UiO : Department of Physics
University of Oslo

Faculty of Mathematics and Natural Sciences

Neuromorphic computing for GABA detection

Thesis
for the degree of
Master of Electronics, Informatics and Technology

Abdulkadir Hassen Ali

March 2023

Abstract

The goal of this thesis is to investigate the potential of Neuromorphic Computing for detecting the neurotransmitter GABA levels in the human brain. Neuromorphic Computing is a novel approach to computing that mimics the functioning of the brain, offering energy-efficient real-time processing for AI applications. In contrast to traditional computing architectures, Neuromorphic Computing offers several advantages in many applications, including parallel processing, unsupervised learning, and real-time data processing.

To demonstrate the potential of Neuromorphic Computing for GABA detection, the thesis will first implement traditional neural networks to analyze the feasibility of detecting GABA levels and compare the actual experimentally measured GABA levels. Traditional neural networks are known for their ability to model complex relationships between inputs and outputs, and for their generalization capability, which means they can perform well on unseen data. They have been widely used in various applications. However, traditional neural networks are computationally expensive and require a significant amount of data and power to perform their calculations. This can make them unsuitable for certain applications, such as those that require real time processing or those that operate on battery-powered devices where energy efficiency is crucial, and there is an insufficient amount of data. This has led to the development of alternative computing architectures, such as Neuromorphic Computing, which includes Spiking Neural Networks, as a potential solution to overcome these limitations.

Spiking Neural Networks (SNNs) are a growing trend in the field of AI and Machine Learning due to their unique approach to data processing. Unlike traditional artificial neural networks, SNNs operate in the spike domain and have the potential to be more energy-efficient and provide real-time data processing. Although much research is still needed to fully comprehend the capabilities and limitations of SNNs, it is possible to train these algorithms on the same datasets used for traditional neural networks. With their potential for energy efficiency and real time data processing, SNNs are considered a promising development for a wide range of applications.

The fact that there are so many interfacing tools and systems that can interface between the neuromorphic chip and the measuring system.The Tkinter framework is a popular and widely used graphical user interface (GUI) library in Python, that makes an ideal choice for many applications. It allows easy transmission of data between the memory unit and processing unit such as Akida Neuromorphic Processor.The system will be designed in such a way that it can be easily modified or expanded in the future if needed. The Tkinter framework provides a range of tools and functions to create a responsive and intuitive GUI, making it a perfect fit for this project. The aim is to develop a system that is not only effective in terms of functionality but also user-friendly, allowing for efficient and seamless access to previously recorded data and analyzing it before deploying to the hardware.Although DAK-3.5 is a window operating system and Akida1000 is a Linux operating system, but both can be interfaced by a python programmed system.

Acknowledgements

I would like to thank Professor Ørjan Grøttem Martinsen for providing me with the opportunity,and for his consistent support, and encouragement to work on such an intriguing project. Without his guidance and backing, I would not have been able to deliver and complete this project.

I would also like to express my sincere appreciation to my Assistant Supervisor Jie Hou. Her expertise and experience in the field have been vital to the success of this project. Her mentorship and support have been instrumental in helping me understand the various complexities involved and execute the project with precision. I am immensely grateful to her for providing me with all the necessary resources, guidance and time to perform the required measurements at her workplace.

I am grateful to my brothers Omar Hassan Ali and Abukar Hassan Ali and my closest friend Nur Mohamed Farah who helped with some technical support. Their contributions and support have been instrumental in shaping the final outcome of this project and I am deeply grateful to them.

I would also like to extend my sincerest gratitude to everyone who has supported and encouraged me throughout this project. Their contributions, insights, and critiques have been fundamental to the successful completion of this project, and helped me refine my ideas and approach, leading to a better final product.
Lastly, I want to express my gratitude to my parents, my wife and my children for their unwavering support and dedication throughout my journey. Their commitment and encouragement have provided me with the opportunities to grow both personally and professionally and have been instrumental in helping me achieve my goals.

Contents

List of Abbreviations

AI	Artificial Intelligence
BIS	Bioimpedance spectroscopy
BMI	Brain Machine Interfaces
CNS	Central Nervous System
CPU	Central Processing Unit
DAK	Dielectric Assessment Kit
DMA	Direct Memory Access
ECG	Electrocardiogram
EEG	Electroencephalography
FFNN	Feedforward Neural Networks
FPGA	Field Programmable Gate Array
GABA	Gamma-Aminobutyric Acid
GPU	Graphics Processing Units
GUI	Graphical User Interface
IoT	Internet of Things
LSTM	Long Short Term Memory
MAE	Mean Absolute Error
MI	Motor Imagery
ML	Machine Learning
MRI	Magnetic Resonance Imaging
MSE	Mean Squared Error
NPUs	Neural Processing Units
OL	Online learning
OLS	Ordinary Least Squares
RMSE	Root Mean Squared Error
SNNs	Spiking Neural Network
SVMs	Support Vector Machines
VNA	Vector Network Analyze

List of Figures

Chapter 1

Introduction

The measurement of gamma-aminobutyric acid (GABA) concentration levels is a crucial aspect of understanding the functioning of the central nervous system (CNS). GABA is a neurotransmitter that acts as an inhibitory signal in the brain, slowing down the activities of nerve cells and reducing stimulation. It is responsible for controlling the excitability of the brain, and disruptions in its concentration levels have been associated with several neurological disorders, including epilepsy, anxiety, and depression. The accurate measurement of GABA concentration levels is essential for understanding the underlying biological mechanisms of these disorders and developing effective treatments[40].

The purpose of this study is to examine the use of dielectric relaxation spectroscopy (DRS) to measure GABA concentration levels. DRS is a non-invasive technique that measures the dielectric properties of a material in response to an alternating electrical field. The knowledge gained from DRS on pure GABA solutions in deionized water for different concentrations, and a constant temperature of 22°C has contributed to the advancement of this thesis. Additionally, this can further help the field to develop hardware that is more energy efficient.

The primary objective of the study is to develop a model that could accurately predict GABA concentration levels using the data collected by DAK-3.5. These data will be applied to a neural network, which is then converted to an Akida-compatible network which is an event-based neural network. The results of this study will provide insights into the feasibility and challenges of deploying neuromorphic computing to measure GABA concentration levels using the current available neuromorphic kit in the market The Akida 1000.

There is still much to be learned about the role of GABA in the brain, but what is clear is that this neurotransmitter plays a critical role in regulating brain activity and is involved in many important physiological processes. The determination of GABA levels has been a topic of research in the field of neuroscience for many years, and various studies have been carried out using different methods For example, Some scientists have used brain imaging techniques such as magnetic resonance imaging (MRI) to examine the levels of GABA in the brains of individuals with epilepsy, depression, and anxiety disorders[64]. while we will use dielectric relaxation spectroscopy techniques to collect data from concentrations of GABA and use to artificial neural network [16]. This new method for determining physiological GABA concentration uses machine learning and a computer-based Vector Network Analyzer (R140) and an Open Coaxial probe (DAK-3.5). It is important to note that the accuracy and reliability of the new method will depend on the quality of the data obtained from the R140 Vector Network Analyzer and the DAK-3.5 Open Coaxial probe, as well as the accuracy and effectiveness of the machine learning algorithms used.

The detection of GABA levels in the brain has been a challenging task due to the complexity of the underlying signals and the difficulty in accurately measuring GABA levels in vivo. There have been growing applications of deep learning algorithms on dielectric relaxation spectroscopy data to learn the relationship between the measured signals and the actual GABA concentration in the solution.
Researchers are still working to develop a hardware architecture that can accurately mimic the physiological information processing of GABA in the human brain. Several types of sensors that mimic bio-organisms such as e-skin[49],electronic nose and bio-inspired cameras have been developed. They all require real time computation with a minimum amount of energy. The current computer architectures, is based on the Von Neumann model, and has significant limitations in terms of energy efficiency and processing speed[46]. In the Von Neumann architecture, the memory and processing units are separate entities. This means that when computations are performed, data must be constantly sent back and forth between the memory and the processor. This continuous communication consumes a large amount of energy and leads to significant delays in processing time. The energy efficiency of this traditional computing system is therefore limited.

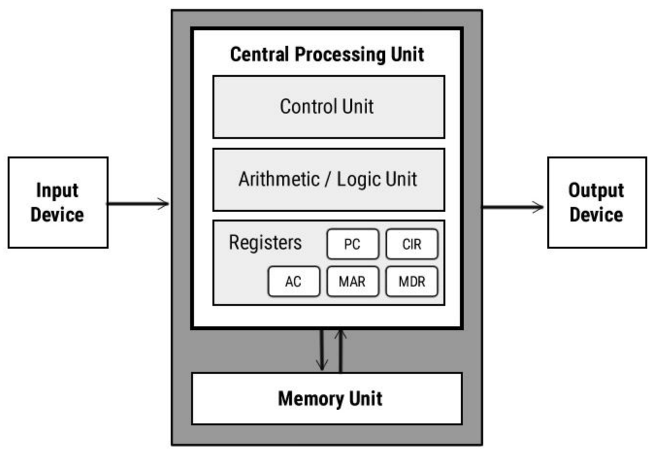

Figure 1.1: Von Neumann Architecture

There has been a significant amount of research focused on demonstrating the potential use of neuromorphic systems for various applications, including spike-based learning, which is a type of learning that is inspired by the way neurons communicate in the brain. But still they are limited to developers and research groups. The only available at the market for all users is Akida1000 which does not support all network models such as long short-term memory networks (LSTM)[14], used in the field of Deep Learning. The system has limited building blocks that are only compatible for the Akida1000. Akida1000 uses an approach of spikes, or discrete events, to communicate information between neurons, rather than continuous signals. inputs that are 32-floating has to be converted into a uint-8 format.

One of the major challenges in developing a new hardware architecture is to overcome the communication and energy consumption problems associated with the Von Neumann architecture. In the human brain, both the memory and processing functions are co-located, allowing for simultaneous data processing and storage. To address these limitations, researchers have explored the development of hardware architectures that integrate memory and processing capabilities in a single unit. Onether promising solution is the development of hardware architectures that are designed to process and store data in a manner that is similar to the human brain. This approach, known as neuromorphic computing, has the potential to significantly improve the energy efficiency of computing systems and increase the processing speed of these systems[39]. AKD1000 neuromorphic processor is one such example of a neuromorphic processor available at the market for all users. It have been developed to simulate the behavior of biological neurons and synapses. Akida1000 has a unique architecture that enables high-speed, low-power operation for AI and machine learning applications. The chip can perform computations in an event-driven manner, which allows it to be highly energy-efficient compared to traditional Von Neumann architectures. Additionally, the chip includes on-chip learning, which enables it to adapt to changing inputs in real time.It has the capability to learn from one shot.

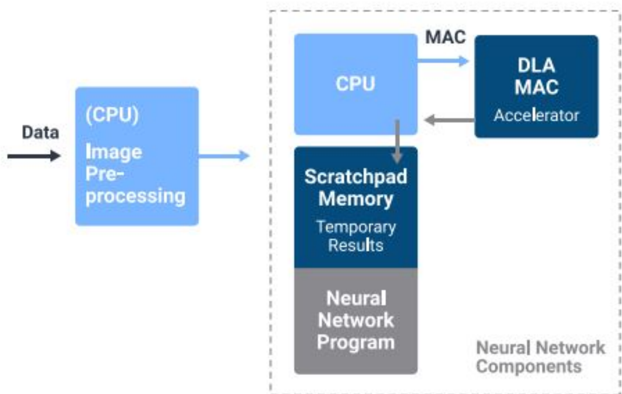

Figure 1.2: Akida Architecture

Akida1000 has been used to train deep neural networks to perform tasks such as text, audio, and image classification without performance trade-of. The development of these hardware architectures has the potential to significantly improve the energy efficiency and processing speed of computing systems, making them more suitable for a wide range of applications. Akida1000 is one example of a neuromorphic research chip that has been developed to demonstrate the potential of this new computing paradigm.

