an observation that seemingly deviates from the prevailing distribution of a dataset. This departure can be attributed to human or instrumental anomalies in the data acquisition process [24]. Figure 5.1 vividly depicts the distribution of minimum and maximum EEG observations across individual samples within the dataset. While the majority of data points exhibit values within the range of -1.5 to 3 μV, a subset extends beyond these bounds. It is crucial to underscore that these data points do not emerge as stochastic errors stemming from the acquisition process. Instead, they intentionally embody distinctive EEG signals that enrich the dataset's representational fidelity within authentic real-world contexts.

5.4 Back propagation algorithm

The back propagation algorithm is a fundamental tequniqe used in nenural networks in order to adjust the weights for the purpose of minimizing the cost function. To explain the implementation details of this technique, we follow the guidance provided in the book 'A high-bias, low-variance introduction to machine learning for physicists' (Pankaj Mehta, et al., 2019) as it offers a comprehensive treatment of the topic. The back propogation tequniqe leverages the chain rule from calculus to compute gradients for weight adjustments and can be summarized using four equations.

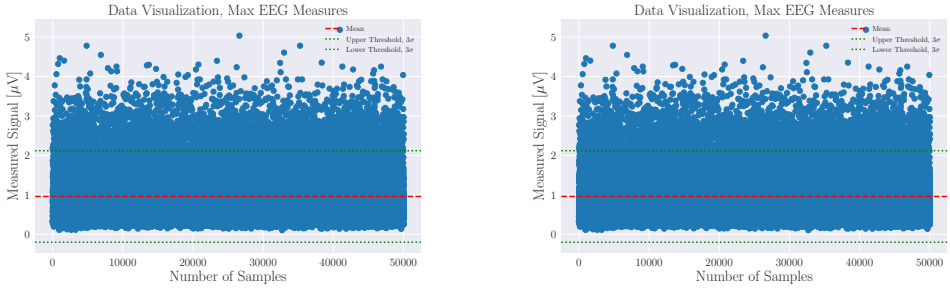

Figure 5.1: **Visualization of data set.** The panels visualize the maximum and minimum EEG measures for each sample within the data set. The plots need modification. Remove threshold etc...

Before introducing the equations, Mehta et al. establish some useful notation. They start by concidering a total of L layers within the neural network, with each layer identified by an idec l ranging from 1 to L. For each layer, they further assign weights denoted as $\mathbf{w}_{ik}^l$, which represent the connections between the k-th neuron in the previous layer, $l-1$, and the i-th neuron in the current layer, l. Additionally, they assign a bias value b_i^l to each neuron in the current layer.

The first eqation setting up the algorithm is the definition of the error δ_i^l of the i-th neuron in the l-th layer:

$$\delta_i^l = \frac{\partial C}{\partial(z_i^l)}, \tag{5.3}$$

where (z) denotes the weighted input. This equation can be thought of as the change to the cost function by increasing z_i^L infinitesimally. The cost function quantifies the discrepancy between the network's output and the target data. If the error δ_i^L is large, it indicates that the cost function has not yet reached its minimum.

The error δ_i^l can also be interpreted as the partial derivative of the cost function with respect to the bias b_i^l. This gives us the analogously defined error:

$$\delta_i^l = \frac{\partial C}{\partial(z_i^l)} = \frac{\partial C}{\partial(b_i^l)}\frac{\partial C}{\partial(z_i^l)} = \frac{\partial C}{\partial(b_i^l)} \tag{5.4}$$

where it in the last line has been used that the derivative of the activation function with respect to its input evaluates to 1, $\partial b_i^l/\partial z_i^l = 1$, meaning that the rate of change of the activation function does not depend on the specific value of the weighted input z_i^l.

By applying the chain rule, we can express the error δ_i^l in Equation 5.3 in terms of the equations for layer $l+1$. This forms the basis of the third equation used in the backpropagation algorithm:

$$\begin{aligned}\delta_i^l = \frac{\partial C}{\partial z_i^l} &= \sum_j \frac{\partial C}{\partial z_j^{l+1}} \frac{\partial z_j^{l+1}}{\partial z_i^l} \\ &= \sum_j \delta_j^{l+1} \frac{\partial z_j^{l+1}}{\partial z_i^l} \\ &= \sum_j \delta_j^{l+1} w_{ij}^{l+1} f'(z_i^l) \end{aligned} \tag{5.5}$$

Finally the last equation of the four back propagation equations is the derivative of the cost function in terms of the weights:

$$\frac{\partial C}{\partial w_{ij}^l} = \delta_i^l a_j^{l-1} \tag{5.6}$$

With these four equations in hand we can now calculate the gradient of the cost function, starting from the output layer, and calculating the error of each layer backwards. We then have a way of adjusting all the weights and biases to better fit the target data. The back propagation algorithm then goes as follows:

1. **Activation at input layer:** calculate the activations a_i^1 of all the neurons in the input layer.

2. **Feed forward:** starting with the first layer, utilize the feed-forward algorithm through **??** to compute z^l and a^l for each subsequent layer.

3. **Error at top layer:** calculate the error of the top layer using equation 5.3. This requires to know the expression for the derivative of both the cost function $C(\boldsymbol{W}) = C(\boldsymbol{a}^L)$ and the activation function $f(z)$.

4. **"Backpropagate" the error:** use equation 5.5 to propagate the error backwards and calculate δ_j^l for all layers.

5. **Calculate gradient:** use equation 5.4 and 5.6 to calculate $\frac{\partial C}{\partial z_i^l}$ and $\frac{\partial C}{\partial w_{ij}^l} = \delta_i^l a_j^{l-1}$.

6. **Update weights and biases:**
   $w_{jk}^l = w_{jk}^l - \eta \delta_j^l a_k^{l-1}$
   $b_j^l = b_j^l - \eta \delta_j^l$

This description makes clear the incredible utility and computational efficiency of the backpropagation algorithm. We can calculate all the derivatives using a single "forward" and "backward" pass of the neural network. This

computational efficiency is crucial since we must calculate the gradient with respect to all parameters of the neural net at each step of *gradient descent*, an optimalization algorithm that we will be delving into in the next section. Check with cite.

5.5 Optimization Algorithm

Doublecheck citing. In order to minimize the cost function and find the optimal values for the model parameters, $\boldsymbol{\theta}$, an optimization alorithm is typically employed. One widely used optimization algorithm is *gradient descent*, which iteratively updates the parameters based on the negative gradient of the cost function [15].

Gradient Descent is an iterative optimization algorithm used to locate a local minima of a differentiable function. The core concept of the algorithm is based on the observation that a function $F(\mathbf{x})$ will decrease most rapidly if we repeatedly move in the direction of the negative gradient of the function at a given point $\mathbf{w}$, $-\nabla F(\mathbf{a})$. This means that if

$$\mathbf{w}_{n+1} = \mathbf{w}_n - \eta \nabla F(\mathbf{w}_n) \tag{5.7}$$

for a sufficiently small *learning rate* η, we are always moving towars a minimum, since $F((w)_n) \geq F((w)_{n+1})$ [25]. After each update, the gradient is recalculated for the updated weight vector $\mathbf{w}$, and the process is repeated [26]. Based on this observation, the iterative process begins with an initial guess x_0 for a local minimum of the function F. It then generates a sequence $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n$ such that each element in the sequence is upated according to the rule:

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \eta_n \nabla F(\mathbf{x}_n), n \geq 0, \tag{5.8}$$

where $\eta_n \geq 0$. This process the forms a strictly decreasing process:

$$F(\mathbf{x}_0) \geq F(\mathbf{x}_1) \geq F(\mathbf{x}_2) \geq ... \geq F(\mathbf{x}_n). \tag{5.9}$$

Hence, with this iterative process, it is hoped that the sequence $(\mathbf{x}_n)$ converges to the desired local minimum [25].

However, it is important to note that the error function in gradient descent is computed based on the training set, so that each step requires that the entire training set, reffered to as the *batch*, is processed in order to evaluate the new gradient. In that sense, gradient descent is generally considered a suboptimal algorithm. This perception aligns with the algorithms sensitivity to the initial condition, $\mathbf{w}_0$, and the choice of the learning rate η. The sensitivity to initial conditions can be explained by the fact that we to a large extent most often deal with high-dimensional, non-convex cost functions with numerous local minima - where the risk of getting stuck in local

minimums if the initial guess is not accurate. Additionally, guessing on a too large learning rate may result in overshooting the global minimum, leading to unpredictable behavior, while a too small learning rate increases the number of iterations required to reach a minimum point, thereby increasing computational time. Stochastic gradient descent, however, is a version of gradient descent that has provided useful in practise for training machine learning algorithms on large data sets, and is the optimalization algorithm we have choosen when training DiLoc [26].

Stochastic Gradient Descent

The Stochastic Gradient Descent (SGD) method, as applied in the context of training DiLoc [26], is a powerful optimization technique that diverges from traditional gradient descent by randomly selecting subsets of the data during each iteration, as opposed to considering the entire dataset. This approach is particularly beneficial when dealing with extensive datasets. The update step in SGD can be expressed as:

$$\mathbf{w}\tau + 1 = \mathbf{w}\tau - \eta \nabla F_n(\mathbf{w}_\tau) \tag{5.10}$$

Here, $\mathbf{w}\tau$ represents the weight vector at iteration τ, η is the learning rate, and $\nabla F_n(\mathbf{w}\tau)$ is the gradient of the cost function F_n computed on a mini-batch, which refers to a randomly selected subset of the complete dataset.

In essence, SGD mirrors regular gradient descent but restricts its focus to a single mini-batch at each iteration. The introduction of this stochastic element, achieved by sampling from subsets of the data, offers a valuable advantage: it allows the algorithm to explore and potentially escape from local minima, promoting the discovery of superior solutions.

Furthermore, the incorporation of *momentum* into the SGD algorithm enhances convergence speed. Momentum introduces a form of memory into the optimization process by accumulating information about previous movements in parameter space. Specifically, momentum is realized through the following equations:

$$\mathbf{v}\tau = \gamma \mathbf{v}\tau - 1 - \eta \nabla F_n(\mathbf{w}_\tau) \tag{5.11}$$

$$\mathbf{w}\tau = \mathbf{w}\tau - 1 + \mathbf{v}_\tau \tag{5.12}$$

In these equations, $\mathbf{v}\tau$ represents the momentum vector at iteration τ, γ is the momentum coefficient, η is the learning rate, and $\nabla F_n(\mathbf{w}\tau)$ signifies the gradient of the cost function F_n computed on the mini-batch.

During the training of DiLoc, we observed that a learning rate of 0.001 yielded the most favorable results. Moreover, a mini-batch size of 32 was employed. The momentum coefficient was set to 0.35, a value that struck

an optimal balance between convergence speed and sensitivity to the initial learning rate parameter.

Add information about the value we are using!!

5.6 Regularization Techniques

Regularization techniques in machine learning help prevent overfitting and improve model interpretability. One common regularization technique is called the L2-norm penalty. In this technique, we add an extra term to our cost function that depends on the size of the model's weights. This added term ensures that the weights do not become excessively large while fitting the training data. By doing this, regularization reduces the risk of the model fitting the training data too closely, which can lead to overfitting. [16].

We can measure the size of the weights using the L2-norm, which results in our cost function taking this form:

$$C(\theta) = \frac{1}{N-1}\sum_{i=0}^{N}(y_i - \tilde{y}_i)^2 + \lambda \sum_{i,j=1}^{M,K} w_{ij}^2. \tag{5.13}$$

Here, M represents nodes in a specific layer, K is the number of input features considered for each node and λ is the tuning parameter that controls how much we penalize the flexibility of our DiLoc model. Here, flexibility means how much the model can adjust its coefficients to fit the data precisely. When $\lambda = 0$, the L2-norm has no effect, and the estimates produced will be equal to when the cost function is simply MSE. On the other hand, as λ approaches larger values, the impact of the shrinkage pentalty grows, and the coefficients estimates will approach smaller values. With smaller coefficients the model becomes less flexible, and the model will try to find simples pattern in the data, which in turn reduces the chances of overfitting [27].

5.7 Learning Rate Scheduling

The learning rate plays a critical role in the iterative Stochastic Gradient Descent procedure used to update model parameters during training, influencing the size of steps taken to minimize the cost function. To simplify the process of selecting an appropriate learning rate—neither too high nor too small—we employ a *learning rate scheduler*. This scheduler dynamically adjusts the learning rate during gradient descent optimization, enhancing both performance and training efficiency.

Various methods can be used to make the learning rate adaptive, but a common approach involves starting with a higher learning rate at the beginning of training. This allows for larger steps during the initial phases, speeding up progress. As training progresses and the model performance

approaches convergence, smaller steps are taken to fine-tune the parameters near the minimum, preventing overshooting. This strategy balances rapid initial convergence with meticulous fine-tuning later on [28].

For our model we have used the build-in PyTorch scheduler, ReduceLROnPlateau. This learning rate scheduler adjusts the learning rate when DiLoc's loss stops improving after a specified number of epochs. In other words, the learning rate remains unchanged as long as it enhances model performance but is reduced when results starts converging.

The scheduler offers several tunable arguments. We have set the *factor* to 0.2 in DiLoc, indicating the reduction factor for the learning rate when the loss plateaus. After the first update, the new learning rate becomes $\eta_{\text{new}} = 0.001 \cdot 0.2$, and this pattern continues for subsequent updates. The *patience* argument, set to 50, determines the number of epochs with no improvements before reducing the learning rate. The remaining arguments retain their default values and can be explored in greater detail, along with additional information about the scheduler, at https://hasty.ai/docs/mp-wiki/scheduler/reducelronplateau.

Chapter 6

Localizing Single Current Dipole Sources

Here comes the results In this chapter we will be looking at DiLoc's performance of the inverse problem. With the use of error metrics such as mean absolute error (MAE), mean euclidean distance (MED) and root mean euclidean distance (RMED) we will study how the DiLoc, as a simple feed forward neural network ables to solve the inverse EEG problem.

6.1 Testing Methodology

Upon achieving full training convergence, a neural network is deemed to have successfully discerned relevant patterns within the training dataset and optimized its parameters to minimize the cost function. To assess the performance of DiLoc, we turn our attention to an independent test dataset, encompassing 20,000 samples. This distinct dataset remains entirely unexposed to the network during the training process, serving as an unbiased and objective measure of the model's predictive accuracy and its capacity to generalize effectively to novel, unseen data.