1.0.1 Aim and objectives of this thesis

The aim of this research project is to further explore the capabilities of SNNs and to gain a deeper understanding of their potential for use in real-world applications. To do this, the project will focus on four key areas: studying a new method for GABA detection, acquiring a suitable system for neuromorphic computing, conducting a literature review to explore what has been done in this field thus far, and developing a system for interfacing the neuromorphic chip with the measuring system.

One of the key motivations for this research is the growing demand for energy efficient and real time data processing solutions. With the exponential growth of data being generated and processed, there is a pressing need for computing systems that can handle this data in a fast and efficient manner. SNNs have the potential to provide such a solution, as they operate in a way that is inherently more energy efficient than traditional neural networks.

Another important motivation for this research is the increasing importance of real time data processing in various industries. For example, in healthcare, there is a need for real time monitoring of patients and prompt responses to changes in their conditions.SNNs have the potential to meet these needs by processing data in real time, thereby providing faster and more accurate results at the edge rather than the cloud where security is an issue. The project aims to contribute to the body of knowledge in this field and to help pave the way for the widespread adoption of neuromorphic chips in real world applications.

The use of neuromorphic computing for the detection of GABA is an area of research that has yet to be fully explored. While there has been some research into the use of neuromorphic computing for other types of biosensing applications, such as odor and gas sensing, there is currently a gap in the literature when it comes to using this technology for GABA detection. This gap in the literature may be due to the complexity of GABA detection, which involves analyzing the electrical properties of tissues and molecules and general-purpose neuromorphic hardware. GABA,being an inhibitory neurotransmitter plays a crucial role in the regulation of brain activity. It is real timed from glutamate, an excitatory neurotransmitter, and is found in high concentrations in certain regions of the brain. The detection of GABA is an important area of research as it can help to improve our understanding of the role of GABA in brain function and transfer this into a hardware.

Neuromorphic computing is a type of computing that is inspired by the structure and function of biological neural networks. These networks are designed to process information in a way that is similar to the way that the brain processes information. Neuromorphic computing has shown promise for a variety of applications, including image and speech recognition, and it has the potential to revolutionize computing by enabling the development of brain-inspired systems. While there has been some research into the use of neuromorphic computing for other types of biosensing applications, there is a gap in the literature when it comes to using this technology for GABA detection. One potential reason for this gap is that detecting GABA is a complex process. However, there is still potential for the use of neuromorphic computing in GABA detection, particularly as the technology continues to advance..

One potential approach to using neuromorphic computing for GABA detection is to develop a biosensor that is designed to mimic the properties of GABA receptors in the brain. Such a biosensor could be designed to detect the presence of GABA by analyzing changes in the electrical properties of the receptor. The design of such a biosensor could be inspired by the structure and function of biological neural networks and could leverage the power of neuromorphic computing to process the signals involved in GABA detection. Another potential approach to using neuromorphic computing for GABA detection is to develop a machine learning algorithm that is trained on a large dataset of GABA-related signals. Such algorithm could be trained to recognize patterns in the signals that are indicative of the presence of GABA and its concentration. Once trained, the algorithm could be used to analyze the electrical signals involved in GABA detection and to provide accurate and reliable GABA detection.

As the technology continues to advance, it is likely that new opportunities will emerge for using this approach to improve our understanding of the role of GABA in brain function and to develop new therapies for GABA-related disorders. While there are currently some challenges involved in GABA detection using neuromorphic computing. One approach that has gained significant momentum in recent years for addressing these challenges is the study of Spiking Neural Networks (SNNs). SNNs are unique in their operation within the spike domain, allowing them to process data in a manner that differs from traditional artificial neural networks. The potential for energy efficiency and real time data processing make SNNs an attractive option for a wide range of applications, including the analysis of GABA data. Despite ongoing research and a significant amount yet to be understood about their capabilities and limitations, SNNs have already demonstrated their potential to be trained on the same datasets used to train traditional neural networks. This opens up exciting possibilities for their use in various industries, from healthcare and finance to defense and transportation.

1.0.2 The structure and Outlining of the thesis

The aim of this Thesis is to provide an insight into the potential use of neuromorphic computing in bioimpedance signals such as detecting the levels of GABA. Even though it is in early stage for the field,but it has shown a significant improvement in power reduction and fast computational on applications such as image classification, object detection, and olfactory classification.

- **Chapter 2** A broad contextual framework is established. This encompasses a theoretical overview of bioimpedance relevant to the research, as well as a discussion on the dielectric properties of molecules. The chapter also details the tools and instruments employed in the study, and provides a general introduction to GABA, a neurotransmitter of importance to the research. Furthermore, the chapter explores several advanced techniques in neural computing, such as machine learning, deep learning, spiking neural networks and Neuromorphic computing.
- **Chapter 3** A literature review is conducted to investigate the potential of neuromorphic computing in the context of bioimpedance signals. The objective of this review is to provide a comprehensive overview of the existing research on neuromorphic computing and its potential applications. Specifically, the focus is on the acquisition of a neuromorphic chip that can interface with the DAK-3.5 dielectric constant measurement system. Although several neuromorphic hardware devices have been produced in recent years, they are primarily confined to the research domain. Therefore, the review is conducted in a systematic and comprehensive manner taking into account both theoretical and practical aspects of neuromorphic computing.
- **Chapter 4** Presents the methodology adopted for the research. This includes a detailed description of the experiments carried out, which involved dielectric relaxation spectroscopy measurements on various GABA solutions. Additionally, the chapter describes the training of several machine learning methods such as Keras-tensorflow sequential API, including feedforward multi-layer neural networks, convolutional neural networks, and spiking neural networks. Lastly, the methodology section details the transfer learning process and the deployment of the trained model to the Akida1000 neuromorphic processor.Moreover, the development of a system for interfacing the neuromorphic chip with the measuring system is outlined in this chapter.The system is designed to enable the acquisition and processing of data from the neuromorphic chip, which is then used to train the machine learning methodology.
- **Chapter 5** Present our findings by analyzing the results obtained from various methods and algorithms that were deployed, each with a different number of features. The objective of this analysis is to achieve a robust and accurate result with minimum loss.
- **Chapter 6** Serves to summarize the work done throughout the thesis and to present potential avenues for future research. This chapter features a comprehensive discussion of all the results obtained, placing them in the broader context of the thesis. It also serves to provide a valuable contribution to the field of neuromorphic computing and its application to bioimpedance signals.

Chapter 2

Theory

2.1 GABA And It's Function

The human brain is an incredibly complex system, with millions of neurons communicating with each other to control our thoughts, movements, and emotions. To facilitate this communication, neurons release chemical compounds called neurotransmitters that bind to receptors on other neurons and regulate their activity. One of the most important neurotransmitters in the central nervous system is GABA (gamma-aminobutyric acid), which has an inhibitory effect on neurons and helps regulate brain activity. GABA blocks chemical messages and slows down the activities of the brain by reducing the stimulation of nerve cells in the CNS. The process of GABA, like other neurotransmitters and chemical solutions in the brain, occurs in solutions containing electrolytes.GABA is produced in the brain and acts by binding to specific receptors on neurons, causing a decrease in neuronal excitability and inhibition of neurotransmitter release. This helps regulate the brain's overall activity and plays a critical role in the processes of controlling muscle tone, reducing anxiety, and promoting sleep. In addition, GABA is involved in a wide range of other physiological processes, including regulation of hormone secretion, regulation of heart rate, and regulation of respiratory function[62].

2.1.1 GABA Neurotransmiter System

The GABA neurotransmitter system involves the release of GABA from a presynaptic neuron and its binding to specific receptors in the brain. These receptors are divided into three main categories: GABAA, GABAB, and GABAC. The GABAA receptor, when activated by GABA, opens an ion channel that allows chlorine ions to flow into the cell, which results in the hyperpolarization of the membrane potential and inhibits the neuron's activity by preventing the action potential threshold from being reached. Similarly, the activation of the GABAB receptor leads to the opening of a potassium channel and the outflow of potassium ions, also causing hyperpolarization and inhibition of the neuron's activity. This inhibitory signaling is regulated by the GABAergic system in the brain, which helps to modulate the activity of excitatory neurons through two forms of inhibition: phasic and tonic[15]. Phasic activation of the GABA receptors is closely related to the regulation of neuronal activity rhythm. It is referred to as the fast and short-lived inhibitory signals transmitted between neurons in the brain. The role of phasic inhibition in the brain is crucial in maintaining the proper functioning of individual neurons and network oscillations. These oscillations are believed to be connected to higher cognitive processes. The process of phasic inhibition starts with the release of GABA from the presynaptic terminal. The rapid spread of the high concentration of GABA from the presynaptic site into the surrounding neuropil affects postsynaptic neurons, leading to their inhibition. After GABA binds to its receptors, it is removed from the synaptic cleft through molecular diffusion and taken up by the presynaptic GABAergic neuron. The neurons in the brain by binding to extrasynaptic GABAA receptors and generating tonic inhibition. while tonic activation mainly occurs in glial cells and enables the regulation of neuronal excitability. Its inhibition acts as a modulator of the excitability of neuron subtypes, and it has been shown to differentially affect the excitability of neurons according to variations in chloride gradient[8].

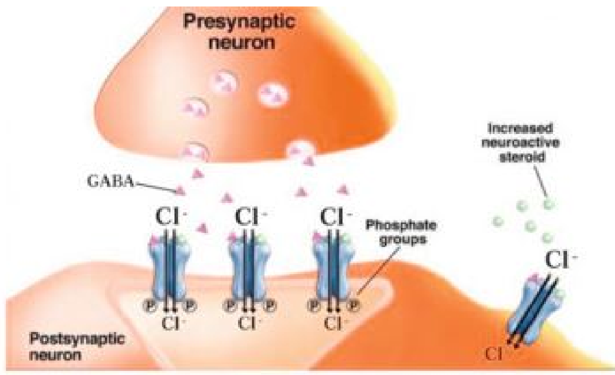

Figure 2.1: GABAergic neurotransmission

Recently, researchers have used biophysically detailed neuron models to investigate the effects of tonic inhibition on the excitability of different neuron subtypes. The results showed that tonic inhibition can modulate the excitability of neurons according to variations in electrophysiological properties. In particular, the results showed that tonic inhibition increased the responsiveness (or gain) in models with features typical for somatostatin interneurons but decreased the gain in models with features typical for parvalbumin interneurons. The mechanism underlying gain modulation results showed that gain modulation was dependent upon the magnitude of tonic current generated at depolarized membrane potential, a property associated with outward rectifying GABAA receptors. Furthermore, tonic inhibition produced two biophysical changes in the models that were relevant to neuronal excitability. Enhanced action potential repolarization through increased current flow into the dendritic compartment, and reduced activation of voltage-dependent potassium channels.
Reduced potassium channel activation selectively increased the gain in models with action potential dynamics typical for somatostatin interneurons. In parvalbumin-type models, potassium channels deactivate rapidly and are unavailable for further modulation. GABA can differentially modulate interneuron excitability through differences in intrinsic electrophysiological properties and provide a mechanism for this modulation.

GABA can exist in different conformations due to its flexible carbon backbone, and the pH of the local environment can determine its ionization state, which can be in the form of an acid, neutral, or base. The acid and neutral forms of GABA have been reported to have a folded and open topology, respectively. The folded topology involves the formation of an intramolecular hydrogen bond between the carboxylate and amino groups, while in the open topology, the two side groups are far apart from each other. The regulation of GABA in the brain is maintained by the interaction between neuron and glial cells [34]. However, there are multiple factors that can lead to dysregulation of GABA in the brain and result in neurodegenerative diseases. These factors include alterations in the synthesis or release of GABA. Abnormal levels of GABA have been implicated in several neurological and psychiatric disorders, including epilepsy, depression, and anxiety disorders[9]. For example, in epilepsy, seizures are caused by the abnormal activity of neurons in the brain, and increasing GABA levels have been shown to suppress this activity and reduce the frequency and severity of seizures. Similarly, depression and anxiety disorders are thought to be related to imbalances in the levels of neurotransmitters in the brain, and increasing GABA levels have been shown to be effective in treating these disorders in some patients[59].