To comprehensively evaluate the predictive capabilities of our DiLoc neural network on the test dataset, we have employed a diverse set of error metrics. Our primary objective is to ensure that DiLoc accurately predicts the locations of current dipoles in the cerebral cortex based on EEG data recorded from randomly distributed sources. Thus, the main focus lies on minimizing the MED as an overarching measure of performance, however we recognize the importance of diving into the performance of specific target coordinates, i.e., the x, y, and z values.

The MAE offers a straightforward measure of average prediction error for each individual target coordinate. When evaluating the distance between a specific target coordinate (e.g., x, y, or z) and the corresponding predicted value, this metric provides a clear and easily interpretable measure of the

accuracy of DiLoc's predictions for each dimension. It allows us to assess whether the network gives precise predictions for all target values or if it exhibits variations in accuracy across different coordinate dimensions.

In contrast, MSE introduces a squaring process that significantly penalizes larger errors across all dimensions. While this aspect helps in emphasizing and addressing significant deviations between the predicted and true coordinates, it presents values in squared units of the target values, which can be less intuitive to deal with when assessing individual coordinate components.

To address the squared unit issue of MSE while still considering error spread, we have incorporated RMSE into our evaluation framework. RMSE balances the emphasis on significant errors while yielding results in units consistent with the target coordinate values, making it more suitable for evaluating individual coordinate predictions (e.g., x, y, and z).

6.2 Need a name

According to Ness et al. (2021) [6], EEG signals are not particularly sensitive to minor shifts in the precise location of neural current dipoles. This insensitivity can be explained by the fact that relative to the dimensions of individual neurons and the thickness of the human cortex, EEG electrodes are located far away from cortical neural sources. Before commencing the assessment of DiLoc's performance, we aim to investigate whether it has the capability to differentiate EEG signals originating from neighboring neural dipoles and provide distinct predictions for their respective spatial locations. By .... Pearson correlation coefficient ....

The *Pearson correlation coefficient* is a useful statistical measure that quantifies the relationship between two variables. This coefficient is defined as the ratio between the covariance of the two variables and the product of their standard deviations. The coefficient is always a single number within the range of -1 to 1, with -1 denoting a perfect negative correlation and 1 indicating a perfect positive correlation. When the correlation coefficient equals 0, it signifies the absence of a linear relationship between the variables [29].

In our exploration of EEG insensitivity to small differences in dipole locations, we utilize the correlation coefficient as a measure. Figure 6.1 illustrates three EEG signals corresponding to dipoles positioned at neighboring points within the NYHM cortical matrix. The central blue dipole is positioned by the green and red dipoles, situated at the leftmost and rightmost positions within the cortical matrix, respectively. The Euclidean distance between the blue and green dipoles is 0.914 mm, while the blue and red dipoles are separated by 1.926 mm.

The EEG signals of the blue and green dipoles in Figure 6.1 exhibit signi-

ficant overlap, supported by a high correlation coefficient of 0.966, indicating a strong positive relationship between these measurements. Conversely, the EEG signals of the blue and red dipoles show less resemblance, with a correlation coefficient of 0.695, denoting a somewhat smaller linear relationship. Notably, these differences can be attributed to the rotation in the normal vector of the red dipole and the somewhat larger distance between the blue and red dipoles compared to the blue and green dipoles.

The rotation of the normal vector is also evident for the green dipole but is more pronounced for the red dipole. When selecting neighboring dipoles within the NYHM based on distance, rotations in the normal vectors occur due to constraints during data generation that ensure dipole orientation is perpendicular to the cerebral cortex. Given the complex folding of the cortex, such constraints may lead to rotations in the dipole normal vector when shifting the dipole's location, as demonstrated in this example. Consequently, distinct differences in simulated EEG signals may emerge even for neighboring dipoles within the NYHM.

Figure 6.1: EEG signals plotted against electrode number for three neighbouring dipoles with normal vectors (-0.60, 0.11, 0.79), (-0.75, -0.15, 0.64), (-0.21, 0.32, 0.93) and positions (-12.04, 34.71, 60.07), (-12.37, 33.92, 59.75), (-10.24, 34.61, 60.75). The correlation coefficient between the blue and green dipole is 0.97, while it is 0.70 for the blue and red dipole. Which coordinate system? Where is (0,0,0) located?

6.3 Performance Evaluation

The neural network was trained for 500 epochs, with MED loss convergence depicted in Figure 6.2. The training time for one epoch was approximately 11.00 seconds, summing up to a total training time of 5500 seconds, equivalent to roughly 1.5 hours. It's important to note that this training duration does not include data loading and preprocessing.

The validation loss appeared to stabilize after approximately 350 epochs. At this point, the training could have been stopped. However, we extended the training to the full 500 epochs. This decision was made to confirm that no further improvements in the validation data's performance would occur and to emphasize that the network had fully converged when undergoing evaluation.

Figure 6.2 illustrates a clear trend of decreasing loss, indicating that the network effectively learned patterns in the data. The validation loss stabilization is noticeable around 350 epochs, while the training loss continues to decrease until it stabilizes between 400 and 500 epochs. This observation suggests that the model did not exhibit signs of overfitting, as the validation loss remained constant, and the training loss eventually stabilized.

Additionally, Figure 6.3 provides insight into the development of the validation loss for the separate target coordinates, x, y, and z, plotted against training epochs. This figure confirms that all separate target coordinates were equally weighted, resulting in similar loss values for each. Moreover, it demonstrates that the small fluctuations in loss, observed before 350 epochs, disappeared beyond this threshold, indicating a stabilization of the loss for all three target coordinates. This observation aligns with the trend of the validation loss stabilizing at approximately 350 epochs, that we could see in Figure 6.2.

6.3.1 Position Spesific Performance

Range of values should be placed somewhere else. Errors should be contextualized within the range. In evaluating the DiLoc network's performance, we further examined its accuracy across three spatial coordinates: x, y, and z. These coordinates represent the positions of dipole sources in three-dimensional space. Notably, the x-coordinate ranges from -72 to 72 mm, the y-coordinate from -106 to 73 mm, and the z-coordinate from -53 to 82 mm, providing context for the interpretation of error metrics.

Add histogram instead of table? The MED of DiLoc's predictions on the unseen test data measures 1.33 mm. In Table 6.1, we have provided the percentages of samples falling within various MED threshold values. Impressively, DiLoc provides an accuracy smaller than 12 mm for the entirety of the samples within the test dataset. Furthermore, it is noteworthy that an impressive 96.6% of the samples exhibit a MED smaller than 3 mm, an

Figure 6.2: Training- and validation MED loss for DiLoc with 50,000 samples and tanh as activation function.

Mean Euclidian Distance for Test Samples			
MED <3 mm	MED <5 mm	MED <10 mm	MED <12 mm
96.595 %	99.735 %	99.995 %	100.000%

Table 6.1: **MED evaluation on test samples; Predicting Location** Performance of the network on thes test dataset comprising 20,000 samples, presented as the percentage of samples falling within MED thresholds of 3 mm, 5 mm, 10 mm, and 12 mm, respectively.

achievement that underscores the network's high precision.

Table 8.1 below presents the results for the MAE in the DiLoc network's predictions. Notably, the MAE values for the x, y, and z-coordinates range from 0.645 mm to 0.678 mm. These findings indicate that, on average, the network's predictions exhibit an error smaller than 1 mm in each coordinate, underscoring its high level of accuracy.

The MSE values, ranging from 0.747 mm^2 to 0.824 mm^2, corroborate the network's remarkable precision. Smaller MSE values signify high performance, and in this evaluation, all MSE values are comfortably below the 1 mm^2 threshold. This observation highlights the network's precision, especially given the wide range of coordinates involved. The small MSE values indicate minimal error spread, implying few significant deviations between

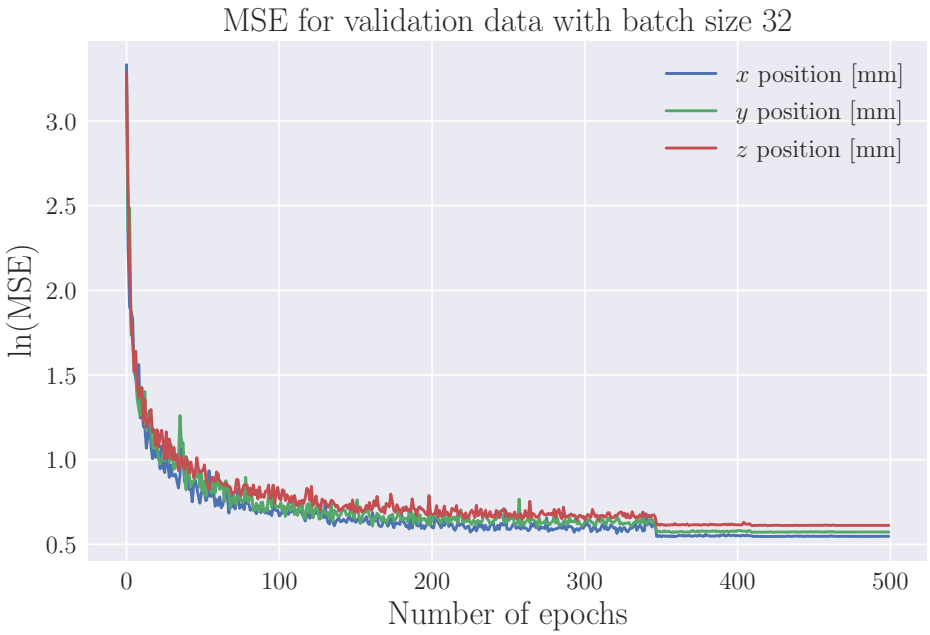

Figure 6.3: Validation MED loss for the separate target values; the x-, y-, z-coordinate.

predicted and true coordinates.

To quantify the accuracy further, RMSE values were computed, ranging from 0.864 mm to 0.908 mm. It is worth noting that RMSE is slightly higher than the corresponding MSE values due to the square root operation involved. Importantly, all RMSE values remain below the 1 mm threshold, signifying that DiLoc's predictions maintain a limited error spread, typically staying within 1 millimeter of the actual values.

To provide overall positional errors given the MAE, MSE, and RMSE, the mean of each error metric is also given in the table. The resulting error values remain small, further reaffirming the network's precision in predicting dipole locations for the inverse problem.

Notably, among the three coordinates, the z-coordinate exhibits the highest error values. This observation suggests that the DiLoc network encounters greater challenges in accurately predicting the z-coordinate of the dipole source. One plausible explanation is related to the nature of the inverse problem, where the depth of the dipole sources primarily influences the magnitude rather than the pattern of EEG recordings. Additionally, the z-coordinate's smaller representation compared to the x- and y-coordinates could contribute to the consistently larger errors in the z-direction. However, we underscore that this discrepancy in error magnitude is very small and could also be influenced by randomness. Overall, these error metrics in-

dicate that the DiLoc network can predict dipole locations with a reasonable level of accuracy.

Make figure to show. Mention the practical significance of the example, such as how close the predicted values need to be for specific applications. To showcase the networks performance, we look the prediction of a randomly chosen sample within the test set. For a dipole located at x = 66.9 mm, y = -26.1 mm and z = 41.7 mm the network predicts the coordinates to be x_{pred} = 66.5 mm, y_{pred} = -26.4 mm and z_{pred} = 41.9 mm. The predicted values are considerable close to the true values, with an error of 0.4 mm in the x-coordinate, 0.3 mm in the y-coordinate, and 0.2 mm in the z-coordinate, giving an eucleadian distance of 0.54 mm.

	Error Metrics for Target Values			
	x-coordinate [mm]	y-coordinate [mm]	z-coordinate [mm]	Position Error [mm]
MAE	0.645	0.665	0.678	0.662
MSE	0.747	0.775	0.824	0.782
RMSE	0.864	0.880	0.908	0.884

Table 6.2: **Evaluation of the DiLoc performance utializing different Error Metrics.**
Network performance on test dataset consisting of 20000 samples. The errors are measured using Mean Squared Error (MSE), Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE).

6.3.2 Performance at Different Brain Structures

In order to conduct a further detailed analysis of the network's performance, Figure 6.4 presents the Mean Eucleadian Distance for various dipole locations within the New York head model cortex matrix. The figure provides valuable insights into the distribution of errors across different regions of the cortex, with three cross-sections — front, top, and side — depicted for examination. It is important to note that these cross-sections unavoidably include data points from the training, validation, *and* test datasets, making these results indicative of the network's overall performance rather than real-world scenarios. However, the analysis aims to examine the distribution of errors and identify potential areas where the network's performance may be weaker, helping to gain valuable insights into its predictive capabilities.

The MED values presented in the panels are mostly below 1 mm, which indicates a high level of accuracy in the network's predictions. These results are promising and demonstrate the network's ability to estimate dipole locations with a high level of precision. The panels also offer an opportunity to assess whether the network performs differently for dipoles located in the gyrus compared to the sulcus.

Initially, it might be assumed that EEG signals originating from dipoles in the sulcus present greater challenges for the network's analysis and prediction. This assumption is based on the deeper placement of dipoles within the sulcus compared to those in the gyrus, as well as the potential complexities introduced by the dipole's orientation within the cortex. However, upon closer examination of Figure 6.4, it becomes evident that the distribution of MED values does not exhibit a clear correlation with the brain's structural characteristics. The MED values appear to vary randomly across different regions, indicating that the network's performance is not significantly influenced by the distinction between the gyrus and sulcus.

Surprisingly, the Mean Euclidian Distance (MSE) for all data points where dipoles are located in sulci is measured to be 1.68 mm, which is even smaller than for dipoles in the gyrus, where the MSE measures 1.70 mm. This observation challenges the initial assumption, and rather indicates that the network demonstrates exceptional accuracy in predicting dipole locations, irrespective of their placement within the cortex. The small MED values further underscore the network's remarkable capacity to effectively capture the intricate features and variations associated with deeper cortical placements. These findings not only attest to the network's robustness but also reinforce its potential for precise dipole localization within the human brain across different cortical structures.

The "theory" should be mentioned somewhere else and already be known for the reader. Furthermore, the figures reveal a somewhat noticeable concentration of data points with red marks in the deeper locations within the cortex, indicating higher MED values. This observation is consistent with the slightly higher error values observed for the z-coordinate, as presented in Table 8.1, and aligns with the theory related to the nature of the inverse problem. According to this theory, EEG patterns for dipole sources do not cause substantial changes in the pattern of electrical potential recordings; instead, they primarily influence the magnitude of the signals, which may challenge DiLoc's ability to discriminate deep sources from each other. Said with other words, the presence of higher MED values in deeper regions might be attributed to the decreasing signal-to-noise ratio of EEG signals originating from these cortical areas.

In conclusion, the detailed analysis of the network's performance through cross-sectional representations provides valuable insights into its predictive capabilities. The consistently low MED across different cortical regions demonstrate the network's remarkable accuracy in estimating dipole locations. Moreover, the absence of a clear correlation between MED and brain structural characteristics suggests that the network performs robustly across diverse cortical structures. These results have significant implications for the network's potential clinical and research applications, as it showcases its ability to accurately predict dipole locations within the human brain, regardless of their depth and orientation within the cortex.

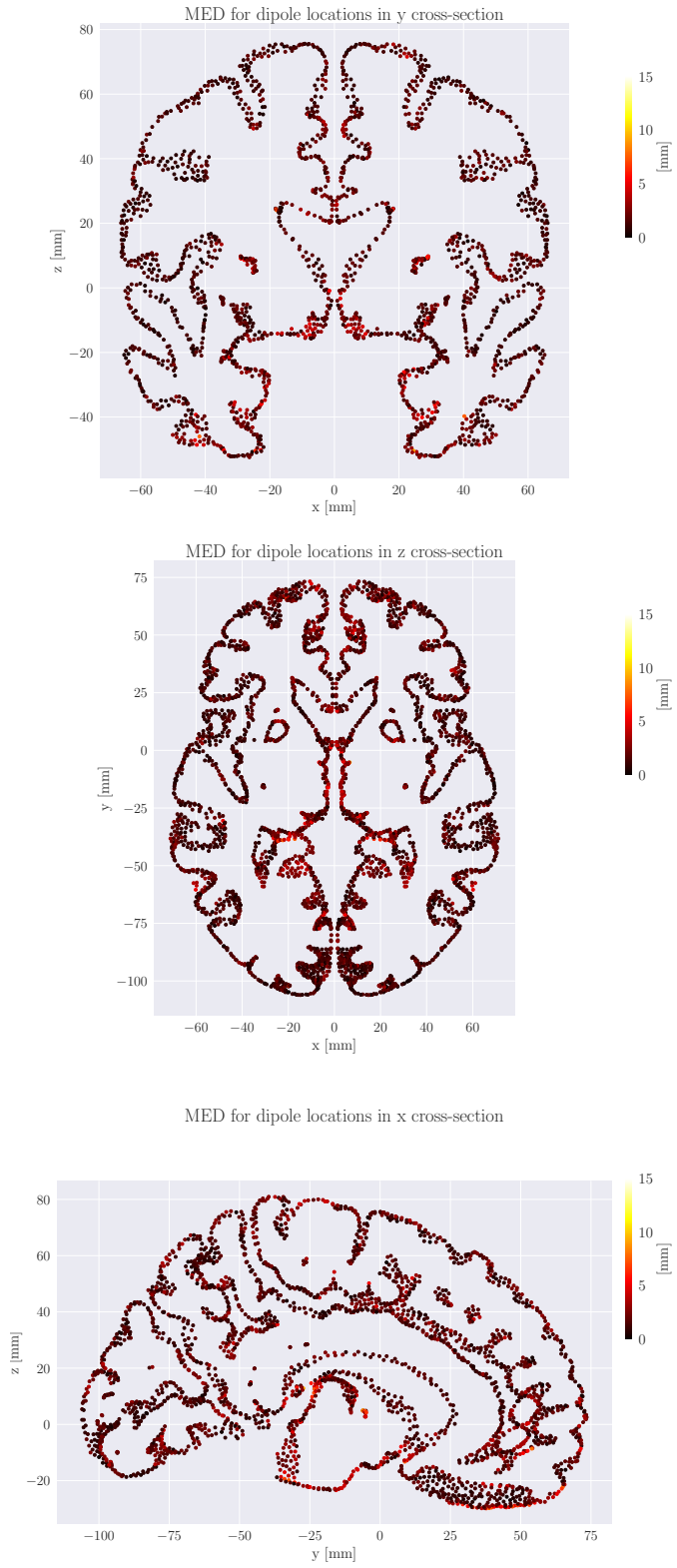

Figure 6.4: Different cross-sections of the cortex from the New York head model, seen from front, top and side. Each point represents a possible position in the cortex matrix. The color of the each point indicates the mean absolute error (MAE) of the neural network when predicting that specific dipole location.

Chapter 7

Convolutional Neural Network Approach for Localizing Single Dipole Sources

In this chapter, we explore the utilization of a Convolutional Neural Network (CNN) for the task of localizing simple current dipoles from EEG recordings. The CNN is a sophisticated type of feed-forward neural network that excels at learning spatial features from images. The objective of this investigation is to assess whether leveraging spatial information in EEG recordings as images can enhance Dilocs's ability to analyze the data and yield more accurate predictions for localizing the sources generating the neural signals.

Convolutional Neural Networks

Convolutional neural networks (CNNs) is an other variant of FFNNs that have drawen inspiration from the functioning of the visual cortex of the brain. In the visual cortex, individual neurons exhibit selective responses to stimuli within small sub-regions of the visual field, known as receptive fields. This property allows the neurons to effectively exploit the spatially local correlations present in natural images. Mathematically, the response of each neuron can be approximated using a convolution operation [16].

CNNs mimic the behavior of visual cortex neurons by utializing a specific connectivity pattern between nodes in adjacent layers. Unlike fully contected FFNNs, where each node connects to all nodes in the preceding layer, CNNs local connectivity. In other words, each node in a convolutional layer is only connected to a subset of nodes in the previous layer. Typically, CNNs consist of multiple convolutional layers that learn local features from the input data. These layers are followed by a fully connected layer that combines the learned local information to produce the final outputs. CNNs find wide applications in image and video recognition tasks [16].

7.0.1 Data Set

Convolutional Neural Networks (CNNs) are well-known for their effectiveness in processing image data. To harness the potential of CNNs for EEG data analysis, we convert the original dataset presented in Chapter 4 into image-like data through interpolation. Interpolation, a widely-used mathematical technique, estimates values between known data points. This transformation results in a regular 2D grid representation of the original one-dimensional EEG data, effectively creating EEG data with image-like characteristics and preserving spatial structures.

The resulting data is represented as a 20x20 matrix, where each element holds the intensity value of the EEG potential recorded at the corresponding electrode location. We construct this matrix to resemble the shape of a grayscale image, with a single channel added to represent the spatial distribution of the measured EEG signals. However, unlike typical grayscale images where each pixel represents color intensity, in this context, each pixel's value denotes the intensity of the recorded EEG signal at that specific electrode location. This unique representation enables the CNN to leverage the spatial arrangement of the EEG data and learn relevant patterns and local relationships, much like how CNNs process traditional image data.

The preserved spatial structures enable the network to exploit local relationships between neighboring recording electrodes. For instance, when neighboring electrodes record high EEG values, it suggests that the specific electrode should also record a relatively high EEG value due to their close spatial proximity. These spatial relationships may contribute to faster training times for the network, as it can efficiently learn meaningful representations by leveraging these patterns.

Figure **??** illustrates the process of interpolation. The right panel shows the original data, representing the cortex seen from above (x-y-plane). Each measuring electrode is depicted as a circle holding the EEG recording at that specific electrode. The middle and left panels display the contour plots of the original EEG data and the interpolated data, respectively. The contour plot of the interpolated data illustrates how the input data for the convolutional neural network appears in an image-like manner.

Architecture, Hyperparameters and Training

As explained in Chapter 3, Convolutional Neural Networks (CNNs) are structured as a sequence of interconnected layers designed to process and extract meaningful features from the input data. In the case of EEG input data, we adopt a specialized CNN architecture tailored to handle image-like data representations. The data transformation involves constructing a 20x20 matrix, akin to the shape of a grayscale image, with a single channel added to represent the spatial distribution of the measured EEG signals.

The first layer in the network, is a 2D convolutional layer. It takes the input image with one channel and applies six distinct filters, each of size 5x5. These filters are responsible for learning specific spatial patterns and detecting relevant features within the input image. As a result of this convolutional operation, the output tensor's spatial dimensions reduce to 16x16, and the depth becomes six, signifying the extraction of six distinct feature maps. Following the convolution layer, a Max Pooling layer, with kernal size 2x2 and stride 1 is employed. This pooling layer aims to downsample the spatial dimensions of the feature maps while preserving the most salient features. The pooling operation reduces the spatial resolution to 15x15, and the depth remains unchanged at six. Next, a second 2D convolutional layer, takes the six-channel output from the previous pooling layer. This layer employs 16 filters of size 5x5, extracting a more complex hierarchy of features from the input data. The output tensor from this layer has spatial dimensions of 11x11 and a depth of 16, signifying the presence of 16 distinct feature maps. Following another Max Pooling layer is employed. Similar to the previous pooling operation, this layer further downsamples the spatial dimensions while preserving the depth, resulting in a feature map size of 10x10 with 16 channels. Further, the output from the last pooling layer is flattened into a one-dimensional vector. This process collapses the spatial dimensions of the feature maps, resulting in a 1D tensor of size 1600 (10x10x16). After flattening, the network proceeds with three fully connected, dense, layers. These layers are responsible for incorporating global context and making high-level abstractions from the learned features. The first fully connected layer, consists of 120 neurons, followed by 64 neurons. Lastly,we have the output layer with three neurons, corresponding to the three coordinates of the source generating the eeg signal. In Figure 7.2, we have provided an illusration of the architecture of the Convolutional Neural Network.

The activation function ReLU is applied after each convolutional and pooling layer, introducing non-linearity to the network and enabling it to learn complex relationships within the data. The fully connected layers use the hyperbolic tangent activation function, which introduces non-linearity and scales the output between -1 and 1. Finally, in the output layer, we opted for a linear transformation without the use of any activation function. This setup allows the neural network to provide direct and unconstrained predictions for the x-, y-, and z-positions of the desired dipole source, as required in our application.Throughout the network, the weights of the fully connected layers are initialized using the Xavier normal distribution, a widely used technique to set initial weights in deep neural networks, promoting better convergence during training.

The training process for the specialized convolutional neural network (CNN) followed similar techniques to the original DiLoc network, as described in detail in Chapter 5. To ensure effective learning and accurate predictions, stochastic gradient descent (SGD) with momentum was utilized

as the optimizer, and mean squared error (MSE) served as the chosen cost function. Additionally, L1 and L2 regularization techniques were incorporated to mitigate overfitting and enhance the network's ability to generalize to new data. During training, mini-batches of size 32 were employed to introduce variability in the data and prevent the network from becoming stuck in local minima. Notably, the CNN employed specific hyperparameters, setting the learning rate to 0.001 and the momentum to 0.009, which were tailored to accommodate the processing requirements of image-like EEG data. To facilitate convergence, a learning rate scheduling approach was adopted, gradually reducing the learning rate during training, striking a balance between rapid initial convergence and fine-tuning towards the later stages. Subsequently, the CNN's performance was rigorously evaluated on an independent test dataset to provide an unbiased assessment of its predictive accuracy and generalization capabilities to novel data.

7.0.2 Performance Evaluation

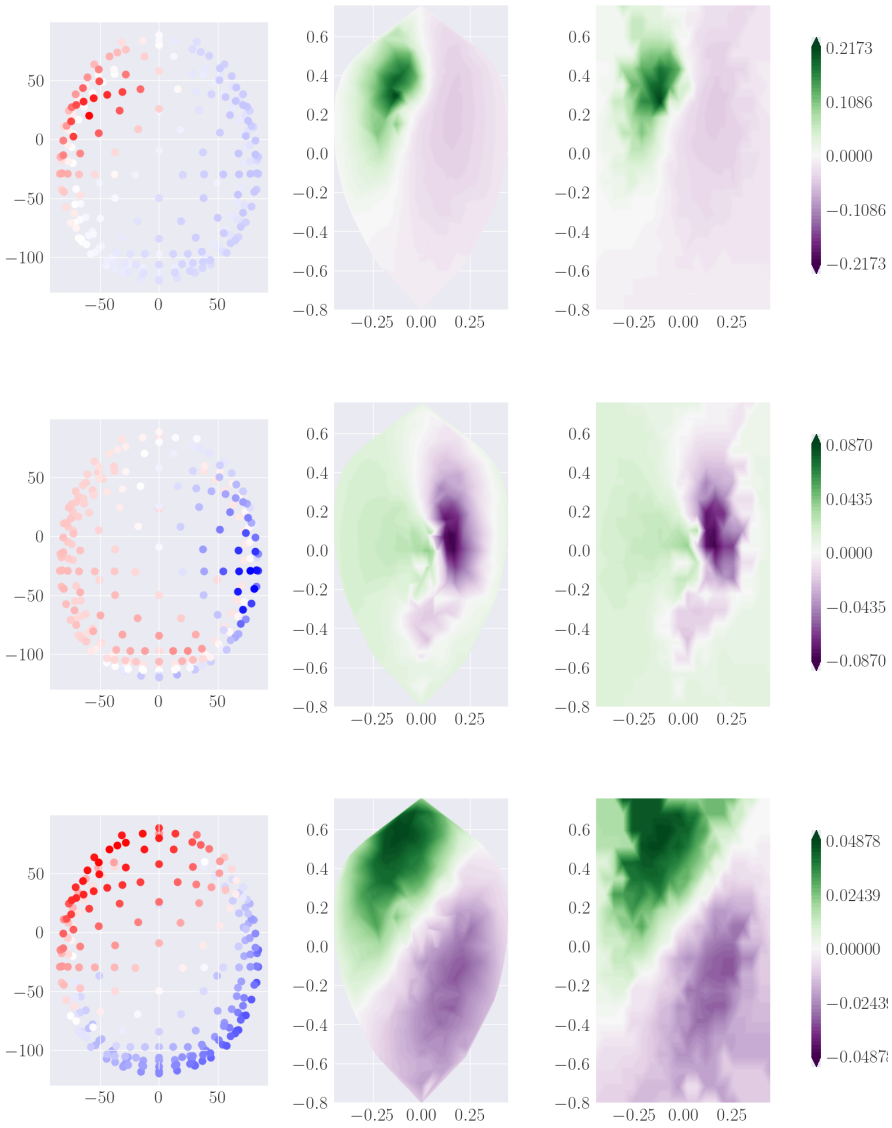

Figure 7.1:
Right Panel: EEG measures for three different samples, expressed in microvolts (μV). Each sample represents an EEG recording at specific electrode positions.
Middle and Left Panels: Illustration of the interpolation of the EEG data into a two-dimensional matrix. The interpolated data represents the transformation of original electrode recordings into a regular 2D grid, effectively converting the one-dimensional EEG data into an image-like format. The contour plots visualize the spatial distribution of EEG potential intensities, with each point in the matrix corresponding to a specific electrode location.