One of the key properties of GABA is its ability to promote relaxation and calmness, which makes it a crucial neurotransmitter for controlling anxiety and promoting sleep. This is because GABA is involved in inhibiting the activity of neurons in the brain that produce excitatory neurotransmitters such as glutamate. When the activity of these neurons is suppressed, the overall activity of the brain is reduced, leading to feelings of calmness and relaxation[20].

There is still much to be learned about the role of GABA in the brain, but what is clear is that this neurotransmitter plays a critical role in regulating brain activity and is involved in many important physiological processes. By understanding more about GABA and its effects on the brain, we may be able to develop more effective treatments for a wide range of neurological and psychiatric disorders.In recent years, much research has been devoted to understanding the function of GABA in the brain and its involvement in the development of neurological and psychiatric disorders. For example, Some scientists have used brain imaging techniques such as magnetic resonance imaging (MRI) to examine the levels of GABA in the brains of individuals with epilepsy, depression, and anxiety disorders[64]. While others have used dielectric relaxation spectroscopy techniques to study the electrical properties of GABA and linked this property to measure the concentration of GABA[16]. GABA is a big dipole, meaning that it is affected by electrical fields. When an alternating electrical field is applied to a solution containing dissolved GABA molecules, the GABA molecules will rotate in order to attain a minimum energy level.The rotation of the GABA molecules will continue to increase as the frequency of the alternating electrical field increases until the GABA molecules can no longer follow the fast-alternating field. The effect of energy loss in GABA solutions depends on the dielectric constant values and the concentration of the GABA in the solution[29]. The dielectric constant is a measure of the electrical polarization of a material and is proportional to the concentration of the material in the solution. Therefore, solutions containing higher concentrations of GABA will show higher dielectric constant values. Additionally, these characteristics are important for understanding the behavior of GABA solutions in biomedical fields and can be used for further research to develop new methods for analyzing GABA levels and implement this on systems that can mimic the human brain.

2.1.2 bioimpedance measurement

Bioimpedance measurements are a widely used technique for detecting structural changes in cells. The method is versatile and can be adapted to detect various substances, including GABA. However, detecting the presence of GABA is challenging due to its non-electroactivity. This means that it will not react to an applied current, making it difficult to detect with general bioimpedance analysis methods such as the cyclic voltammetry technique which is commonly used to detect chemical changes in electroactive substances[42]. However, since GABA is non-electroactive, it is undetectable with the method of using cyclic voltammetry. Despite this, there are still ways of detecting the presence of GABA. The GABA molecule is considered a polar molecule, with its positively and negatively, charged sides well separated. This property makes it possible to detect GABA with dielectric relaxation spectroscopy. Dielectric relaxation spectroscopy is a technique that measures the dielectric response of a material to an alternating electric field[53]. The dielectric response is the change in the electrical polarization of the material in response to an applied electric field. The technique is based on the fact that polar molecules, such as GABA, can be detected through their dielectric properties. In dielectric relaxation spectroscopy, the dielectric response of a sample is measured over a range of frequencies. The spectrum obtained from the measurement contains information about the molecular structure and dynamics of the sample, including the presence of GABA.

Dielectric substances are classified as polar and non-polar dielectrics based on the response of their molecules to an external electric field[66]. In the absence of an external electric field, the molecules of a polar dielectric substance are arranged in an irregular manner due to thermal energy. This results in any volume of the substance that has a large number of molecules having a zero resultant dipole moment. As a result, the resultant dipole moment of the substance is zero. However, when a polar dielectric substance is placed in an external electric field, each molecule experiences a torque that tries to align the molecule along the direction of the external electric field. As the intensity of the external electric field is increased, more and more molecules align themselves in the direction of the field, resulting in a net dipole moment in the substance.

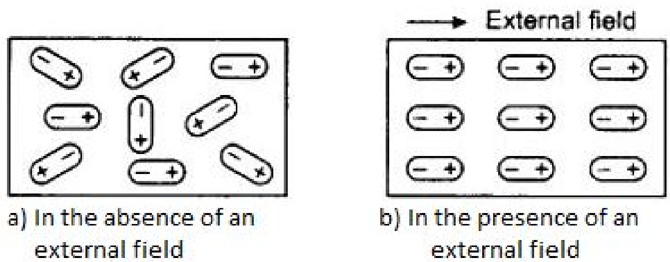

Figure 2.2: Polar Dielectric Molecule

In contrast, non-polar dielectrics are substances in which the molecules do not have a permanent dipole moment. As a result, they do not experience any torque in an external electric field and do not align themselves in the direction of the field. Instead, the non-polar molecules get distorted due to the presence of the external electric field, but do not produce a net dipole moment in the substance.

2.2 Dielectric relaxation spectroscopy

Dielectric relaxation spectroscopy is a type of impedance spectroscopy that can measure the dielectric properties of a material as a function of the frequency of an applied electric field. These properties are important in understanding the structure and dynamics of biological materials, and accurate knowledge of them is crucial for medical diagnostic, monitoring, and therapeutic technologies[35]. Dielectric relaxation spectroscopy can be used to study changes in the concentration of substances and the formation of chemical species in biological materials, as well as investigate the motion of molecules with an electric dipole moment. The technique measures the energy storage and dissipation properties of a material through its ability to pass an alternating current, making it sensitive to molecular mobility and structural changes. While dielectric relaxation spectroscopy has been around for some time, it has not been thoroughly explored for life science applications, despite its potential as a rapid non-invasive technique for structural characterization. Studying dielectric relaxation is one of several techniques used to obtain information about the physicochemical properties of biomolecules in aqueous solution, making it a useful tool for measuring GABA solutions.

2.2.1 DAK-3.5 Kit

The DAK System is a dielectric assessment system that provides an easy-to-use solution for assessing the dielectric properties of various media by combining the DAK technology with miniature R60 and R140 vector reflectometers from Copper Mountain Technologies. It is a powerful tool used in the field of biomedicine for the measurement of the electrical properties of biological tissues. This system combines the capabilities of a Dielectric Analysis Kit (DAK-3.5) and a Vector Network Analyzer (VNA) to measure the impedance of biological tissues at multiple frequencies. The DAK-3.5 is used to apply a small alternating electrical current to the tissue, while the VNA measures the resulting voltage and impedance. The DAK System has many potential applications in the field of dielectric assessment, and it can be used to measure the dielectric properties of various materials, including solids, liquids, and gases. Additionally, the system can be used to measure the dielectric properties of living tissues, such as skin and muscle, making it an ideal tool for medical research.

Moreover, bioimpedance theory is the basis for the measurement of the electrical properties of biological tissues. Bioimpedance spectroscopy (BIS) is a non-invasive technique used to measure the electrical properties of biological tissues, such as resistance and capacitance, which are related to their cellular structure and composition. The impedance of a tissue is related to its resistance and capacitance, and by analyzing the impedance at different frequencies, it is possible to determine the properties of different components within the tissue. Dielectric properties of molecules are related to their ability to store and release electrical energy when exposed to an electric field. The dielectric constant of a material, also known as its permittivity, is a measure of its ability to store electrical energy. It is a dimensionless quantity that is related to the molecular structure of the material, including the type and arrangement of atoms and molecules, as well as the presence of electrical charges. The dielectric properties of molecules are important in many applications.

Recently, bioimpedance theory and the measurement of dielectric properties have been applied to the study of GABA, using the DAK-3.5 integrated with VNA Vector Network Analyzer R140 to measure changes in the electrical properties of tissues in response to changes in GABA levels, providing a non-invasive tool for understanding the effects of GABA on brain activity that can be used for diagnosing and treatment of diseases related to GABA metabolisms, such as epilepsy, anxiety disorders, and depression. The DAK-3.5 integrated with VNA Vector Network Analyzer R140 has the potential to provide valuable insights into the role of GABA in brain function by measuring changes in the electrical properties of tissues in response to changes in GABA levels. In the study of GABA using the DAK-3.5 integrated with VNA Vector Network Analyzer R140, electrodes are used to apply the alternating electrical current to the tissue and to measure the resulting voltage. The VNA measures the impedance of the tissue at multiple frequencies to determine its electrical properties. The impedance spectra obtained using this system provide information on the resistance and capacitance of the tissue, which is related to its cellular structure and composition.

One advantage of the DAK-3.5 integrated with VNA Vector Network Analyzer R140 is that it allows for the measurement of impedance at a wide range of frequencies, from low to high[23]. This enables the determination of the properties of different components within the tissue, such as cells, water, and ions. Additionally, the VNA provides accurate and precise measurements of impedance, which can be used to detect changes in the electrical properties of tissues in response to changes in GABA levels.

Another advantage of this system is that it is non-invasive, making it suitable for repeated measurements over time. This can be particularly useful in the study of the effects of GABA on brain activity, as changes in GABA levels can be monitored over time and correlated with changes in the electrical properties of tissues. The non-invasive nature of this system also makes it safe for use in

Figure 2.3: DAKS 3.5 (200 MHz- 14 GHz)

human subjects, as there is no need for invasive procedures to obtain the data.

The data obtained using the DAK-3.5 integrated with VNA Vector Network Analyzer R140 can be analyzed using various mathematical techniques, such as circuit analysis and complex plane analysis, to extract information on the electrical properties of tissues. This information can then be used to gain insights into the role of GABA in brain function and to understand the underlying mechanisms by which GABA produces its effects.

2.3 Machine Learning

Machine learning is a subset of artificial intelligence (AI) that enables computers to learn and make predictions or decisions without being explicitly programmed. It involves feeding a computer system a large dataset of training examples and allowing the system to learn and make predictions based on this data. The goal of machine learning is to build models that can generalize well and make accurate predictions on new, unseen data[41]. There are several types of machine learning, including supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning.

Supervised learning is the most common type of machine learning and involves training a model on labeled data. In this type of learning, the model receives inputs and the corresponding outputs, and the goal is to learn a mapping from inputs to outputs. For example, in a classification problem, the inputs are features, and the outputs are class labels. The model is trained on a labeled dataset and then used to predict the class label of new, unseen examples.

Unsupervised learning, on the other hand, involves training a model on unlabeled data. The goal of unsupervised learning is to find patterns or structures in the data, such as clusters, without having prior knowledge of the output labels. This type of learning is used in tasks such as dimensionality reduction, anomaly detection, and association rule learning.

Semi-supervised learning is a combination of supervised and unsupervised learning. It involves training a model on a mix of labeled and unlabeled data. The goal is to leverage the large amount of unlabeled data to improve the performance of the model on the limited amount of labeled data.

Reinforcement learning is a type of machine learning where the goal is to maximize a reward signal by taking actions in an environment. In this type of learning, an agent interacts with an environment, takes actions, and receives rewards. The goal is to learn a policy that maximizes the cumulative reward received by the agent over time.

Machine learning algorithms can be divided into two categories: parametric and non-parametric. Parametric algorithms have a fixed number of parameters, while non-parametric algorithms do not. For example, linear regression is a parametric algorithm, while decision trees and random forests are non-parametric algorithms. The choice of machine learning algorithm depends on several factors, including the size and structure of the dataset, the complexity of the task, and the resources available for training and inference. Some popular machine learning algorithms include linear regression, logistic regression, decision trees, random forests, support vector machines (SVMs), k-nearest neighbors (k-NN), and neural networks. Regression and classification are two main types of problems in supervised learning.

2.3.1 Regression And Classification

Regression and classification are two main types of problems in supervised learning. Regression models are widely used in machine learning to predict a continuous output variable, such as a numerical value[11]. They are commonly used in many applications, such as sales forecasting, stock price prediction, and weather forecasting. The use of regression models has become increasingly popular in recent years due to advances in computational power and the availability of large datasets. The goal of regression is to find a mathematical relationship between the input and output variables that can be used to make accurate predictions on new, unseen data.There are several different types of regression, but the most common type is linear regression, which models the relationship between the input variables and the output variable using a linear equation. The equation takes the form:

$$y = b0 + b1x1 + b2x2 + ... + bn * xn \tag{2.1}$$

where y is the output variable, b0 is the intercept or bias term, b1 to bn are the coefficients or weights for the input variables x1 to xn, and x1 to xn are the input variables.The goal of linear regression is to estimate the values of the coefficients that minimize the difference between the predicted output and the actual output. This is done by minimizing a cost function, which measures the difference between the predicted and actual outputs for the training data.

The most common method for estimating the coefficients is Ordinary Least Squares (OLS) regression, which finds the values of the coefficients that minimize the sum of the squared errors between the predicted and actual outputs for the training data. This can be solved analytically using matrix algebra, or iteratively using optimization algorithms such as Gradient Descent or Stochastic Gradient Descent. Once the coefficients have been estimated, the model can be used to make predictions on new, unseen data. To make a prediction, the input variables are fed into the model, and the output variable is calculated using the estimated coefficients and the linear equation.

Classification, on the other hand, is used to predict a categorical output variable, such as a label or a class. In classification, the model learns to assign each input to one of several possible categories. The goal of classification is to learn a decision boundary that separates the different classes in the input space, so that new, unseen data can be classified correctly. There are several different types of classification algorithms, but the most common type is binary classification, which predicts one of two possible classes. Multi-class classification, which predicts one of more than two possible classes, is also commonly used. The input variables used in classification are known as features, and the output variable is known as the target or label. The training data is a set of labeled examples, where each example consists of a set of features and a corresponding target or label. The goal of classification is to learn a function that maps the features to the correct label.

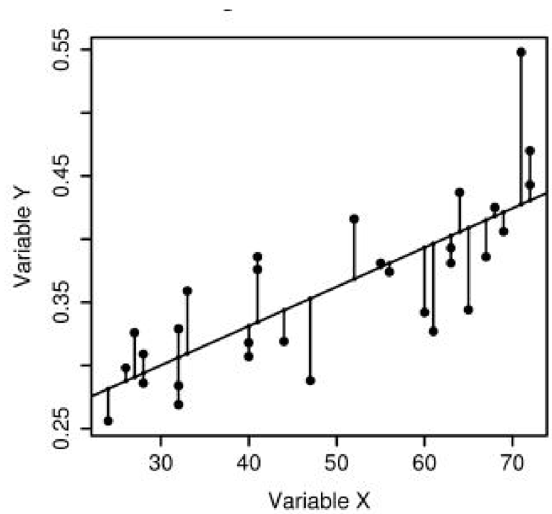

Figure 2.4: Ordirnary Least Squares

One common algorithm for binary classification is logistic regression, which models the probability of the positive class as a function of the input variables using a logistic function. The logistic function takes the form:

$$p(y = 1|x) = 1/(1 + exp(-z)) \tag{2.2}$$

where p(y=1—x) is the probability of the positive class given the input variables x, and z is a linear combination of the input variables and their corresponding weights or coefficients:

$$z = b0 + b1x1 + b2x2 + ... + bn * xn \tag{2.3}$$

where b0 is the intercept or bias term, and b1 to bn are the coefficients or weights for the input variables x1 to xn.

The goal of logistic regression is to estimate the values of the coefficients that maximize the likelihood of the training data. This is done by minimizing a cost function, which measures the difference between the predicted and actual probabilities for the training data. Once the coefficients have been estimated, the model can be used to make predictions on new, unseen data. To make a prediction, the input variables are fed into the model, and the probability of the positive class is calculated using the estimated coefficients and the logistic function. The predicted class is then determined based on a threshold, which is typically set to 0.5.

Classification is widely used in fields such as finance, healthcare, and natural language processing, and it has many applications such as fraud detection, spam filtering, and sentiment analysis. However, it is important to choose the appropriate type of classification algorithm and to carefully validate the model to ensure that it is accurate and robust.

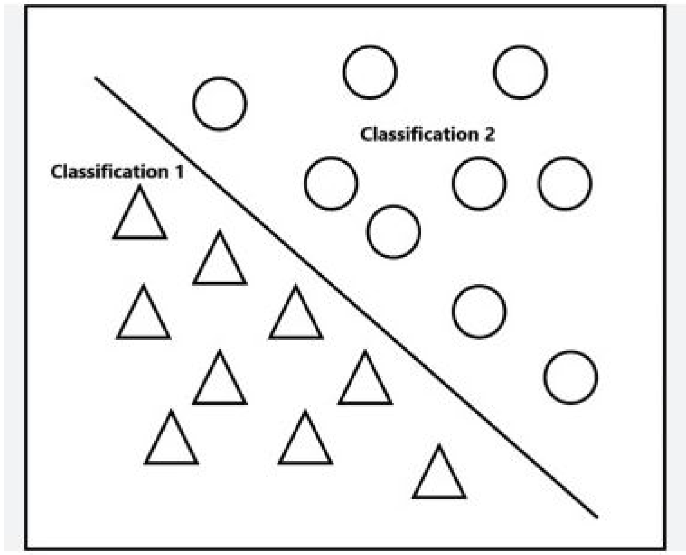

Figure 2.5: Classification

2.3.2 Neural Networks

Neural networks, inspired by the structure and function of the brain, are a type of machine learning algorithm[48]. They consist of interconnected layers of artificial neurons and can be utilized for various tasks such as image classification, speech recognition, and natural language processing. These networks are designed to recognize the underlying relationships in a given dataset by imitating the way the human brain functions. The systems of neurons can either be organic or artificial in nature, and they have the ability to adapt to changing input. As a result, neural networks can generate the best possible results without requiring the redesign of output criteria. An artificial neural network behaves the same way. It works on three layers. The input layer takes input. The hidden layer processes the input. Finally, the output layer sends the calculated output.

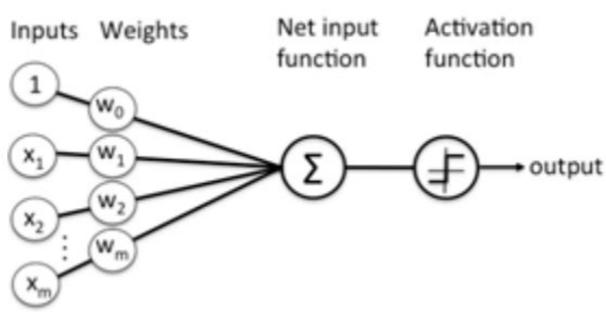

Figure 2.6: Neural Network Structure

Neural networks have been around for many years, and they have gone through several periods during which they have fallen in and out of favor. But recently, they have steadily gained ground over many other competing machine learning algorithms. They were created to overcome the precision problems faced by earlier classification algorithms known as perceptrons. The invention of the

Multi-layer perceptrons algorithm allowed for greater accuracy by incorporating layers of perceptrons. The input is propagated forward in the network, with the values being multiplied by weights and the bias being added. To increase the accuracy of the classifier, the neural network tunes these variables in each cycle through a process called training.

2.3.3 Deep learning

Deep learning is a subset of machine learning that uses artificial neural networks to learn from large amounts of data. These networks are made up of multiple layers of interconnected neurons that work together to identify patterns and relationships in the data. Deep learning has become increasingly popular in recent years due to its ability to solve complex problems in a wide range of domains. These methods have dramatically improved the state-of-the-art in speech recognition, visual object recognition, object detection, and many other domains such as drug discovery and genomics[37]. Deep learning discovers intricate structures in large data sets by using the backpropagation algorithm to indicate how a machine should change its internal parameters that are used to compute the representation in each layer from the representation in the previous layer. Some of the most impressive applications of deep learning include self-driving cars, facial recognition, and language translation.

One of the key advantages of deep learning is its ability to automatically learn features from raw data without the need for manual feature engineering. This means that the network can identify relevant features in the data on its own, without the need for human intervention. This is particularly useful in tasks where the relevant features are not well understood or where the data is high-dimensional.

Another important aspect of deep learning is its ability to handle large amounts of data. Deep learning algorithms are typically trained on large datasets consisting of millions of samples. This allows the network to learn patterns and relationships that may not be apparent in smaller datasets. There are several types of neural networks used in deep learning, including feedforward neural networks, convolutional neural networks, and recurrent neural networks. Feedforward neural networks are the simplest type of neural network and are used for tasks such as classification and regression. Convolutional neural networks are commonly used in computer vision tasks and are designed to identify features in images. Recurrent neural networks are used for tasks involving sequential data, such as speech recognition and natural language processing. The training process in deep learning involves feeding large amounts of data into the network and adjusting the weights of the connections between the neurons to minimize a loss function. The loss function measures the difference between the output of the network and the desired output. During training, the network adjusts its weights to minimize the loss function, allowing it to learn from the data.

Feedforward Neural Network

Feedforward Neural Networks, or FFNNs, are a type of artificial neural network that is widely used for various applications, including image and speech recognition, natural language processing, and control systems. FFNNs consist of interconnected nodes, known as artificial neurons, which process and transmit information through the network. The information in FFNNs flows in one direction, from the input layer to the output layer, through any hidden layers[57]. The input layer receives the input data, which is then processed by the hidden layers using weighted connections and activation functions. The output layer produces the network's prediction or output. To train the FFNN, the weights of the connections between the neurons are adjusted using a supervised learning algorithm, such as backpropagation. FFNNs have several advantages, including their versatility to approximate any continuous function and their ability to handle large amounts of data[27]. However, they also have some limitations, such as the vanishing gradient problem and the potential for overfitting. Despite these limitations, FFNNs are still a valuable tool in the field of artificial intelligence, with their ability to solve complex problems.

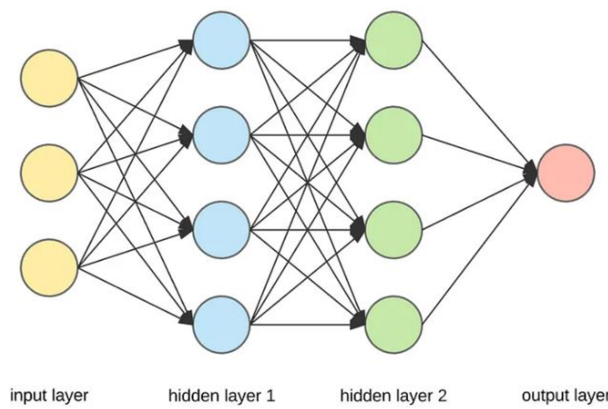

Figure 2.7: Feedforward Neural Network

One of the potential challenges, which is overfitting occurs when the network becomes too specialized to the training data and is unable to generalize to new data[50]. To address this issue, several techniques have been developed, including regularization, dropout, and early stopping. Regularization is a technique that penalizes large weights in the network, reducing the risk of overfitting. Dropout is a technique that randomly drops out some of the neurons in the network during training, preventing the network from becoming too specialized to any particular feature[27]. Depending on the architecture and application it may have an effect on the model. Early stopping is also another technique that stops the training process when the performance on a validation set stops improving, preventing the network from overfitting to the training data.

Despite its impressive performance in many domains, deep learning also has several limitations. One of the main limitations is the need for large amounts of labeled data and powerful hardware. Deep learning algorithms require large datasets to learn patterns and relationships in the data. In domains where labeled data is scarce, deep learning may not be the best approach. Another limitation of deep learning is its lack of interpretability. Deep learning algorithms are often described as "black boxes" because it can be difficult to understand how the network is making its predictions. This lack of interpretability can make it challenging to trust the predictions made by the network, particularly in high-stakes domains such as healthcare. As deep learning continues to advance, it is likely that these limitations will be addressed, enabling even more impressive applications in the future.