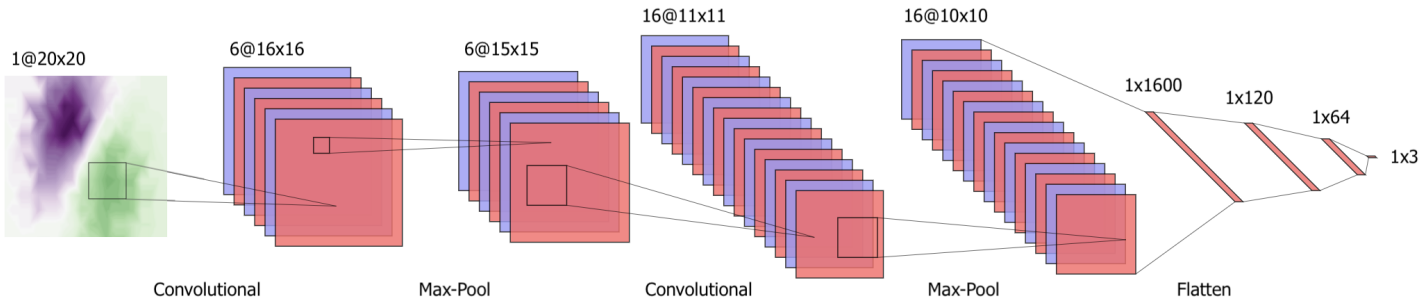

Figure 7.2: The architecture of the Convolutional Neural Network.

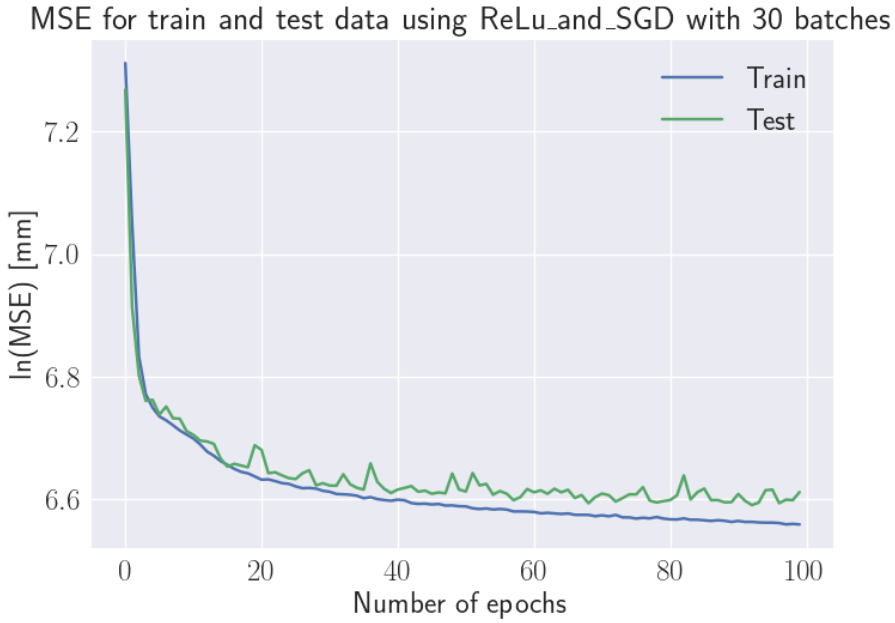

Figure 7.3: The validation accuracy for Convolutional Neural Network with 10 000 samples (20x20 matrix) with ReLU activation function.

Chapter 8

Extending the DiLoc Network

In this chapter, we explore various extensions and small modifications to the DiLoc network aimed at enhancing its capability to identify characteristics of current dipoles besides also addressing the inverse problem. We delve into three distinct extensions of the initial inverse problem, challenging DiLoc's predictive capabilities in handling more complex scenarios.

The first case involves extending the dataset by assigning individual magnitudes to each dipole source. This extension presents the network with the task of predicting both location of the source and its corresponding magnitude. Subsequently, the second scenario transitions from predicting the location of a single dipole source to estimating the center and radius of a population of dipoles, alongside the magnitude of the electrical signals generated. This extension introduces additional complexities to the localization process. Lastly, we investigate DiLoc's ability to predict the positions of two individual dipole sources located at different areas within the cortex, jointly contributing to the EEG signals recorded by the scalp electrodes. These extensions aim to comprehensively evaluate the network's adaptability and generalization to more intricate problems that might be of interest in real-world applications.

8.1 Method: Adjustments in Data Set and Architecture

Throughout the chapter, we methodically introduce distinct modifications to DiLoc for each problem, assess the network's performance, and offer insights into its strengths and limitations when addressing these novel challenges. However, before delving into the results, we will first address important considerations that apply to all extensions. This encompasses ensuring DiLoc effectively handles varying units in its output, selecting an optimal cost function tailored to our specific problems, and establishing criteria for assessing DiLoc's performance in the context of its intended tasks.

8.1.1 Scaling of Target Values

Do we scale multiple dipoles even withiot magnutide??? In addressing the extended problems within DiLoc, the network is tasked with predicting target values with significant variations in both range and units. This necessitates an essential preprocessing step: the scaling of target data. The importance of this scaling stems from the fact that the traditional cost function calculates differences between predicted output and actual target values, and add together the distinct terms within the loss. When the target values exhibit disparities in their ranges and units, several challenges arise.

Firstly, it is imperative to acknowledge that attempting to aggregate add together losses from distinct target values with varying units lacks a straightforward and intuitive approach. Combining values with different units into a single loss function can lead to an ambiguous interpretation and hinder the meaningful evaluation of the network's performance.

Secondly, the variation in the range of target values poses a potential issue. Variations can result in certain dimensions being imbalanced in their influence on the overall error, potentially overshadowing dimensions with smaller ranges. This asymmetry in the error calculation can lead to a biased optimization process and hinder the network's ability to effectively learn from the data.

To mitigate these challenges, we employ a normalization technique that transforms the output values into a consistent range, specifically ranging from 0 to 1. This normalization is achieved using the following formula:

$$\tilde{y_i} = \frac{\tilde{x_i} - \tilde{\min}(x)}{\tilde{\max}(x) - \tilde{\min}(x)} \tag{8.1}$$

Here, $\tilde{y_i}$ represents the i^{th} normalized value in the dataset for a specific target category, x_i is the i^{th} value in the corresponding target dataset, and $\min(x)$ and $\max(x)$ denote the minimum and maximum target values for the specific target category.

It is essential to emphasize that this normalization process is performed separately for each target category. By normalizing the target values in this manner, we strive to create a more equitable cost function where all target values contribute equally to the overall error calculation. Consequently, DiLoc can harness its learning capabilities more effectively and hopefully perform better in its tasks.

8.1.2 Sigmoid as Last Layer Activation Function

In the problem concerned with predicting the postion of a single dipole source, DiLoc did not have any activation function in the last layer. However, considering that the output data has been normalized to a range from

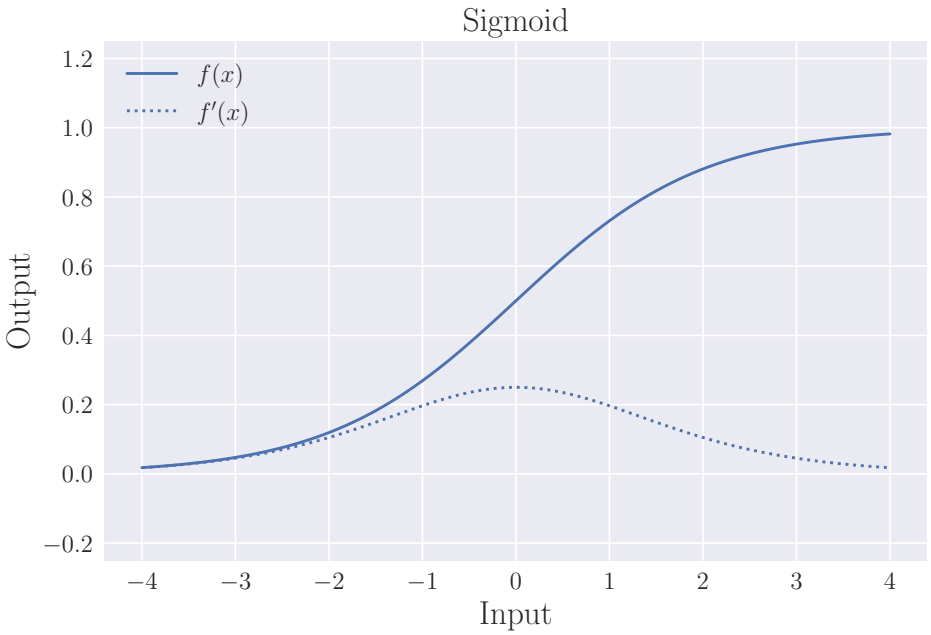

Figure 8.1: Need cite and figure text. Sigmoid activation function.

0 to 1, we deem it appropriate to employ the *Sigmoid* activation function in the output layer.

The Sigmoid function is known as one of the more biologically plausible activation functions as the ouput of inactivated neurons returns zero [**Jensen2022**]. More precised it is a logistic mathematic function meaning that it maps its input to a value between 0 and 1:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{8.2}$$

The Sigmoid function maps the output values to a range between 0 and 1, which aligns with our desired output range, for the exteded problems. This choice of last layer output function may potentially facilitate the training process, as it enables the network to converge more effectively towards the desired outputs, and resticts the network from outputting values outside the desired rarget range.

8.1.3 Choosing an Optimal Cost Function

Selecting an appropriate cost function for is pivotal in addressing machine learning challenges. In regression tasks like ours, the mean squared error (MSE) is a widely used cost function, particularly suitable for linear regression. The MSE is preferred due to its simplicity and continuous measure of model performance during training. It calculates the squared differences

between the model's predictions and target values, then takes the mean across the entire dataset:

Having a clear understanding of our desired model outcomes, we aim to develop a customized cost function that aligns precisely with our specific objectives. This tailored cost function may potentially outperform the Mean Squared Error (MSE) cost function, which we previously utilized.

Our new objective is to create a model capable of accurately predicting not only the positions of individual dipoles but also various aspects of dipole signals, including their magnitudes, the radii of extended dipole populations, and the positions of dipole pairs where both contribute to the complete EEG signal.

To achieve this objective, our ideal cost function is composed of several components. Firstly, it should minimize the Euclidean distance between target dipole localizations $\theta x, y, z$ and their true values $\tilde{\theta} x, y, z$:

$$\mathrm{MED}_{x,y,z}(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=0}^{n-1}\sqrt{(\theta_{x,i}-\tilde{\theta}_{x,i})^2+(\theta_{y,i}-\tilde{\theta}_{y,i})^2+(\theta_{z,i}-\tilde{\theta}_{z,i})^2}, \quad (8.3)$$

Additionally, the cost function should minimize the absolute error between the predicted magnitude θA and the true magnitude $\tilde{\theta} A$:

$$\mathrm{MAE}_A(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=0}^{n-1}\|\theta_{A,i}-\tilde{\theta}_{A,i}\|, \quad (8.4)$$

Similarly, it should minimize the error between the predicted radius θr and the true radius $\tilde{\theta}$r:

$$\mathrm{MAE}_r(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=0}^{n-1}\|\theta_{r,i}-\tilde{\theta}_{r,i}\|, \quad (8.5)$$

Finally, the cost function should minimize the Euclidean distance among a set of m dipoles in the $x-$, $y-$ and $z-$ dimensions. This is expressed by the equation:

$$\begin{aligned}\mathrm{MED}_{x_1,y_1,z_1,\ldots,x_m,y_m,z_m}(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=0}^{n-1}\Bigg(&\sqrt{(\theta_{x_1,i}-\tilde{\theta}_{x_1,i})^2+(\theta_{y_1,i}-\tilde{\theta}_{y_1,i})^2+(\theta_{z_1,i}-\tilde{\theta}_{z_1,i})^2}\\ &+\sqrt{(\theta_{x_2,i}-\tilde{\theta}_{x_2,i})^2+(\theta_{y_2,i}-\tilde{\theta}_{y_2,i})^2+(\theta_{z_2,i}-\tilde{\theta}_{z_2,i})^2}\\ &+\ldots\\ &+\sqrt{(\theta_{x_m,i}-\tilde{\theta}_{x_m,i})^2+(\theta_{y_m,i}-\tilde{\theta}_{y_m,i})^2+(\theta_{z_m,i}-\tilde{\theta}_{z_m,i})^2}\Bigg)\end{aligned} \quad (8.6)$$

Deside wether or not it should be two or more dipoles. We note, that in the context of our problem involving multiple dipoles, the target vector comprises two sets of coordinates corresponding to the positions of the dipoles we aim to predict. Similarly, the output vector contains two sets of coordinates. Traditional cost functions, such as Mean Squared Error (MSE), straightforwardly map the first set of coordinate vectors in the target with the first set of coordinates in the output, assuming that the order is correct and that every element should align perfectly. However, this approach can overlook the possibility that another permutation could represent the correct mapping.

Our network's primary task is to decipher patterns and learn from EEG data. It may not inherently grasp the correct order in which to place the coordinates of two dipoles (i.e., which should come first or second in the output vector), potentially leading to suboptimal mappings that misguide the network's weight updates during loss calculation. This limitation could increase computational time or, in the worst case, hinder the network's learning process.

This step involves systematically evaluating all possible permutations of target and output vectors This approach ensures that we consider all valid combinations while still maintaining the clarity and simplicity of our cost function equatio

To overcome this challenge, our customized cost function systematically explores all possible permutations of target and output vectors, ensuring that the network explores all valid combinations and determine which combination yields the minimum loss. Once we identify the optimal permutation, we use it to calculate the cost using the equation above 8.6. This approach fosters more effective learning and adaptation to various scenarios, enabling the network to capture the genuine underlying patterns in the data. By enabling the cost function to compute all permutations and select the one yielding the minimum loss, we provide the network with the necessary flexibility to excel in this complex task.

Add where unit tests can be found? For all terms within the cost function, comprehensive unit tests have been developed to confirm its intended functionality. Each problem introduced for the network corresponds to a distinct form of the customized cost function:

$$C(\boldsymbol{\theta}) = \begin{cases} \text{MED}_{\text{x,y,z}}(\boldsymbol{\theta}), & \text{if } \|\boldsymbol{\theta}\| = 3 \\ \text{MED}_{\text{x,y,z}}(\boldsymbol{\theta}) + MAE_{\text{A}}(\boldsymbol{\theta}), & \text{if } \|\boldsymbol{\theta}\| = 4 \\ \text{MED}_{\text{x,y,z}}(\boldsymbol{\theta}) + MAE_{\text{A}}(\boldsymbol{\theta}) + MAE_{\text{r}}(\boldsymbol{\theta}), & \text{if } \|\boldsymbol{\theta}\| = 5 \\ \text{MED}_{\text{x}_1,\text{y}_1,\text{z}_1,\ldots,\text{x}_m,\text{y}_m,\text{z}_m}(\boldsymbol{\theta}), & \text{otherwise} \end{cases} \tag{8.7}$$

Here, $|\boldsymbol{\theta}|$ signifies the length of the target vector. When $|\boldsymbol{\theta}| = 3$, the simplest problem is considered, where the network predicts the coordinates of a single-point current dipole, as explored in the previous chapter(s). If $|\boldsymbol{\theta}| = 4$, the network predicts the x-, y-, and z-coordinates of a single dipole, in addition to the magnitude of the signal strength. When $|\boldsymbol{\theta}| = 5$, the target vector encompasses all previously mentioned values, along with the size of a current dipole population with radius. Finally, for $|\boldsymbol{\theta}|$ greater than 5, the multiple dipole problem is addressed, where the network predicts the locations for two or more point source dipoles situated at distinct positions within the cortex.

IMPORTANT: This need modifications! In crafting our customized cost function, it is important to acknowledge that, like the built-in Mean Squared Error (MSE) cost function, our formulation inherently treats all target values equally during the optimization process. In other words, the algorithm assigns the same weight to each target value when striving to reduce the overall loss. This approach ensures that errors of equal percentage magnitude in different target values are treated on a level playing field. Consequently, a 1% error for one target value is considered as important as a 1% error for another target value, regardless of the specific range or scale of these values.

It is worth noting that our choice to uniformly weight all target values is an intentional design decision. While alternative approaches, such as assigning different weights to different target values, could have been explored, we prioritize the creation of a balanced model that can accurately predict all facets of our target values. This approach stems from our objective of achieving a comprehensive understanding of EEG signal sources through a holistic and equitable modeling approach.

Moreover, this uniform weighting approach aligns with our broader modeling philosophy, emphasizing the creation of a model that is versatile and adaptable across a spectrum of EEG data variations. Our aim is not only to develop a model capable of accurately predicting target values but also to ensure that its predictive capabilities are unbiased and comprehensive, covering the multifaceted aspects of EEG signal analysis.

In this way, our customized cost function showcases the fusion of machine

learning principles with the nuanced requirements of clinical medicine, as we strive to bridge the gap between technical prowess and real-world medical applications.

Another title, fex, new architecte?

8.1.4 Overview on Arcitecture and Hyperparameters

In figure 8.2 we have illustraded the architecte of the extended DiLoc network, outputting m target values, depending on the problem to solve. We see that DiLoc still takes an input of 231 data points corresponding to the number of recoring electrodes, and still has the same number of layers and nodes in each layes, except from the output layer.

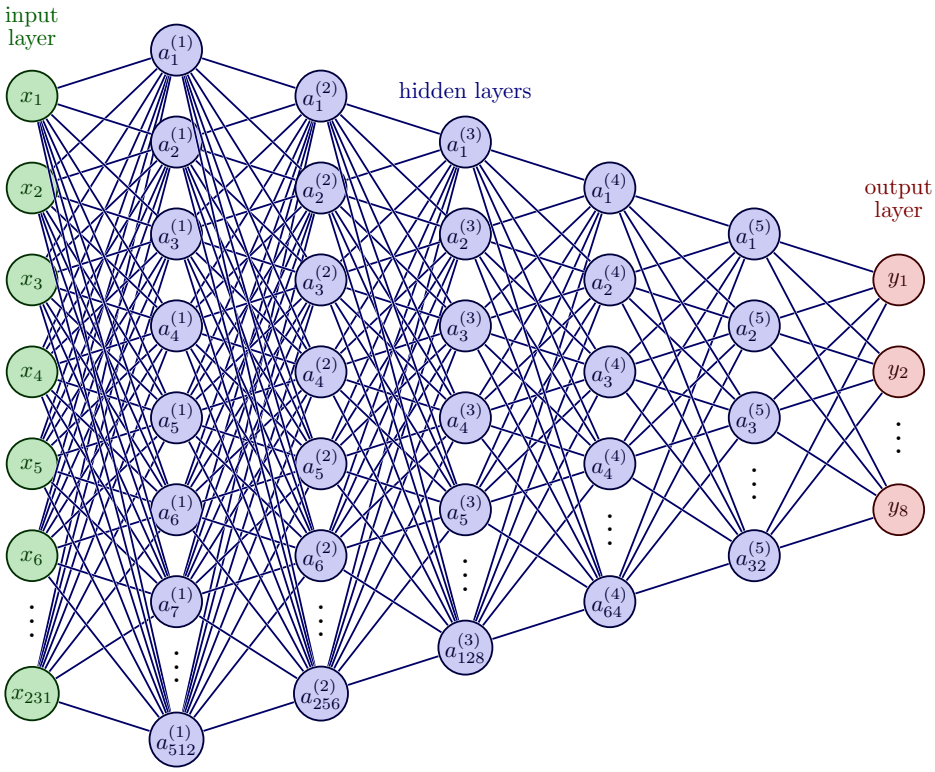

Figure 8.2: Architecture dipole with amplitude.

In the extension of the DiLoc network, we maintain the use of ReLU as the activation function in the first layer, and hyperbolic tangent for the hidden layers, as this architecture, combined with the chosen hyperparameter for all problems considered gave the most promising results. However, as explained above, we deem it appropriate to employ the Sigmoid activation function in the output layer.

For an overview of the overall parameters employed in the extended

DiLoc for localizing current dipoles with amplitude	
Hyperparameters	**Value**
Hidden layers	5
Optimizer	SGD
Learning rate (initial)	0.001
Momentum	0.35
Weight decay	0.1
Minibatch size	32
Dropout	0.5
Act.func in first layer	ReLU
Act.func in hidden layers	Tanh
Act.func in last layer	Sigmoid

model, please refer to Table **??**, which provides a summary of these essential elements.

8.1.5 Metrics of success

In this extended version of DiLoc, assessing network performance through standard train-validation-loss plots becomes less informative due to the normalization of target values and a more complex cost function. Reading off unitless loss values from plots, where the cost function value is plotted against epochs, provides little insight beyond whether the network is capable of decreasing the loss with an increasing number of training iterations.

When evaluating the network's performance on the test dataset, which still comprises 20,000 samples, it is therefore essential that the predictions outputted by DiLoc undergo denormalization. This enables us to facilitate a meaningful evaluation against the true target values. The denormalization process takes a straight forward approach, where we simply do the opposite operations from the once performed during normalization 8.1:

$$y_i = (x_i + \min(x))\,(\max(x) - \min(x)) \tag{8.8}$$

To address the performance of DiLoc, we establish threshold values that represent acceptable errors for a majority of predictions. In particular, we are interested in determining the percentage of samples for which the network predicts the Euclidean distance of one or more dipoles within specific thresholds—3 mm, 5 mm, 10 mm, and 15 mm, where 3 mm is considered optimal.

This you need to ask Torbjørn.... Mention/ add histograms Regarding amplitude and radius predictions, the analysis involves studying the percentage of samples where the network provides predictions with absolute errors equal to 1, 2, and 3 mAμ m, and 1, 3, and 5 mm. Providing a MAE for the amplitude equal to 3 mAμ m corresponds to an error of 30%,and a MAE

for the radius equal to 5 mm corresponds to an error of 33%, both of which are intuitively considered large errors. However, if we consider these target values in relation to potential clinical significance, a different perspective emerges. In real-world clinical cases, it could be of significant interest to discern whether a neuron source exhibits the characteristic of a small magnitude (ranging from 1 to 3 mAμ m), a medium magnitude (ranging from 3 to 6 mAμ m), or a strong magnitude (ranging from 4 to 10 mAμ m). Similarly, a small radius (ranging from 1 to 5 mm), a medium radius (ranging from 5 to 10 mm), or a large radius (ranging from 5 to 10 mm) might hold clinical significance when considering the underlying neuronal mechanisms.

This shift in perspective highlights the nuanced interpretation of errors and underlines the importance of clinical context in evaluating the performance of DiLoc. In this light, even what might initially appear as substantial errors can offer valuable insights into the behavior of neuronal sources within real-world scenarios.

In sum, the suite of error metrics, coupled with threshold-based assessments, facilitates an in-depth evaluation of the network's capabilities. This multi-faceted approach bridges the realms of machine learning principles and clinical applicability, encapsulating the overarching goal of achieving accurate, meaningful, and real-world clinical predictions.

8.2 Predicting Single Dipole Sources with Varying Magnitudes

In this section, we introduce the concept of various magnitudes for single current dipole sources, which adds an additional dimension to the output of DiLoc. Besides predicting the coordinates of the dipoles for each sample, the network now also estimates the magnitude of the dipole signals. In real-world scenarios, it might be of interest to not only pinpoint the source of the abnormal activity but also comprehend the extent of abnormality. By incorporating magnitude prediction into our network, we gain valuable insights into the problem at hand and achieve a deeper understanding of the underlying brain activity.

magnitude is too small ? include some sort of calculation of the stenght of the recording signal. Alternatively, explain that we find it good enough that magnitude is in the same scale/størrelsesskala. Weight decay is adjusted to 0.1.

8.2.1 Adjusting Data Set and Hyperparameters

We assign magnitudes to each dipole ranging between 1 and 10 nAμm. By now the dataset still has the same number of features, however the number of target values increases by 1. Figure 8.3 provides two examples from the

dataset, where the dipole location remains constant while the magnitude of the dipole signal varies. While we observe that the shape of the EEG signal remain consistent, the strenght of the EEG signal is significanly higher for the dipole with the largest initiated magnitude. It should therefor not be a problem for the network to separate such cases and it is fair to expect that the network is able to provide accurate predictions for the magnitude in both cases. From the figure it is also apperent that the EEG recordings ranges between -10 and 10 μV. Can I find litterature that support the range?

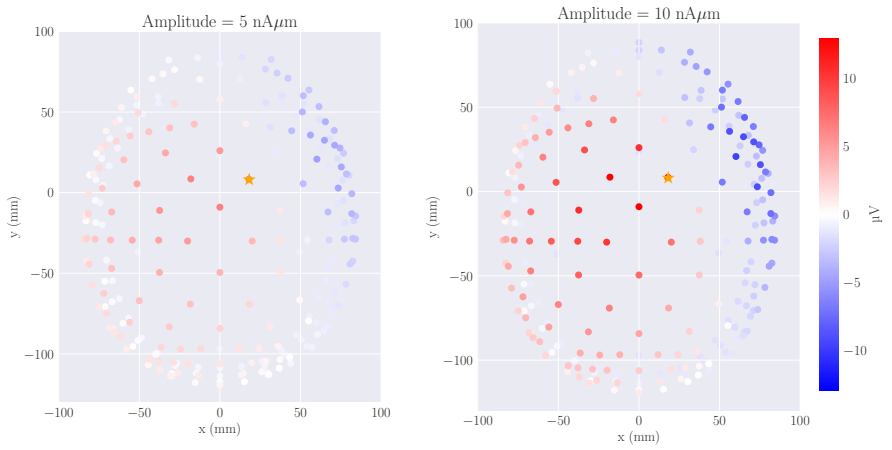

Figure 8.3: EEG data for two samples with current dipole amplitude equal to 5 and 10 nAμm. The EEG recordings have a range between -10 and 10 μV.

8.2.2 Performance Evaluation

To assess the network's performance, we start by analyzing the accuracy in relation to training epochs, as depicted in Figure 8.4. Once again we emphasize that it is important to note that the target values have been normalized, resulting in a unitless loss measurements. Therefore, the figure provides a qualitative representation of the network's training progress rather than precise loss values. The plot clearly demonstrates a consistent pattern of decreasing loss as the number of epochs increases, indicating that the network effectively captures the underlying data patterns. Moreover, both the training and validation loss stabilize after approximately 1100 epochs, suggesting that the network has reached its optimal performance level. Each epoch takes about 20 seconds to finish, leaving us with a training time of roughly 8.5 hours in total.