2.3.4 Convolutionl Neural Network

Convolutional Neural Networks (CNNs) are a type of artificial neural network that is commonly used for image and video recognition tasks[55]. Unlike traditional feedforward neural networks, CNNs are designed to take advantage of the structure and spatial relationships in image data. CNNs consist of multiple layers, including the input layer, hidden layers, and the output layer. The hidden layers are composed of convolutional layers, pooling layers, and activation layers. The convolutional layers apply filters to the input data, while the pooling layers reduce the spatial size of the data. The activation layers introduce non-linearity into the network.The training process in CNNs involves adjusting the weights of the filters in the convolutional layers to minimize the error between the network's prediction and the actual output. To ensure the network does not overfit to the training data, techniques such as dropout and data augmentation can be used. One of the key advantages of CNNs is their ability to learn local patterns in the data. This allows them to automatically extract features from the input data, reducing the need for manual feature engineering. Additionally, CNNs can be trained on large datasets, making them well-suited for image and video recognition tasks. Furthermore, CNNs have the capability to learn hierarchical representations of the data, where lower-level features are combined to form higher-level features. This allows CNNs to capture increasingly complex and abstract features in the data as the network goes deeper. Additionally, CNNs are able to share weights across different parts of the image, reducing the number of parameters in the network and improving the efficiency of the training process. Another advantage of CNNs is their translational invariance, meaning they can recognize the same features regardless of their position in the image. This makes CNNs robust to small translations and rotations in the input data and allows them to generalize well to new data.

In recent years, CNNs have achieved state-of-the-art results on a wide range of image and video recognition tasks, and have been applied to areas such as object detection, semantic segmentation, and generative models. The success of CNNs has led to the development of deeper and more complex architectures, such as Residual Networks and Inception Networks, which have further improved the performance of CNNs on these tasks.

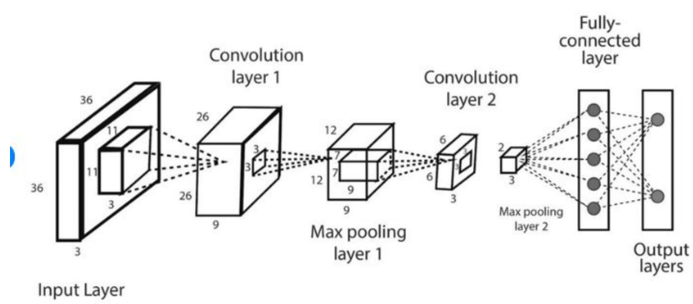

Figure 2.8: Convolutional Neural Network with Fully connected Layer

2.4 Spiking Neural Network

Numerous Machine Learning (ML) algorithms have been developed for stream learning. However, most off-the-shelf models require retraining in evolving environments and struggle to scale due to their learning algorithm. In recent years, Artificial Neural Networks (ANNs), inspired by the biological process by which the brain acquires and processes sensory information, have been used to tackle fast-evolving information flows. The Spiking Neural Network is a biologically plausible neuron model that is popular for capturing informational dynamics among real biological neurons, allowing for more accurate and powerful computational and integrating various information dimensions into a single model and dealing with large volumes of data[25]. SNNs are considered the third generation of ANNs and have the ability to learn continuously and incrementally, making them adaptable to non-stationary and evolving environments and useful as drift detectors. SNNs have also demonstrated their ability to capture temporal associations between variables in streaming data. They are the new development in Artificial Intelligence (AI) that utilizes machine learning methods to train models in the spike domain[26].

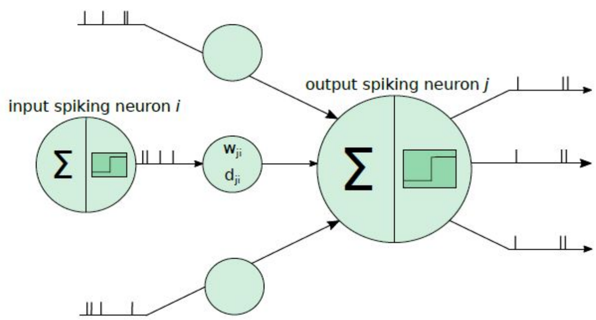

Figure 2.9: Spiking Neuron

SNNs learn from data that has been trained on, much like traditional Artificial Neural Networks. Despite being a relatively new field of study, there has been a significant amount of research into understanding how biological neurons learn, and how that knowledge can be applied to train SNNs to perform various tasks.

The process of learning in biological neurons occurs primarily through the strengthening and weakening of synapses. In the simplest terms, when an incoming spike causes an output spike, the larger the synaptic weight between the neurons, the stronger the connection. This strengthening or weakening of synapses is observed experimentally and is a key aspect of biological learning. In addition to this, experimental data suggests that the addition or removal of synapses can also play a role in learning.

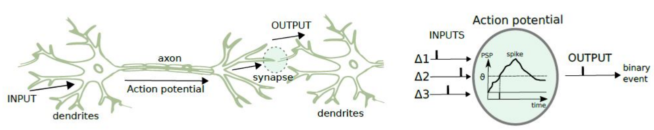

Figure 2.10: Biological neuron and its association with an artificial spiking neuron

One model for biological learning that has received a lot of attention is Spike-Timing-Dependent Plasticity (STDP)[17]. This model is based on the idea that if a presynaptic neuron fires just before a postsynaptic neuron, the connection between the two will be strengthened. Conversely, if the postsynaptic neuron fires before the presynaptic neuron, the connection will be weakened. While STDP has shown promise in early research, it has been found that building a large, complex functional system using this model is much more difficult than initially thought.

Despite the challenges, researchers have continued to make progress in the field of SNNs. One key development has been the use of supervised learning algorithms to train SNNs, such as the backpropagation algorithm. This allows for the weights of the synapses in an SNN to be adjusted in a way that minimizes the error between the predicted output and the actual output. One of the benefits of SNNs is their ability to model the asynchronous and parallel nature of information processing in the brain. This makes them a promising solution for real-time applications, such as image and speech recognition, where traditional ANNs can struggle. In addition, SNNs can be implemented in hardware much more efficiently than traditional ANNs, making them well-suited for applications where power consumption is a concern.

Data and information represent as Spikes and before feeding input data to a Spiking Neural Network, it must first be encoded into spike trains, which are spatio-temporal patterns of spikes that represent the input stimuli. This encoding process remains an open issue in neuroscience, as questions persist about what information is contained in these spiking patterns and what code neurons use to transmit that information. However, research has traditionally shown that most of the relevant information is contained in the mean firing rate of neurons[25].

Two main encoding schemes exist for encoding input data into spike trains: temporal encoding and rate-based encoding[7]. These encoding schemes are also referred to as temporal coding and rate coding, respectively. Temporal encoding is used when patterns within the encoding window provide information about the stimulus that cannot be obtained from spike count. This encoding scheme is based on spike timing and includes methods such as time-to-first-spike, where a code for the timing of the first spike contains all information about the new stimulus; phase, which applies a time-to-first-spike encoding scheme when the reference signal is not a single event, but a periodic signal; and correlations and synchrony, which uses spikes from other neurons as the reference signal for a spike code [25]. Rate-based encoding, on the other hand, is based on a spiking characteristic within a time interval (e.g., frequency) and includes three different notions of mean firing rate: rate as a spike count, rate as a spike density, and rate as a population activity [7]. The rate-based encoding scheme is used when the information is encoded in the mean firing rate of the neuron, and not in the timing of individual spikes.

The choice of encoding scheme depends on the specific characteristics of the input data and the task at hand and can impact the performance of the SNN.

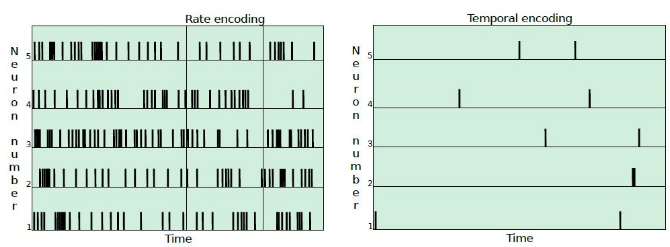

Figure 2.11: Rate-based encoding versus Temporal encoding

Despite the exciting developments in the field of SNNs, researchers still face significant challenges in effectively training them, such as incorporating biologically inspired learning rules like STDP into the training process and dealing with the high-dimensional, time-varying nature of spike data. Nonetheless, SNNs are highly regarded in the online learning research community due to their adaptability and accurate representation of brain-like information processing, making them suitable for high-performance hardware platforms. While the field of SNNs is still in its early stages, there is a great deal of promise for these networks to be used in a wide range of applications, and researchers will continue to build on knowledge gained from biological and machine learning studies to develop more effective training methods for SNNs, making them more widely applicable.

The field of machine learning and deep learning has seen explosive growth in the last decade, achieving breakthrough results in various domains such as image recognition, natural language processing, autonomous vehicles, financial prediction, and medical diagnosis. However, the traditional computing architecture used for implementing these algorithms has limitations, particularly for certain applications like wearable devices and bioimpedance-based sensory systems. Separating memory and processing units in traditional computing architectures leads to significant energy consumption and communication delays between the memory and processor, which is a critical issue for these applications. In contrast, neuromorphic computing is a new approach to computing that aims to create more energy-efficient architectures by emulating the way the brain processes information. This approach can potentially address the limitations of traditional computing architectures and enable the development of more energy-efficient systems that can operate in real-time, making them well-suited for applications with strict power constraints, such as IoT and wearable devices.

2.5 Neuromorphic Computing

The field of machine learning and deep learning has seen an explosive growth in the last decade, solving complex problems and achieving breakthrough results in various domains. From image recognition, natural language processing, and autonomous vehicles to financial prediction and medical diagnosis, the impact of machine learning algorithms has been profound. However, the traditional computing architecture used for implementing these algorithms has its limitations, which become more pronounced for certain applications, such as wearable devices and bioimpedance-based sensory systems. In particular, the separation of memory and processing units in traditional computing architectures leads to significant energy consumption and communication delays between the memory and processor, which is a critical issue for these applications.

Neuromorphic computing is the new approach to computing that aims to create more energy-efficient and faster computers by mimicking the way the human brain processes and stores information. It's a departure from the traditional Von Neumann architecture that has been the cornerstone of modern computing for decades. In the Von Neumann architecture, memory and processing are separate entities, leading to inefficiencies in communication and energy consumption[51].

In the human brain, the processing and storage of information are integrated into a single entity, allowing for fast and energy-efficient processing. Similarly, Neuromorphic computing tries to achieve this integration by using novel hardware and software systems that aim to replicate the functions of neurons and synapses in the human brain[10]. One key difference between traditional computing and Neuromorphic computing is the way information is stored and processed. In traditional computing, data is stored in memory, and the processor accesses the data to perform computations. This constant back and forth communication between memory and processor consumes a significant amount of energy and leads to delays in processing. In contrast, Neuromorphic computing systems store and process data in a single physical location, eliminating the need for constant communication between memory and processor. This results in a more energy-efficient computing system that can perform computations faster compared to traditional computing methods. Another advantage of Neuromorphic computing is its ability to perform well under low-power conditions, making it ideal for applications where power consumption is a concern. Neuromorphic systems can use algorithms that are more energy-efficient than traditional algorithms, as they can process and store information in parallel. This ability to conserve energy makes neuromorphic systems ideal for use in portable and embedded devices, such as mobile phones, laptops, wearable devices, and IoT (Internet of Things) devices where power consumption and size are critical factors[22].

In Neuromorphic computing systems, computation is performed using artificial neurons and synapses, which are modeled after their biological counterparts. These artificial neurons and synapses are integrated into the same physical location and are designed to perform the same functions as biological neurons and synapses. Neuromorphic computing systems can be implemented using both digital and analog circuits. Digital Neuromorphic computing systems use digital circuits to implement the artificial neurons and synapses, while analog Neuromorphic computing systems use analog circuits. Analog Neuromorphic computing systems are known to consume less power compared to digital Neuromorphic computing systems, making them ideal for low-power applications.

One of the challenges in Neuromorphic computing is creating systems that are scalable and can perform complex computations. Currently, Neuromorphic computing systems are limited in their computational capability, and research is being conducted to develop more advanced Neuromorphic computing systems that can perform complex computations.

Chapter 3

Neuromorphic Hardware and Software/Frameworks

3.1 Neuromorphic Hardware

In recent years, there has been a growing interest in mapping machine learning algorithms and deep learning workloads onto neuromorphic hardware. The objective of this mapping is to leverage the inherent energy efficiency and low power consumption of neuromorphic hardware, while still being able to perform complex computations. The use of machine learning and deep learning algorithms in various domains, such as image and speech recognition, autonomous driving, and natural language processing, has increased dramatically over the last decade. However, the energy consumption and computational demands of these algorithms are very high, making it difficult to run them on conventional hardware systems. Neuromorphic hardware provides a promising solution to this problem, as it is specifically designed to perform computations in an energy-efficient manner, mimicking the way the human brain processes information.

There are several approaches proposed for mapping machine learning workloads to neuromorphic hardware, each with different objectives and trade-offs. Some of the popular mapping approaches include Corelet[3], which is used to map Spiking Neural Networks (SNNs) onto TrueNorth hardware[21]. Another approach, PACMAN [24], maps SNNs onto SpiNNaker. PyNN [4] maps SNNs onto different hardware platforms, such as Loihi, BrainScaleS , and Neurogrid[6], by balancing the load on each tile. The primary objective of these approaches is to balance the workload on each tile by distributing the neurons and synapses evenly. This ensures that the computational demands of the machine learning algorithms are met, while also minimizing energy consumption. Beyond load balancing, recent techniques have also explored other objectives. For example, PSOPART is used to map SNNs to neuromorphic hardware, with the goal of reducing energy consumption on the shared interconnect[18]. SpiNeMap performs energy-aware clustering of SNNs and then maps the clusters to tiles, reducing the coication energy[54]. DecomposeSNN decomposes an SNN to improve the cluster utilization[5]. By leveraging the energy efficiency of neuromorphic hardware, it is possible to perform complex computations while minimizing energy consumption and preserving the computational efficiency of machine learning algorithms. These purposed approaches to simulating large-scale spiking neural networks have their own strengths and weaknesses. By considering the technology used to model neurons and synapses, the communication topology, and the support for synaptic plasticity, it is possible to choose the best approach for a particular application.

1. Technology used to model neurons and synapses:
   The technology used to model neurons and synapses can be either digital or analog. Digital

models are implemented on conventional general-purpose computers, including cluster machines and high-performance computers, or on special-purpose hardware such as FPGAs[44], graphics processor units, or custom silicon. Analog models, on the other hand, can be either subthreshold, where real-time performance is achievable, or above threshold, where the circuits are much faster than biological real-time.

Digital models are implemented on conventional general purpose computers, including cluster machines and high-performance computers, or on special-purpose hardware such as FPGAs, graphics processor units, or custom silicon. They have the advantage of flexibility and Their circuits can be easily reconfigured to handle a wide range of neural network models and tasks, making them more versatile than analog circuits. Digital neuromorphic hardware is also capable of performing complex computations with high accuracy, which makes it a good fit for tasks such as natural language processing, image recognition, and autonomous driving.

On the other hand, Analog neuromorphic hardware uses continuous signals, such as voltage or current, to represent and process data, while digital neuromorphic hardware uses binary digits or bits. Both analog and digital neuromorphic hardware have their own strengths and weaknesses, and the choice of which to use depends on the specific application and task. Analog circuits are often more power-efficient and can perform certain types of computations more quickly.This makes analog neuromorphic hardware especially well-suited for applications that require low-power consumption, such as implantable medical devices or sensors for the Internet of Things (IoT), while digital circuits are more flexible and can handle a wider variety of neural network models. Analog circuits are especially well-suited for processing continuous signals in real time, which is important for applications such as signal processing and control systems. Digital circuits, on the other hand, are better at processing discrete signals and can perform a wide range of computations with high precision and flexibility.

Overall, both analog and digital neuromorphic hardware are important in the development of intelligent systems, and advances in this field have the potential to revolutionize computing and artificial intelligence.

2. Communications topology:
   The communications topology employed by a computing system can have a significant impact on its ability to simulate large-scale neural networks. Some designs use conventional topologies, such as those used in cluster machines and high-performance computers, while others use novel topologies, such as the lightweight multicast packet routing mechanism used in SpiNNaker.

3. Support for synaptic plasticity:
   The support for synaptic plasticity is another important factor to consider when evaluating different designs for simulating neural networks. Some designs provide limited support for synaptic plasticity, while others provide full support, allowing for the simulation of plasticity and learning in neural networks.

3.2 Neuromorphic Cognitive Systems

Neuromorphic cognition systems refers to the study and development of computing systems that mimic the structure, function, and behavior of the biological nervous system[65]. This field aims to create novel hardware solutions that can perform complex computation real time on the principles of biological neural networks. Neuromorphic computing systems, such as the Akida 1000 neuromorphic processor, are designed to process and interpret sensory information in real time, making them suitable for various applications in the fields of robotics, artificial intelligence, and bioinformatics. One of the main advantages of neuromorphic computing is its ability to handle event-based data and perform computation with low latency and high energy efficiency, making it a promising platform for implementing machine learning algorithms and neural networks in bioimpedance-based sensory systems.

In recent years, it has been the development of single-chip devices that emulate peripheral sensory transduction, such as silicon retinas, visual motion sensors, and silicon cochleas. These single-chip devices have been successful in a wide range of applications, including robotics, medical devices, and sensory processing. For example, silicon retinas have been used in robotics to create vision systems that can detect and track objects in real time, while silicon cochleas have been used in hearing aids to improve speech recognition and sound localization. The success of these single-chip devices has led to the development of larger multi-chip neuromorphic systems that aim to emulate more complex neural networks. These multi-chip systems typically consist of one or more neuromorphic sensors, which are interfaced with general-purpose neural network chips using spiking silicon neurons and dynamic synapses. One example of such a system is the Neuromorphic Adaptive Plastic Scalable Electronics (NAPSE) chip[56], which was developed by a team at Georgia Tech. The NAPSE chip uses a scalable architecture to emulate the behavior of large scale neural networks and has been used for tasks such as sensory processing, control, and decision making. Some other examples of a multi-chip neuromorphic system are the SpiNNaker (Spiking Neural Network Architecture) project,IBM TrueNorth,Loihi,SynSense and Akida Neuromorphic processor.

3.2.1 SpiNNaker

SpiNNaker is a chip designed for high-performance and scalable neural network simulations. It is an architecture designed to be highly efficient and scalable, making it an ideal choice for large-scale neural network simulations. The chip was developed by the University of Manchester and it is still not commercially available, but limited to research groups. One of these groups is the computational neuroscientists group that uses SpiNNaker to simulate large neural models to understand the brain.The lightweight nature of the packet-routing mechanism, combined with its support for high connectivity, makes it possible for the SpiNNaker system to simulate large scale neural networks with millions of neurons and billions of connections. SpiNNaker's architecture is based on a scalable, parallel processing model that allows it to simulate the operation of large numbers of neurons and synapses in real time. The chip is designed to be highly energy-efficient, which makes it ideal for running large scale neural network simulations on embedded systems, such as robots and autonomous vehicles. Its architecture is based on a mesh-based communication system, which allows it to distribute data between processing nodes with low latency. This makes it ideal for large-scale simulations, as the communication overhead is low and the simulation can be scaled up by adding more nodes.
SpiNNaker has been used in various studies to investigate the functioning of the human brain and to develop new algorithms that mimic biological processes. For instance, researchers have utilized SpiNNaker to simulate the brain's visual system and to understand how it processes visual information. Such studies have led to the development of novel algorithms for image recognition that are more efficient and accurate than traditional approaches.
One of the key applications of SpiNNaker is in the field of machine learning, where it is used to implement large scale neural network algorithms in order to understand how the brain works and implement them in computer architecture. These algorithms are used in a variety of applications, including image recognition, speech recognition, and natural language processing. For example, SpiNNaker has been used in a number of studies to develop deep learning algorithms for image recognition, including the development of Convolutional neural networks (CNNs) for recognizing objects in images.

Another application of SpiNNaker is in the field of robotics, where it is used to implement real time control systems. For example, SpiNNaker has been used to develop real time control systems for robots that are capable of navigating complex environments. These systems use SpiNNaker to process sensory data in real time, allowing the robot to react quickly to changes in its environment. SpiNNaker has also been used in a number of studies to develop algorithms for brain-machine interfaces (BMIs)[36]. BMIs have been extensively researched and used for several years to detect imagined motor tasks by exploiting the natural link between these tasks and physical actions. However, a key challenge in realizing the future potential of neuromorphic computing is identifying and implementing brain-inspired algorithms to decode recorded signals in real time. To address this challenge, researchers have proposed and implemented a novel approach inspired by the olfactory system of insects to decode and predict imaginary movements from electroencephalogram (EEG) signals using a spiking neural network implemented on the SpiNNaker neuromorphic hardware (4 chip, 64 cores)[58]. The study provides a proof of concept for the successful implementation of a functional spiking neural network for decoding two motor imagery (MI) movements on the SpiNNaker system, which can be extended to classify more complex MI movements on larger SpiNNaker systems. There is a growing body of research that demonstrates the effectiveness of SpiNNaker for these applications, making it an important tool for those working in these fields.

The neural simulation on SpiNNaker follows an event-driven model, in which all computational tasks are triggered by events in hardware. The neuron states are computed in discrete timesteps initiated by a local periodic timer event on each processor. During each timestep, the processors evaluate the membrane potentials of all their neurons based on prior synaptic inputs and generate a packet for each neuron that spikes. These spike packets are routed to all processors that model neurons that are efferent to the spiking neuron[52]. When a processor receives a spike packet, it raises a packet event that prompts the efferent processor to retrieve the appropriate synaptic weights from off-chip RAM using a background Direct Memory Access (DMA) transfer. The processor can then perform other computations during the DMA transfer, and is notified of its completion by a DMA done event. The completion of the DMA transfer prompts the calculation of the sizes of synaptic inputs to subsequent membrane potential evaluations. This event-driven approach ensures that the processors only perform computations in response to relevant events, maximizing energy efficiency and computational throughput.

3.2.2 IBM TrueNorth

IBM TrueNorth is a neurosynaptic computing architecture designed to emulate the human brain's function and performance. The architecture was developed by IBM Research and was first introduced in 2014. The main aim of the TrueNorth architecture is to create a new type of computing system that can efficiently handle cognitive and perception tasks while consuming low power. It is based on the concept of neuromorphic computing, which is a new approach to computing that mimics the structure and function of the brain's neurons and synapses. The TrueNorth architecture is designed to be highly scalable, flexible, and programmable. It is based on a novel chip architecture that integrates thousands of tiny processors, each with its own memory, and a large number of programmable connections. This architecture allows the chip to perform multiple tasks simultaneously and to reconfigure itself on-the-fly, which makes it suitable for handling complex and dynamic environments.