In Figure 8.5, we present the progression of the loss for the target parameters. Notably, after approximately 1100 epochs, all target values, exhibit

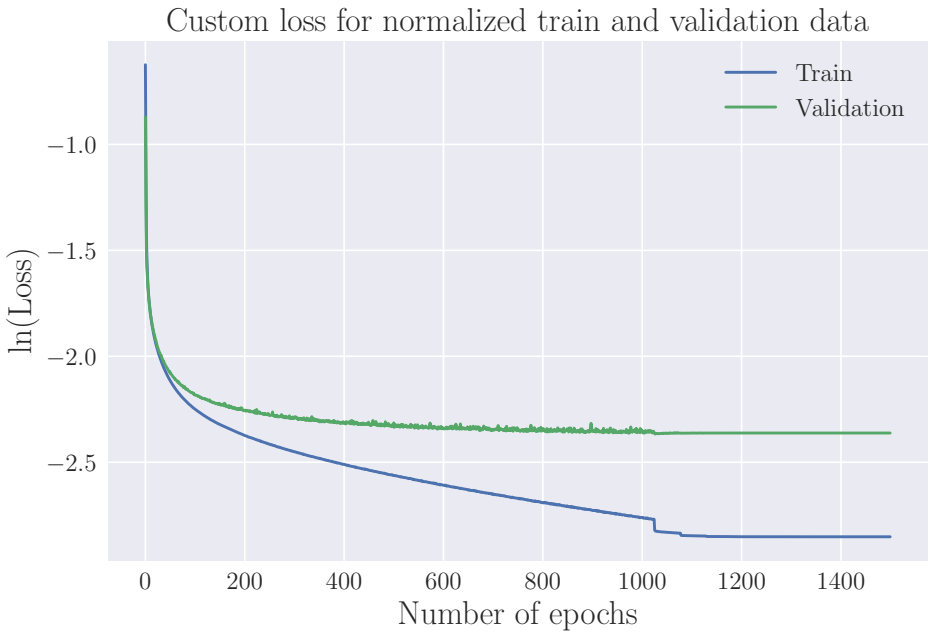

Figure 8.4: The loss for the extended DiLoc network with 50 000 samples and hyperbolic tangent activation function.

a cessation of fluctuation, suggesting potential full convergence at this juncture. It is evident that the loss for the magnitude target converges significantly earlier in the training process, compared to the target coordinates, although at a notably higher numerical value. In contrast, the x, y, and z-coordinate targets display ongoing improvement until approximately 1100 epochs, gradually converging to values that closely align with one another. Among these, the y-coordinate achieves the lowest loss, followed by the x-coordinate, and then the z-coordinate.

In Table 8.1, we present the network's performance across various target categories using distinct error metrics. Analyzing the outcomes pertaining to the target coordinates reveals a noteworthy consistency in mean absolute error, with an average deviation of less than 1.5 mm from the true values. This level of accuracy is highly satisfactory. Considering the range associated with each coordinate profile, we observe that the MAE for the x coordinate, at 1.348 mm, accounts for only 0.936% of the complete range. Similarly, for the y-coordinate, exhibiting a MAE of 1.448 mm, the error relative to the coordinate range stands at 0.809%. Lastly, the z coordinate, with a MAE of 1.411 mm, represents 1.052% of the full range. This balanced distribution of errors suggests that no single dimension to a large extent exhibits higher error propensity than the others, thereby indicating robust overall performance. It is important to note, however, that the z coordinate consistently exhibits the

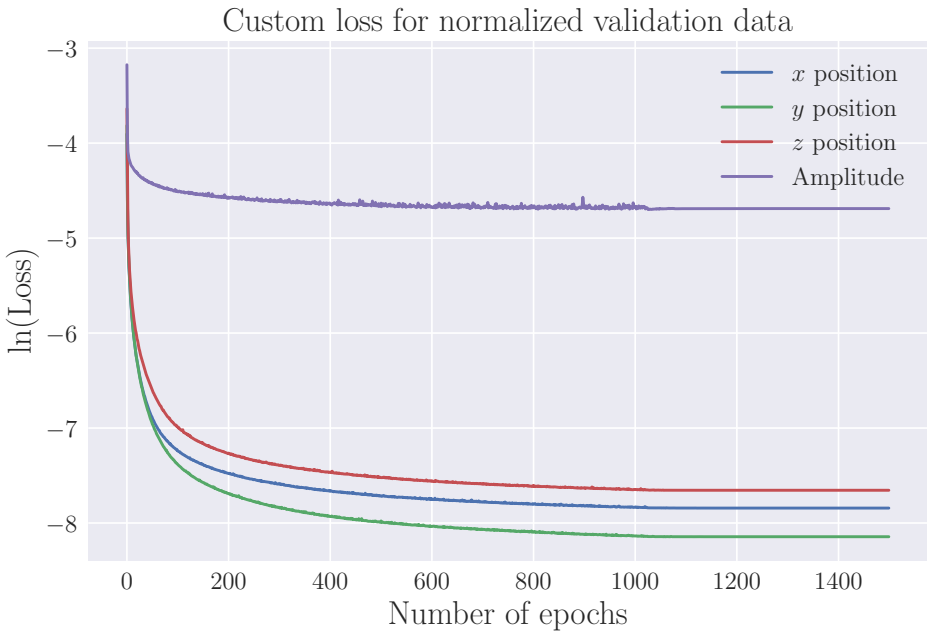

Figure 8.5: The loss developmnent for the different target values as function of epochs.

largest error, implying that specific challenges or characteristics may exist in predicting this dimension, resulting in slightly elevated errors. Moving on to the MAE for the magnitude variable, we ascertain a value of 0.539 nAμm. In the context of a magnitude range spanning from 1 to 10 nAμm, this MAE translates to a relative error of 6.00%, which is noticeably larger when compared to the MAE values for the target coordinates.

In addition to MAE, we look into the network's performance through the metrics of mean squared error and root mean squared error metrics, which offer complementary insights. For the x coordinate, the MSE stands at 3.438 mm^2, while for the y- and z-coordinate, it is 3.860 mm^2 and 3.862 mm^2, respectively. MSE, due to its quadratic nature, penalizes larger errors more prominently. Despite this, the MSE values remain within reasonable bounds when contextualized within the coordinate ranges, implying a stable error profile with few significant outliers. Turning our attention to the MSE for the magnitude target, which measures 0.650 nA$^2\mu$m^2, we observe that it is lower than the MSE values for the coordinate targets. This is a satisfying finding, especially considering the narrower range of the magnitude target. However, it is important to recognize that the theoretical lower limit of MSE is always 0, which means there is still room for further improvement in accurately capturing variations in all the target values.

Lastly, the Root Mean Squared Error (RMSE), which indicates the stand-

ard deviation of prediction errors and their distribution around the mean, reports values slightly lower than the corresponding Mean Squared Error (MSE) values. Specifically, RMSE values measure at 1.854 mm, 1.965 mm, and 1.965 mm for the distinct target coordinates, and at 0.806 nAμm for the magnitude target. These RMSE values suggest that, on average, the prediction errors align with the overall spread of errors, without significant outliers or extreme deviations from the mean error. This indicates a level of stability and predictability in the error distribution.

	Error Metrics for Target Values				
	x-coordinate [mm]	y-coordinate [mm]	z-coordinate [mm]	Position Error [mm]	Magnitude [nAμm]
MAE	1.348	1.448	1.420	1.405	0.539
MSE	3.438	3.860	3.862	3.720	0.650
RMSE	1.854	1.965	1.965	1.929	0.806

Table 8.1: **Evaluation of Diloc's performance utializing different Error Metrics.**
Performance for the extended DiLoc network on test dataset consisting of 1000 samples. The errors are measured using Mean Squared Error (MSE), Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE).

In Figure 8.6, we present the MAE and MSE metrics computed between predicted and target magnitude values, as a function of the magnitude of the dipole strengths. The figure reveals an interesting pattern, where the occurrence of outliers and less favorable predictions is higher when dealing with identification of dipoles with smaller magnitudes.

This apparent trend might initially suggest that the neural network encounters greater challenges in accurately predicting smaller magnitudes. However, we cannot rule out that these disparities might be related to alternative factorsdepth positioning, intricate cortical folding patterns, or other random fluctuations. Nevertheless, the most pronounced trend that emerges from the figure is the prevalence of predictions exhibiting a MAE smaller than 5 nAμm and a MSE smaller than 10 nA$^2\mu$m^2. This trend is of particular significance, serving as a reassuring indicator. It suggests that, despite observed variations in predictions for samples with smaller magnitudes, the majority of our Diloc's predictions consistently cluster within a range with acceptable errors.

The Mean Euclidean Distance (MED) of DiLoc's predictions on the unseen test data is measured at 2.815 mm. This represents a value more than 100% larger than the MED obtained when DiLoc predicted only the dipole locations for dipoles with a constant magnitude of electrical signal. Nevertheless, it is crucial to underscore that, given the dimensions of the brain within the NYHM, this error remains relatively small and satisfactory.

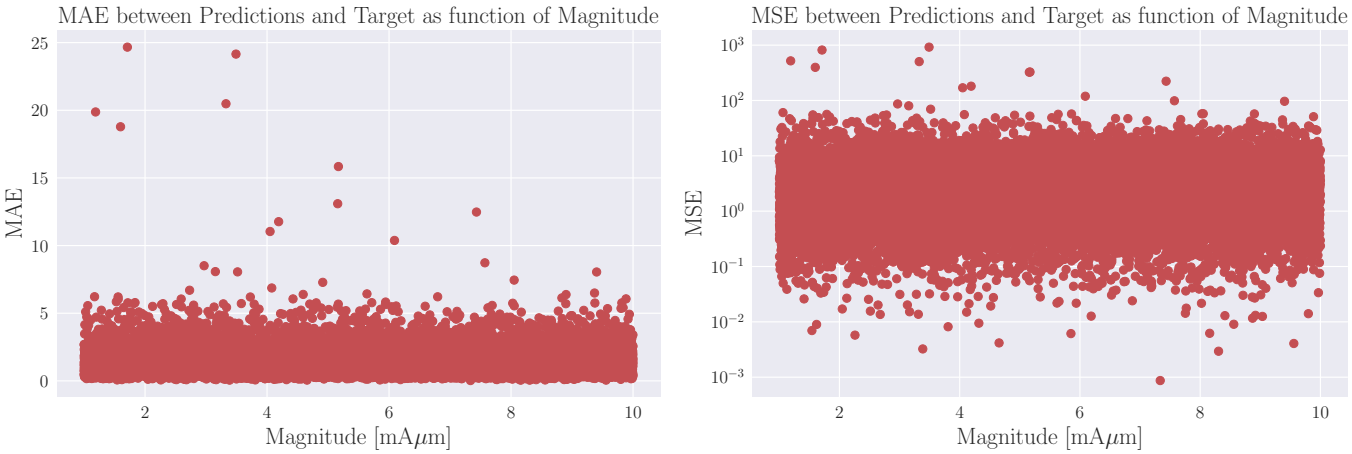

Figure 8.6: Scatter plot of Mean Abolsute Error and Mean Squared Error computed between predicted and target magnitude values, as a function of the magnitude of the dipole strengths.

Mean Euclidian Distance for Test Samples			
MED <3 mm	MED <5 mm	MED <10 mm	MED <12 mm
64.325 %	90.930 %	99.505 %	99.820 %

Table 8.2: **MED evaluation on test samples; Predicting Location and Magnitude**
Performance of the network on thes test dataset comprising 20,000 samples, presented as the percentage of samples falling within MED thresholds of 3 mm, 5 mm, 10 mm, and 12 mm, respectively.

Table 8.2 presents the percentages of samples falling within various MED threshold values. Notably, the network achieves an accuracy smaller than 12 mm for 99.8% of the test data. Furthermore, 64.3% of the samples exhibit a MED smaller than 3 mm, an achievement that, while robust, falls slightly short of DiLoc's performance in the previous, less complex problem.

Figure 8.7 displays two panels, depicting the MED and Mean Absolute Error (MAE) for the magnitude of each test sample within bins of width 1 unit. In the left panel, representing the MED histogram, the bins corresponding to MED values of 2 and 3 mm are the most populated, containing the largest proportion of samples. This panel also illustrates that the majority of predicted locations exhibit a MED error smaller than 14 mm, which is a satisfactory outcome.

Turning our attention to the right panel, which displays the MAE for predicted magnitudes, we observe that the bin holding samples with a magnitude prediction MAE less than 1 nAμm is the most populous. Within this bin, 17,014 out of the 20,000 samples fall, signifying that the network

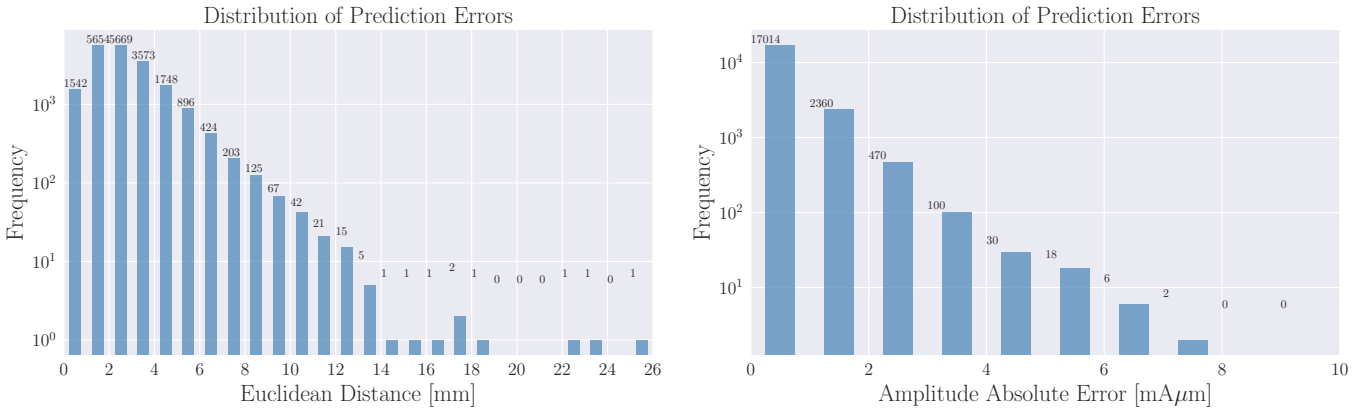

Figure 8.7: Left panel illustrates the distribution of Mean Euclidean Distance for the predicted dipole locations, organized into bins of width 1 mm. Right panel holds the distribution of Mean Absolute Error for the predicted magnitude of dipole electrical signal, presented in a similar format, with bins of width 1 nAμm.

predicts the magnitude with a MED smaller than 1 nAμm for approximately 85% of the sample set.