One of the unique features of TrueNorth architecture is its highly parallel and low-power design. The architecture uses a low-precision, event-driven computing model that is optimized for the spiking neural network algorithms used in many cognitive computing tasks. This approach reduces the power consumption of the chip compared to traditional processors, making it suitable for use in battery-powered devices[3]. Another key aspect of the TrueNorth architecture is its programmability. The architecture uses a high-level programming language called Corelet, which makes it easier for developers to write and debug applications for the chip. Corelet is designed to be flexible and easy to use, allowing developers to easily program the chip to perform a wide range of tasks. The TrueNorth architecture has been used in a variety of fields, including computer vision, robotics, and speech recognition. For example, the architecture has been used to develop a real time image recognition system that can recognize objects and faces in images and videos[13]. The system has been tested on a variety of datasets and has demonstrated high accuracy and low latency.

The IBM Neurosynaptic system features the use of Convolutional Neural Networks (CNNs) in deep learning applications. These algorithms have been effective in solving various visual and audio recognition problems. As the development of new machine learning algorithms accelerates, the IBM TrueNorth chip has been shown to be capable of performing Deep Neural Networks (DNNs) efficiently, achieving near state-of-the-art results on eight standard datasets, including image and speech classification. For instance, the CIFAR100 dataset achieved 65.48% accuracy with the use of 31492 neurosynaptic cores, or about 8 TrueNorth chips[13]. As deep neural networks become increasingly deeper, their accuracy also rises, as evidenced by the use of 152-layer residual networks achieving high accuracy on the ImageNet test[28]. This level of accuracy requires the use of 8-bit neuron states, and while TrueNorth is efficient at accelerating machine learning, it still faces challenges such as high core occupancy and a gap between its performance and state-of-the-art accuracy. One issue with TrueNorth is the low resolution of the data representation, as all communications between the neurosynaptic cores use binary spikes and the synaptic weights are stored as integers and selected through a lookup table. This low resolution leads to quantization loss and a decrease in inference accuracy when mapping a trained neural network in floating-point data format to TrueNorth. To overcome this issue, stochastic computing methods are used, where data is statistically represented in both temporal and spatial domains, by using multiple spikes or multiple hardware copies of the neural networks[2]. With its ability to handle complex and dynamic environments, the TrueNorth architecture has the potential to revolutionize the field of computing and open up new possibilities for research and development in the future. Despite these promising capabilities, there are limitations to the flexibility of TrueNorth for users. Each type of layer is well-written, but they are encrypted in a protected function file, preventing users from accessing or modifying details within the convolutional layer, such as regularization or activation functions.

3.2.3 Loihi

Loihi is a groundbreaking research chip developed by Intel that has the potential to revolutionize the field of artificial intelligence and machine learning. This cutting-edge technology is designed to simulate the behavior of biological neurons and synapses, and has a unique architecture that enables high-speed, low-power operation for AI and machine learning applications. The chip represents a major advance in the field of neuromorphic computing, which aims to develop computing systems that mimic the structure and function of biological neurons and synapses[19]. The architecture of Loihi is based on a spiking neural network, which is a type of artificial neural network that operates using spikes or pulse-like signals. Unlike traditional Von Neumann architectures, which are based on a linear flow of information processing, Loihi can perform computations in an event-driven manner. This means that it can respond to incoming signals in real-time and perform computations only when necessary, which allows it to be highly energy efficient.

One of the key features of Loihi is its ability to perform on-chip learning. This is a type of machine learning that takes place directly on the chip, rather than on a separate computer. This means that the chip can adapt to changing inputs in real-time and continue to improve its performance over time. This ability to learn is particularly useful for AI systems that need to operate in resource-constrained environments, such as IoT devices, autonomous robots, and wearable devices. In addition to its event-driven architecture and on-chip learning capabilities, Loihi has several other unique features that make it well-suited for AI and machine learning applications. For example, it has a hierarchical structure that enables it to process information at different levels of abstraction. This allows it to perform complex computations more efficiently than traditional Von Neumann architectures. Additionally, Loihi includes a flexible interconnect structure that enables it to be easily reconfigured for different applications. The chip's ability to perform computations in an event-driven manner, combined with its ability to learn on-chip, makes it an ideal technology for building AI systems that can operate in real-world scenarios, where low-power consumption and high computational efficiency are critical.

The Loihi 2 second-generation neuromorphic research chip is compatible with the Lava open-source software framework designed for developing applications for neuromorphic hardware architectures. The software currently operates on CPUs and Loihi chips, but the compiler and runtime are open to extensions for other architectures. Access to Loihi 2 will primarily be through the Neuromorphic Research Cloud, which provides shared systems such as the "Oheo Gulch" single-chip system connected to an Aria 10 FPGA for early evaluation[38]. It will soon be joined by the "Kapoho Point," a compact (4x4-inch) stackable 8-chip system with Ethernet. Intel has tested Loihi 1 and 2 chips in various applications, including adaptive robot arm control, visual-tactile sensory perception, odor and gesture recognition, drone motor control with low latency response to visual input, fast database similarity search, modeling diffusion processes for scientific computing applications, and solving optimization problems such as railway scheduling. Moreover, the Loihi chip consumes significantly less power than standard CPU and GPU solutions, making it a promising candidate for neuromorphic AI acceleration. This feature may enable the Loihi chip to provide datacenter-like hardware capabilities for robots, autonomous vehicles, and other applications, with lower power consumption and latency. SNN's efficiency in battery-powered sensors with built-in AI is also expected to benefit from lower-end neuromorphic chips. The Loihi chip has been evaluated in various demonstrations, and in most cases, its power consumption was less than 1 watt. In comparison, standard CPU and GPU solutions typically consume tens to hundreds of watts, which highlights the breakthrough in energy efficiency that Loihi represents. In many of these demonstrations, relative gains reached several orders of magnitude, indicating the vast improvements that Loihi provides in energy efficiency. Furthermore, the Loihi chip exhibits state-of-the-art response times to incoming data samples while also adapting and learning from incoming data streams, making it ideal for the best applications. The combination of low power and low latency, with continuous adaptation, has the potential to introduce new intelligent functionality to power and latency constrained systems at a scale and versatility beyond what any other programmable architecture can currently support.

The Loihi 2 neuromorphic chip represents a significant advancement over the first generation Loihi with several notable improvements. Specifically, the Loihi 2 chip boasts up to 10 times faster processing capability, which includes a 2 times improvement for simple neuron state, 5 times for synaptic operations, and 10 times for spike generation. Additionally, the Loihi 2 chip provides up to 60 times more inter-chip bandwidth, which has been achieved through a combination of higher inter-chip signaling speed (four times faster), more inter-chip links (6 versus 4), and over 10 times reduction in inter-chip bandwidth utilization. Moreover, the Loihi 2 chip can support up to 1 million neurons, which represents a 15 times increase in resource density. The chip is also scalable in three dimensions with native Ethernet support, and fully programmable neuron models with graded spikes. Finally, the Loihi 2 chip boasts enhanced learning and adaptation capabilities. These impressive advancements demonstrate the significant progress made in the field of neuromorphic computing and hold promise for future developments in this area.

3.2.4 SynSense

SynSense is a neuromorphic computing platform that utilizes a novel architecture inspired by the biological structure of the human brain. Neuromorphic computing is a subfield of artificial intelligence that aims to design computing systems that work like the human brain. SynSense is unique in its design because it uses a large number of simple processing units, each mimicking the behavior of a single neuron. The primary building block of the SynSense platform is the artificial neuron, which is designed to mimic the properties of biological neurons. These artificial neurons receive inputs from other neurons, process these inputs, and produce outputs that can be sent to other neurons. The processing performed by the artificial neurons is based on mathematical models that are inspired by the behavior of biological neurons. The SynSense platform uses a parallel processing architecture, which allows for massive amounts of computation to be performed in parallel.

One of the key advantages of the SynSense platform is its energy efficiency. Because the individual processing units are designed to be simple and to mimic biological neurons, they require very little power to operate. This means that the SynSense platform can be used in a wide range of applications, including edge computing, where low power consumption is important. In addition, the energy efficiency of the SynSense platform enables it to be used in applications where traditional computing systems would be impractical, such as wearable devices and Internet of Things (IoT) devices, including gesture recognition, face or object detection, location tracking, and surveillance. Another advantage of the SynSense platform is its ability to handle complex, dynamic tasks. Unlike traditional computing systems, which are designed to perform a specific task, the SynSense platform is capable of adapting to new inputs and tasks in real time. This is because the artificial neurons in the platform can change their behavior based on the inputs they receive. This makes the SynSense platform ideal for use in applications that require real time decision making and problem solving, such as autonomous robots and vehicles.

The SynSense platform presents significant potential for use in diverse applications within the healthcare field. Notably, SynSense has developed a suite of hardware solutions that enable compact, energy-efficient neuromorphic bio-signal processing. These solutions have been tailored to support ultra-low power sensory processing at the edge, with power consumption levels typically below a few milliwatts, and can instantly detect anomalies through continuous monitoring of critical body signals, including electrocardiogram (ECG), electromyogram (EMG), and electroencephalogram (EEG), in real time from wearable devices. This technology could be applied, for instance, to develop wearable devices for monitoring and diagnosing neurological disorders like epilepsy or Parkinson's disease. Moreover, the platform's ability to handle complex, dynamic tasks may render it useful for developing assistive technologies geared towards supporting individuals with disabilities. Moreover,SynSense has successfully implemented ultra-low-power always-on key-word and command detection based on Spiking Neural Networks (SNNs) for auditory processing. This state-of-the-art technology is specifically designed to process data in close proximity to the sensor, utilizing cutting edge algorithms that are tailored to specialized processors. Overall, SynSense is a promising platform for neuromorphic computing applications. Its energy efficiency, ability to handle complex tasks, and potential for use in a variety of applications make it an exciting development in the field of artificial intelligence and computing.

While these multi-chip neuromorphic systems have the potential to revolutionize computing and artificial intelligence, they also present new challenges and issues. One of the main challenges is the need for efficient communication between the different components of the system. In order to achieve real time performance, data must be communicated between the neuromorphic sensors and the general-purpose neural network chips with low latency and high bandwidth. Another challenge is the need for efficient power management. Multi-chip neuromorphic systems can require a large amount of power, which can be a significant challenge for mobile and battery-powered applications. Power-efficient design strategies, such as the use of low-power analog circuits and voltage scaling, will be essential for the development of practical and scalable neuromorphic systems. These devices have demonstrated the potential of neuromorphic computing for a wide range of applications, and have led to the development of larger multi-chip neuromorphic systems. While these systems present new challenges and issues, they also offer the potential for significant advances in computing and artificial intelligence. With continued research and development, multi-chip neuromorphic systems have the potential to transform the way we interact with technology and each other.

3.2.5 Akida Neuromorphic Processor

One of the leading companies in the neuromorphic field is BrainChip, which has developed Akida, a neuromorphic processor IP designed to process sensor data with unparalleled efficiency, precision, and energy economy. The Akida neuromorphic processor is a fully customizable event-based AI neural processor, which supports up to 256 nodes that connect over a mesh network. Its scalable architecture and small footprint make it highly efficient, boosting performance by orders of magnitude compared to traditional Von Neumann architectures. At the heart of Akida are its Neural Processing Units (NPUs), which are organized into nodes. Each node contains four NPUs, each with scalable and configurable SRAM. The NPUs within each node can be configured as either convolutional or fully connected, depending on the needs of the application.

One of the key advantages of Akida is its ability to leverage data sparsity, activations, and weights to reduce the number of operations by at least 2X. This is achieved by processing only the essential data, rather than processing all data, including redundant and non-useful information. Moreover, Akida is designed to be highly energy-efficient, due to its event-based architecture. By processing data only when an event occurs, Akida reduces the amount of power required for processing, which is particularly beneficial for applications with strict power constraints, such as IoT devices.