8.3 Predicting Region of Active Correlated Current Dipoles with Amplitudes

In order to further enhance the complexity of our problem, we extend the DiLoc neural network to incorporate varying radii and magnitudes for the origins generating the electrical activity detected by the recording electrodes. This transformation alters the objective of the DiLoc network from predicting the location of individual current dipole moments to estimating the centers of larger spherical populations. This extension is hopefully valuable for real-life scenarios where understanding the extent of brain damage causing abnormal activity in damaged areas may be of interest. By training the DiLoc network on such complex data, we aim to enhance its ability to generalize and perform effectively in real-world clinical cases.

8.3.1 Adjusting Data Set and Hyperparameters

Ensure realism...? For the purpose of enabling the network to predict the areas of dipole populations, we make adjustments to the dataset. The dipole populations are represented as spherical volumes in the NY head cortex, with the radius for each population ranging from 1 mm to 15 mm. To ensure realism, we maintain the maximum amplitude strength of the total

populations at 10 mAμm. Consequently, we calculate the maximum number of points within a volume sphere with a radius of 15 mm and reduce this number by 10 to determine the strength of each dipole within the given area. This leaves us with a strength of 10/899 for each dipole. The strength of a dipole population is thus directly proportional to the size of the dipole population. While this may not perfectly represent real-world scenarios, it provides a reasonable approximation for our model.

In Figure 8.8, we present an example of a dipole population and the corresponding EEG signal. The yellow filled circles in the plots in the upper panel represents the diple populations, i.e. positions within the cortex where dipoles have been placed. The lower panel shows the EEG signals for the specific sample, with EEG electrode locations presented as filled circles, where the color of the fill represents the amplitude of the measured signal for the given electrode. The plots within the figure are seen from both the x-z plane, x-y plane, and the y-z plane.

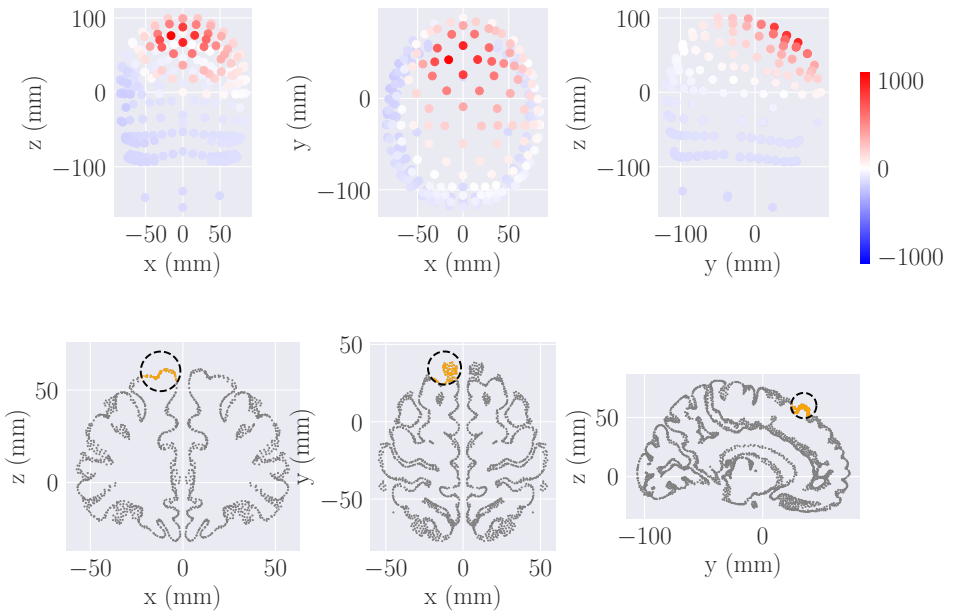

Figure 8.8: EEG for a sample containing a spherical population of current dipole sources with a random center within the cerebral cortex. The EEG measure is seen from both sides (x-z plane and y-z plane) and above (the x-y plane). EEG electrode locations are presented as filled circles, where the color of the fill represents the amplitude of the measured signal for the given electrode.

As for the dataset, the number of target values is now 5: x, y, z-coordinates of the center of the dipole population, amplitude, and radius. The number of features is not modified and still holds the number of 231,

representing the recording electrodes. The new architecture of the DiLoc network is presented in Figure **??**.

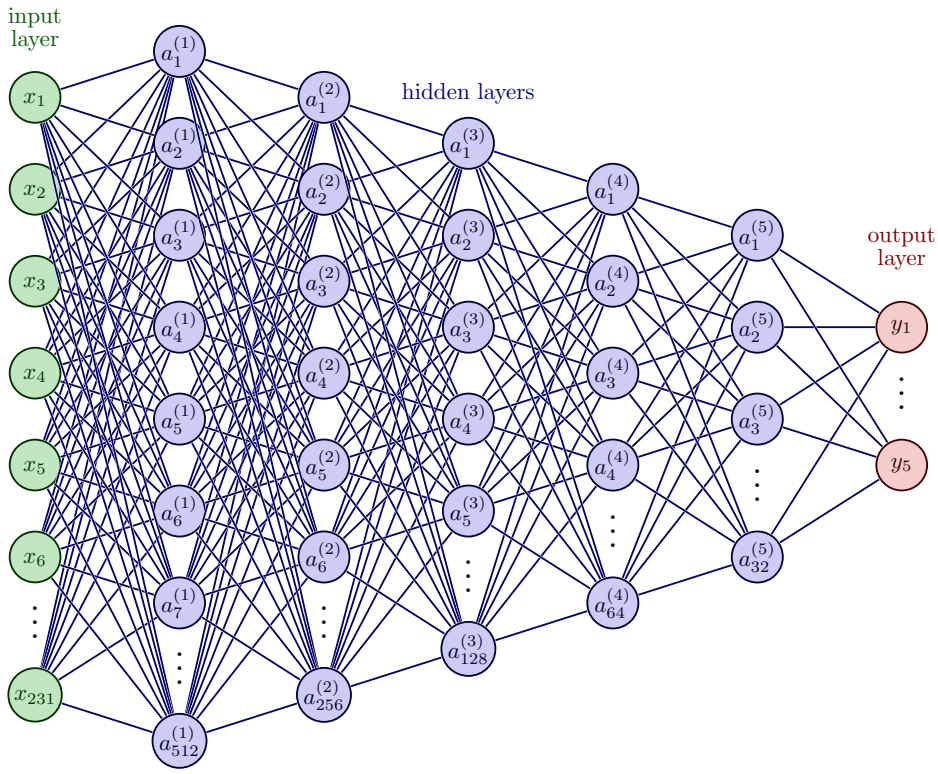

Figure 8.9: Architecture of the dipole area prediction network.

Similar to the previous problem, we normalize the target values to ensure they all range from 0 to 1. Moreover, in this extension of the DiLoc network, we use the same activation functions as in the previous problem with ReLU as the activation function in the first layer, hyperbolic tangent for the hidden layers, and the Sigmoid activation function in the output layer. As with the previous problems, we have explored various network architectures and activation functions, but the current configuration has shown the best performance in terms of accurate predictions for this problem. It is important to emphasize that our primary goal is to find a network that can effectively solve the problem and provide accurate predictions, rather than necessarily seeking the best possible configuration.

8.3.2 Performance Evaluation

In Figure **??**, we present the training and validation Mean Squared Error (MSE) loss for the ConvDip network as a function of epochs. The network was trained for 5000 epochs, but the validation loss appears to converge

at around 3000 epochs, while the training loss stabilizes at approximately 4000 epochs. Notably, there are no signs of overfitting, which is a positive outcome. Each epoch took approximately 7 seconds, resulting in a total training period of approximately 9 hours.

Figure 8.11 displays the validation loss for each target value as a function of epochs. The losses for all coordinate target values (x, y, and z) are minimized almost identically, with the y-coordinate loss slightly smaller. The amplitude target value is minimized most effectively by ConvDip, while the radius target value has the highest loss. We keep in mind that the amplitude and radius taget value correlates to some extent, as the amplitude is proportional to the radius. Although the exact MSE values for the target values cannot be directly read from the figure due to normalization, the overall pattern indicates that ConvDip successfully captures data patterns and minimizes the cost function.

Figure 8.10: The validation accuracy for the simple Feed Forward Neural Network, predicting both center and radius for 50 000 samples, for 5000 epochs, with a learning rate equal to 0.001.

To assess the extent to which the network can predict the center of the dipole populations, in addition to amplitude and radius, we utilize the same evaluation metrics as described in chapter 6. Table 8.3 presents the Mean Absolute Error (MAE), Normalized MEan Absolute Error (NMAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) for the different sets of target parameters.

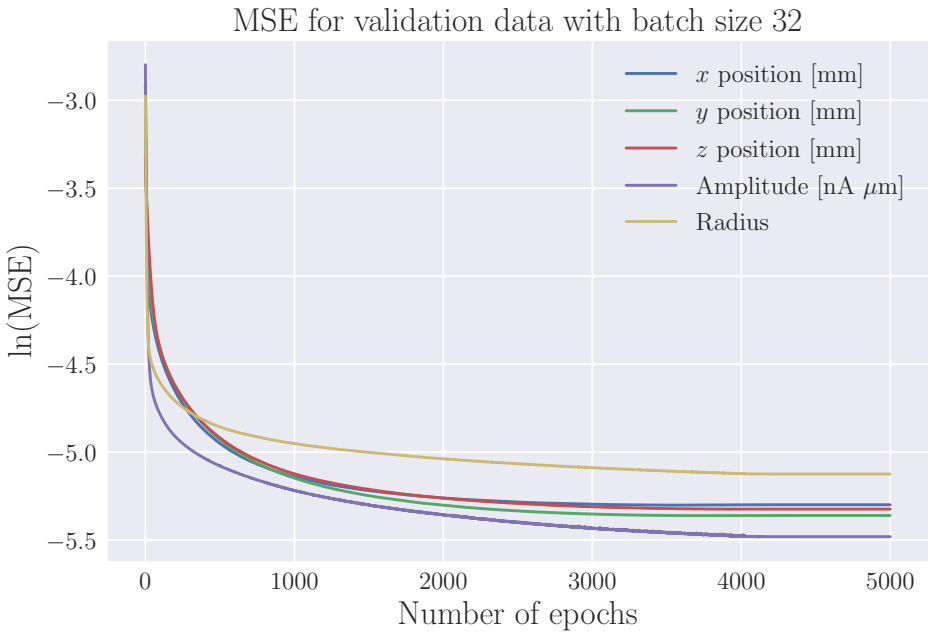

Figure 8.11: The validation accuracy as function of epoch for each target value: x, y, z-coordinates of the center of the dipole population, amplitude, and radius.

The MAEs for all coordinates and the Euclidean distance lie between 4 and 5 millimeters. Looking at the NMAE, we observe that the loss for the z-coordinate is somewhat larger than for the other coordinates, with 3 %, similar to our observations when testing DiLoc's ability to predict amplitude in addition to location. However, this slightly larger error is not significant and may as well be attributed to randomness. What is worth mentioning is that all cordinates

As for the amplitude and radius targets, the MAEs are remarkably small. For amplitude, ranging from 1 to 10 mAμm, the absolute error is approximately 4.33% of the range of the actual amplitude values, indicating that the model's amplitude predictions are reasonably close to the true amplitude values. Similarly, the MAE for the radius, with a range from 1 to 15 mm, is approximately 6.07%, suggesting that the model's predictions are relatively accurate for radius.

Regarding the MSE, we observe relatively small errors for the amplitude and radius, with values of 0.364 and 1.291 (mm^2) respectively. However, as for the coordinate target values, we encounter relatively larger MSE values. This difference in scale between the MAE and MSE suggests the presence of outliers.

	Error for different target values					
	x [mm]	y [mm]	z [mm]	Center [mm]	Amplitude [nAμm]	Radius [mm]
MAE	4.257	4.868	4.126	4.417	0.390	0.850
NMAE	2.955	2.711	3.083	2.916	4.333	6.071
MSE	47.141	68.776	46.192	54.036	0.364	1.291
RMSE	6.866	8.293	6.796	7.351	0.604	1.136

Table 8.3: **Evaluation of DiLoc utilizing different Error Metrics.** Performance of the extended DiLoc network on a test dataset consisting of 20000 samples. The errors are measured using Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) for various target values.

8.4 Localizing Multiple Dipole Sources

In this final extension of the DiLoc neural network we want to train the model in predictinf the positions of not just one but two individual dipole sources, which collabraticely generate the recored EEG signal. This novel extension pushes the boundaries of the network's capabilities, requiring it to grapple with the complex task of identifying and localizing multiple distinct dipole sources within the brain.

8.5 Previous work

We acknowledge that similar research has been conducted by other groups, including the developers of the ConvDip convolutional neural network. The ConvDip network was designed to produce inverse solutions for EEG data, specifically focusing on predicting the positions of varying numbers of sources from a single time point of EEG data.

The researchers behind ConvDip explored the feasibility of utilizing CNNs to solve the EEG inverse problem for multiple sources using training data that adheres to biologically plausible constraints. Similar to DiLoc, ConvDip was trained to operate on single time instances of EEG data and predict the positions of sources based on potentials measured with scalp electrodes. However, it is worth noting that unlike our approach, the ConvDip group considered dipole clusters rather than single dipoles. This approach aligns more closely with the previous problem in which we focused on dipole populations.

For generating the simulated data, the researchers created a source model consisting of 5124 dipoles distributed along the cortical surface (also referred to as the cortex). They selected 31 recording electrodes and computed the leadfield matrix using a head model with dipole orientations fixed ortho-

gonally to the cortical surface, similar to our methodology. To enhance the realism of the training data, real noise from pre-existing EEG recordings conducted with the same set of electrodes was added. Additionally, the group created separate test data using an alternative head model to avoid potential overoptimistic results, a phenomenon they referred to as the "inverse crime." The training dataset consisted of 100,000 samples, while the test dataset comprised 1000 samples.

In order to prepare the EEG input data for spatial convolutions, it was interpolated onto a 2D image of size 7 x 11. As expected with interpolation, this procedure does not introduce new information to the EEG data. The output of ConvDip is a vector of size 5,124, corresponding to the dipoles in the source model. For a comprehensive description of the ConvDip network, we refer readers to the paper: paper: https://www.frontiersin.org/articles/10.3389/fnins.2021.569918/full.

Although the complexity of our original DiLoc network (FFNN) is significantly smaller compared to ConvDip, we still desired to investigate its performance in this more challenging task.

INCLUDE THIS We will now evaluate the ability of ConvDip to estimate the correct size of sources and to correctly localize sources with varying depth.

8.5.1 Adjustments in Data Set and Architecture

To begin with, we simulate EEG data corresponding to the electrical signals originating from multiple individual dipoles located in the brain. Initially, we allow unrestricted distances between the dipole sources. However, to avoid overcomplicating the problem, we assign each dipole within a sample with the same magnitude of amplitude. Consequently, for the dipole population problem, the total amplitude for a set of dipoles is fixed at 10 mAμm. Figure 8.12 displays two plots of randomly selected samples, illustrating the simulated EEG data when multiple dipoles generate the signal. In the first sample, two dipoles generate the EEG signal, each having an amplitude of 2.4 mAμm. In the second sample, three dipoles generate the EEG signal, and each dipole has an amplitude of 1.13 mAμm.

The total number of target values for this problem has increased to 8, encompassing the x, y, and z-coordinates for the location, as well as the amplitude, of each dipole. Since we constrained each dipole within a sample to have the same amplitude, it is not necessary to have separate output values for the amplitudes of each dipole. Nevertheless, we modified the architecture of the network, considering the possibility of outputting amplitude target values with varying values for each dipole. Apart from this adjustment, the network still comprises 231 input nodes, and the target values have been normalized to range from 0 to 1. The logic and choice of activation functions, as well as hyperparameters, remain consistent with those used in previous problems. Figure 8.13 illustrates the updated network architecture.

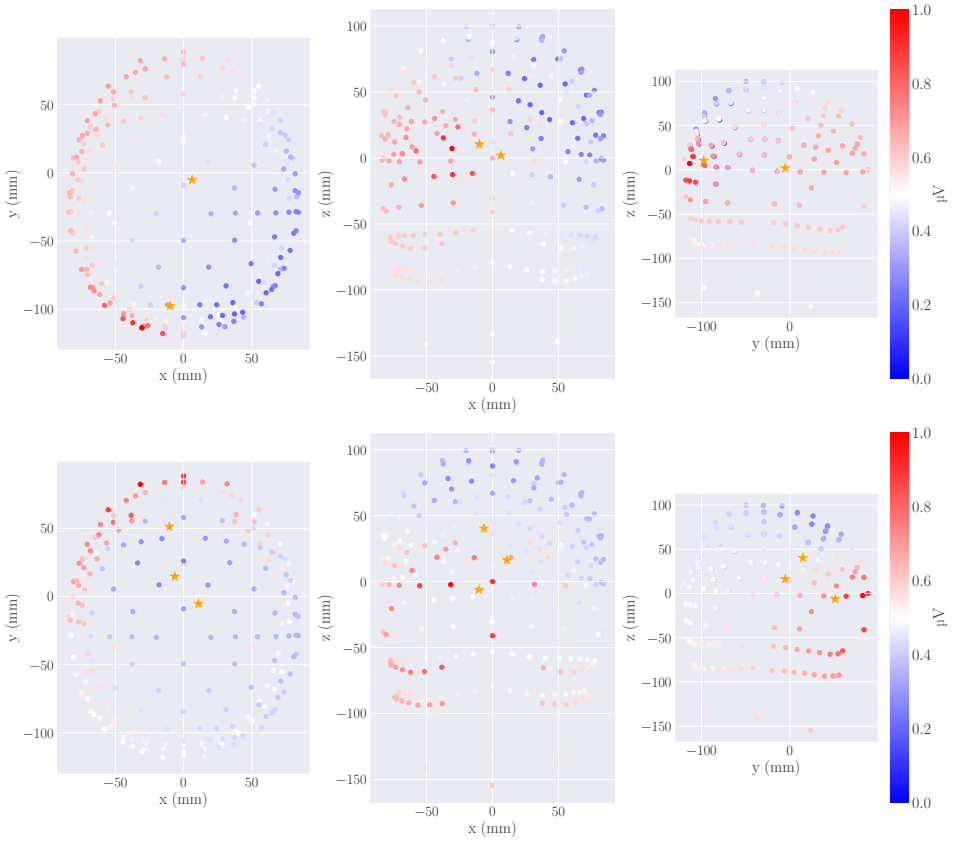

Figure 8.12: EEG for two samples containing two and three current dipole sources, respectively, at random positions within the cerebral cortex. The EEG measures are seen from both sides (x-z plane and y-z plane) and from above the skull (x-y plane). EEG electrode locations are presented as filled circles, where the color of the fill represents the amplitude of the measured signal for the given electrode. The positions of the current dipole moments are marked with yellow stars.

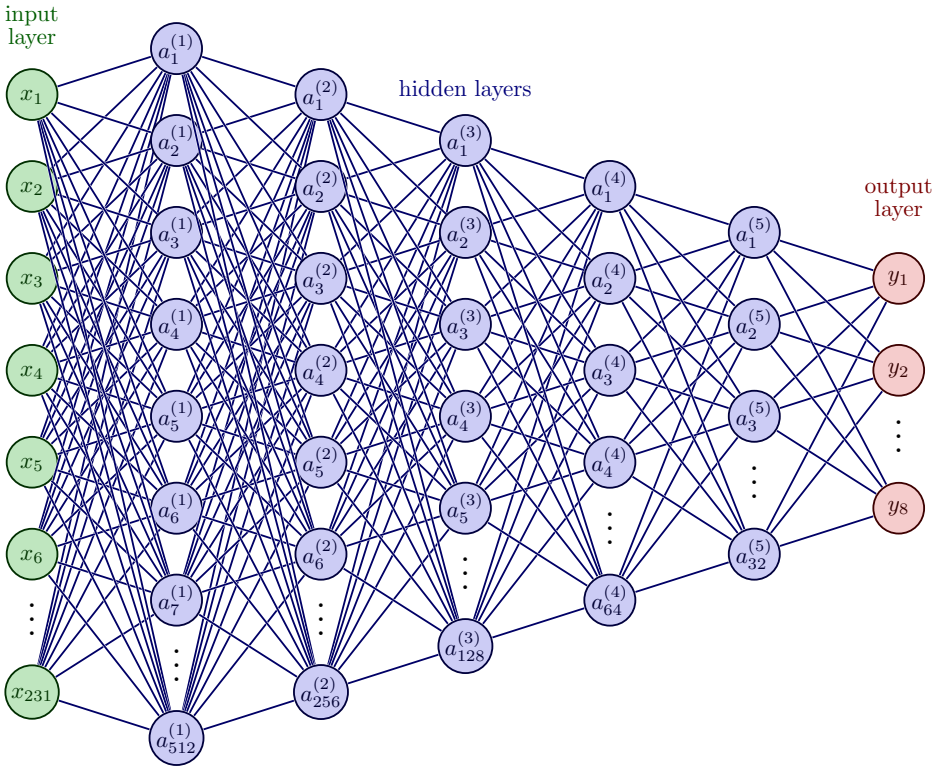

Figure 8.13: Architecture of the multiple dipoles network.

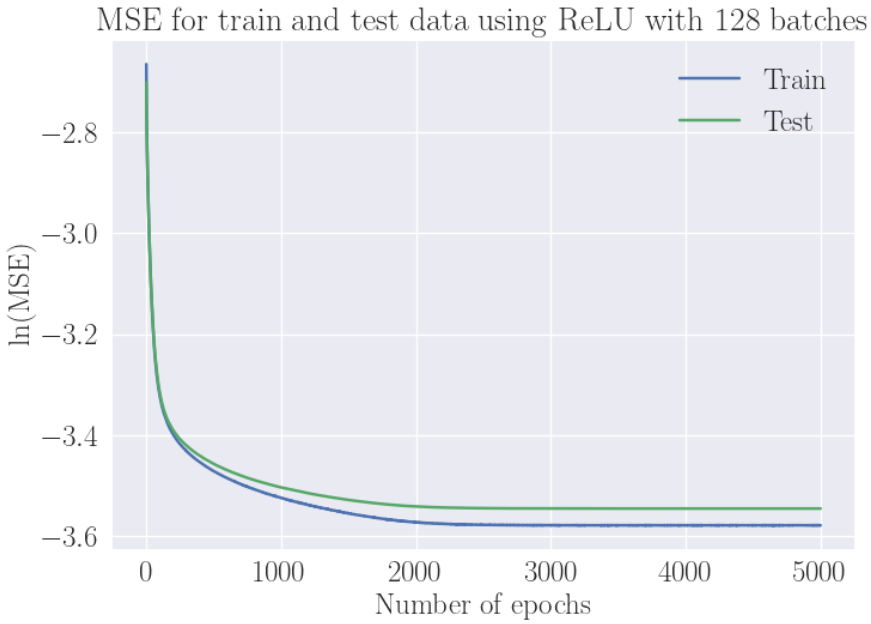

Figure 8.14: The validation accuracy for the simple Feed Forward Neural Network, predicting two current dipole sources.

Bibliography

[1] Wulfram Gerstner et al. *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, 2014.

[2] David Sterratt et al. *Principles of computational modelling in neuroscience*. Cambridge University Press, 2011.

[3] Wikipedia contributors. *Electroencephalography*. Accessed on 29 July 2023. 2023. URL: https://en.wikipedia.org/wiki/Electroencephalography.

[4] Edward B Bromfield, José E Cavazos and Joseph I Sirven. 'An introduction to epilepsy [Internet]'. In: (2006).

[5] Lukas Hecker et al. 'ConvDip: A convolutional neural network for better EEG Source Imaging'. In: *Frontiers in Neuroscience* 15 (2021), p. 569918.

[6] Solveig Næss et al. 'Biophysically detailed forward modeling of the neural origin of EEG and MEG signals'. In: *NeuroImage* 225 (2021), p. 117467.

[7] Yu Huang, Lucas C Parra and Stefan Haufe. 'The New York Head—A precise standardized volume conductor model for EEG source localization and tES targeting'. In: *NeuroImage* 140 (2016), pp. 150–162.

[8] Michele Giugliano, Mario Negrello and Daniele Linaro. *Computational Modelling of the Brain: Modelling Approaches to Cells, Circuits, and Networks*. Springer, 2022.

[9] Estimation lemma. *Estimation lemma — Wikipedia, The Free Encyclopedia*. [Online; accessed 22-May-2023]. 2023. URL: https://en.wikipedia.org/wiki/Multipole_expansion.

[10] John David Jackson. *Classical electrodynamics*. 1999.

[11] Espen Hagen et al. 'Multimodal Modeling of Neural Network Activity: Computing LFP, ECoG, EEG, and MEG Signals With LFPy 2.0'. In: *Frontiers in Neuroinformatics* 12 (2018). Original Research Article, p. 92. ISSN: 1662-5196. DOI: 10.3389/fninf.2018.00092. URL: https://doi.org/10.3389/fninf.2018.00092.

[12] Mona Sazgar et al. 'EEG artifacts'. In: *Absolute Epilepsy and EEG Rotation Review: Essentials for Trainees* (2019), pp. 149–162.

[13] Bitbrain. *EEG Artifacts*. Bitbrain Blog. 2020.

[14] Wikipedia. *Signal-to-noise ratio*. Wikipedia: The Free Encyclopedia. 2023.

[15] Pankaj Mehta et al. 'A high-bias, low-variance introduction to machine learning for physicists'. In: *Physics reports* 810 (2019), pp. 1–124.

[16] Morten Hjorth-Jensen. 'Neural Networks'. In: *Department of Physics, University of Oslo, Norway* (Oct. 2022). [1] Department of Physics, University of Oslo, Norway

Department of Physics and Astronomy and Facility for Rare Ion Beams, Michigan State University, USA.

[17] Chigozie Nwankpa et al. 'Activation functions: Comparison of trends in practice and research for deep learning'. In: *arXiv preprint arXiv:1811.03378* (2018).

[18] Sagar Sharma, Simone Sharma and Anidhya Athaiya. 'Activation functions in neural networks'. In: *Towards Data Sci* 6.12 (2017), pp. 310–316.

[19] Rukshan Pramoditha. 'How to Choose the Right Activation Function for Neural Networks'. In: (Jan. 2022). URL: https://towardsdatascience.com/how-to-choose-the-right-activation-function-for-neural-networks-3941ff0e6f9c (visited on 10/06/2023).

[20] Varun Bhatia. *Activation Functions: How Should the Neurons Trigger?* Medium. [Online]. Available: https://medium.com/analytics-vidhya/activation-functions-how-should-the-neurons-trigger-1349a383ffeb. 2020.

[21] Shweta Saxena. *Activation Functions Compared With Experiments*. 2022. URL: https://wandb.ai/shweta/Activation%20Functions/reports/Activation-Functions-Compared-With-Experiments--VmlldzoxMDQwOTQ (visited on 10/06/2023).

[22] Sparch Gupta. *The 7 Most Common Machine Learning Loss Functions*. URL: https://builtin.com/machine-learning/common-loss-functions (visited on 16/08/2023).

[23] Vic Barnett, Toby Lewis et al. *Outliers in statistical data*. Vol. 3. 1. Wiley New York, 1994.

[24] Kai Zhang and Minxia Luo. 'Outlier-robust extreme learning machine for regression problems'. In: *Neurocomputing* 151 (2015), pp. 1519–1527.

[25] Wikipedia contributors. *Gradient descent — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/wiki/Gradient_descent. [Online; accessed 31-May-2023]. 2023.

[26] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*. Vol. 4. 4. Springer, 2006.

[27] Prashant Gupta. 'Regularization in Machine Learning'. In: *Towards Data Science* (Nov. 2017). URL: https://towardsdatascience.com/regularization-in-machine-learning-76441ddcf99a.

[28] Adrian Tam. *Using Learning Rate Schedule in PyTorch Training*. Last Updated on April 8, 2023. 2023. URL: https://machinelearningmastery.com/using-learning-rate-schedule-in-pytorch-training/.

[29] NumPy Developers. *Numpy Corrcoef*. NumPy. 2023. URL: https://numpy.org/doc/stable/reference/generated/numpy.corrcoef.html (visited on 12/09/2023).