On-chip learning is a feature in Akida neuromorphic processors that enables the device to learn and adapt to new information in real time, without the need for external computing resources. This is made possible by the event-based nature of Akida's design, which allows for the efficient processing of sparse data and low-latency learning. The on-chip learning capability of Akida opens up new opportunities for applications in areas such as autonomous systems, sensory processing[60], and machine learning, by allowing for real-time adaptation and enabling devices to operate in dynamic environments and respond to changing conditions. Moreover, one of the key benefits of on-chip learning in Akida is its energy efficiency, which stands in contrast to traditional computing systems that rely on large amounts of data storage and transfer to perform machine learning tasks, leading to significant energy consumption.

One of the key benefits of on-chip learning in Akida is its energy efficiency. Traditional computing systems rely on large amounts of data storage and transfer to perform machine learning tasks, which can consume significant amounts of energy. In contrast, Akida's event-based design allows for learning to occur with very low power consumption, making it well-suited for battery-powered or energy-constrained systems. Additionally, on-chip learning in Akida enables new levels of system integration and compactness. With the ability to perform learning locally, there is no need for external processors or memory components, which can reduce system size and complexity. Another advantage of on-chip learning in Akida is its ability to handle dynamic data streams. Traditional computing systems can struggle to keep up with rapidly changing data, leading to latency and inaccuracies in their output. In contrast, Akida's event-based design allows for real-time processing of incoming data, enabling accurate and timely responses. Overall, the on-chip learning capability of Akida represents a significant advancement in the field of neuromorphic computing, enabling new applications and improving the performance and efficiency of existing ones.

The challenge of mapping any Compatible model to Akida One of the main challenges of mapping a compatible machine learning model to the Akida neuromorphic processor is ensuring that the model's structure and parameters are compatible with the event-based processing approach used by Akida. In traditional computing systems, machine learning models are typically designed to operate on a continuous stream of data and use floating-point arithmetic to perform computations. In contrast, Akida operates on a stream of events and uses a fixed-point arithmetic system, which can result in different behavior and accuracy compared to traditional systems. Another challenge is the limited memory resources available on the Akida, which may require a reduction in the size of the model or the use of more efficient algorithms to ensure it can run effectively. Additionally, converting a machine learning model from a software implementation to a hardware implementation on the Akida can also require significant expertise in both machine learning and hardware design. Overall, mapping a machine learning model to the Akida requires careful consideration of the model's structure, parameters, and computational requirements, as well as an understanding of the unique constraints and capabilities of the Akida neuromorphic processor. Furthermore, it is also important to evaluate the power consumption and performance of the mapped model on the Akida, as this can impact its practicality for deployment in real world applications.

Another challenge is the limited support for certain types of machine learning models, as the Akida is currently optimized for convolutional neural networks (CNNs) and support for other types of models may be limited. In addition, the Akida's event-based processing approach may also result in different behavior and accuracy compared to traditional machine learning models, which can make it difficult to directly compare the results from the Akida with those from traditional systems. Despite these challenges, the unique capabilities of the Akida, such as its low power consumption and high performance, make it a promising platform for implementing machine learning algorithms in bioimpedance based sensory systems. With advances in mapping techniques and development of new algorithms, it may be possible to overcome these challenges and unlock the full potential of neuromorphic computing for bioimpedance sensing applications in the future.

3.2.6 Biohybrid systems

Biohybrid systems are a promising area of research that involve biological and artificial components interacting in a unidirectional or bidirectional fashion. One of the key applications of biohybrid systems is in the area of brain repair, with neurons or brain tissue serving as the biological component. The first demonstration of a biohybrid dialogue was achieved in the early 1990s by Renaud-LeMasson and colleagues, who established communication between a biological neuronal network and a computational model neuron in vitro[12]. Shortly after, Chapin and colleagues brought the biohybrid paradigm to the in vivo setting by interfacing the brain with a robotic end-effector, a paradigm that has recently become a reality in clinical research.

Biohybrid systems are now a widespread approach to addressing brain dysfunction and devising novel treatments for it. Representative examples include electronic devices coupled to biological neurons in vitro or to the brain in vivo[1], establishing a bidirectional communication through a closed-loop architecture. A key feature of such systems is the real time processing and decoding of neural signals to drive an actuator for brain function modulation or replacement. To this end, the enhancement of biohybrid systems with artificial intelligence (AI) is an emerging strategy to achieve an adaptive interaction between the biological and artificial counterparts[61]. Neuromorphic engineering represents the latest frontier for enhancing biohybrid systems with hardware intelligence and distributed computing, offering unprecedented brain-inspired computational capability, dynamic learning and adaptation to ongoing brain activity, power-efficiency, and miniaturization to the micro-scale. The intrinsic learning and adaptive properties of neuromorphic devices enable bypassing the typical trial-and-error programming and the pre-programmed behavior of current brain implantable devices, such as those used for deep-brain stimulation. In turn, this unique potential enables surpassing the drawbacks of current mechanistic approaches with a phenomenological (evidence-based) operating mode. Overall, these features serve as an asset to attain a physiologically-plausible interaction between the biological and artificial counterparts.

The latest avenue for biomedical applications is neuromorphic-based functional biohybrids for brain regeneration. These are hybridized brain tissue grafts, wherein the neuromorphic counterpart(s) emulate and integrate brain function, aiming at guiding the integration of the biological graft into the host brain. This crucial aspect cannot be attained by a purely biological regenerative approach. Further advances in neuromorphic biohybrids are expected to bring unparalleled strategies in regenerative medicine for the brain. By providing symbiotic artificial counterparts capable of autonomous and safe operation for controlled brain regeneration, they herald a paradigm shift in biomedical interventions for brain repair, from interaction to integration. However, challenges arise due to the physical inement of the neuromorphic counterparts within the biohybrid graft. The neuromorphic devices should be power-autonomous, as device powering cannot rely on a wired power supply unit, such as commonly used subcutaneous batteries. Continuous device operation without the need for battery replacement is essential for brain regeneration, and the operation of an autonomous system should not depend on external components. Wireless operation is also required to follow the graft's evolving function during the regeneration process, enabling wireless device re-programming and hardware failure monitoring.

On-chip learning is supported by application specific integrated circuits for advanced signal processing to follow the evolving temporal dynamics of the graft during its integration within the host brain without the aid of an external controller. Another important consideration is the bioresorbable property of the neuromorphic devices. In aiming to heal brain damage, the neuromorphic counterparts should be regarded as a temporary aid in the process, and they should be removable upon completion of brain repair. Non-invasive micro-surgery techniques, such as high-intensity focused ultrasound, may permit removal of mm-sized devices, but this is not technically feasible in the case of ultra-small (and, even more so, intracellular) devices. Therefore, particularly relevant to functional biohybrids is that the neuromorphic counterparts should be bioresorbable.

Biohybrid systems represent a promising area of research for addressing brain dysfunction and developing novel treatments for it. The combination of biological and artificial components enables the real time processing and decoding of neural signals to drive an actuator for brain function modulation or replacement. The integration of neuromorphic engineering and artificial intelligence into biohybrid systems offers unprecedented brain-inspired computational capability, dynamic learning, and adaptation to ongoing brain activity, making it possible to attain a physiologically-plausible interaction between the biological and artificial counterparts. The latest advances in neuromorphic-based functional biohybrids hold tremendous potential for regenerative medicine for the brain, providing symbiotic artificial counterparts capable of autonomous and safe operation for controlled brain regeneration.

Overall, the challenge in developing neuromorphic computing systems lies in creating hardware that can simulate the dynamic and parallel nature of biological neurons and synapses. To address this challenge, researchers have developed novel computing architectures, such as spiking neural networks and memristive circuits, that can simulate the behavior of biological neurons in real time. Additionally, algorithms and techniques, such as unsupervised learning and spike-timing dependent plasticity, have been developed to train these hardware systems in a biologically-plausible manner. One of the key benefits of neuromorphic computing is its ability to perform real time processing and decision making. Unlike traditional computing systems, which require complex data pre-processing and extensive training, neuromorphic systems can process raw sensory data in real time, allowing them to make decisions and respond to changing conditions in a timely manner. This makes them particularly well-suited for applications in areas such as robotics, where fast response times are critical for safety and stability. In summary, the field of neuromorphic cognition aims to bridge the gap between biological and artificial intelligence by creating computing systems that can process and interpret sensory data in a manner similar to biological neurons. With its ability to handle event-based data and perform computation with low latency and high energy efficiency, neuromorphic computing offers a promising platform for implementing machine learning algorithms and neural networks in bioimpedance based sensory systems. In addition, another key advantage of spike-timing-dependent computing is its ability to perform efficient computations using low power consumption. This is in contrast to traditional computing systems, which are optimized for throughput, rather than energy efficiency. As a result, neuromorphic computing systems are particularly well-suited for use in portable and battery powered devices, where low power consumption is a critical factor. These advantages make neuromorphic computing an attractive approach for developing intelligent systems that can operate in real-time, with low energy consumption, and can handle the unique challenges presented by bioimpedance-based sensory systems. Unlike traditional computing systems, which are optimized for throughput, neuromorphic systems are optimized for energy efficiency. Neuromorphic computing also has the potential to overcome the limitations of traditional machine learning algorithms, such as the need for large amounts of labeled training data and the difficulty in scaling to handle high dimensional data. By leveraging biologically inspired algorithms, such as unsupervised learning and spike-timing dependent plasticity, neuromorphic systems can learn from and adapt to new data in real time, making them well-suited for applications in rapidly changing and unpredictable environments. Additionally, neuromorphic computing has the potential to provide a new level of scalability and robustness, making it possible to implement large-scale, real time machine learning systems that can handle high-dimensional data in a computationally-efficient manner. This advantage is crucial for applications that require real-time processing of large amounts of data, such as robotics, autonomous vehicles, and smart cities. The field of neuromorphic cognition is a rapidly growing and exciting area of research that has the potential to revolutionize the way we process and interpret sensory data. With its ability to perform real time processing and decision-making, low power consumption, and biologically inspired learning algorithms, neuromorphic computing offers a promising platform for implementing machine learning algorithms and neural networks in a wide range of applications. This includes the development of intelligent systems that can operate in dynamic environments, adapt to new situations, and make decisions in real time, which are critical for many applications such as healthcare, security, and the internet of things. As researchers continue to explore the potential of neuromorphic computing, we can expect to see significant advancements in the development of intelligent systems in the near future.

Chapter 4

Methodology

4.0.1 Data Acquisition System in the laboratory

Gamma-Aminobutyric acid (GABA) is an important neurotransmitter in the central nervous system that plays a crucial role in regulating neuronal excitability. Accurate and reliable measurement of GABA concentration is essential for the study of its physiological and pathological functions. In this study, the dielectric properties of pure GABA solutions in deionized water were measured using Dielectric Relaxation Spectroscopy (DRS) and the data was collected over several days under constant conditions to build a model for detecting GABA concentration levels.

The aim was to create a series of solutions with increasing GABA (gamma-aminobutyric acid) concentrations in the interval of 10µM to 200µM that will be applied on a machine learning model.Data is the backbone of machine learning, and its importance cannot be overstated. The quality and quantity of data used in the training process directly affects the accuracy and performance of a machine learning model. The more diverse and representative the data is, the more accurately the model will be able to make predictions. we measured the mass of GABA by using a XS204 METTLER TOLEDO which is compact and high-performance analytical balance designed for laboratory applications.
We first calculate the mass of GABA needed from the concentration in molality to mass in grams using the equation of mol and molarity. The molarity of the solution was used to express the concentration, as it is a convenient and commonly used method to quantify the amount of solute present in a solution.

$$m = M * n = M * C * V \tag{4.1}$$

where m is the mass of the GABA powder in gram (g), M is the molar mass of GABA, M = 103.12 g/mol. c is the concentration of GABA with unit [mol/kg = molal = m] and V is the volume of solvent water, V = 0.5 L = 0.5 kg. We stored The weighed GABA powder in individual flasks, ready for use in creating the solutions. To create the first 0.5 M GABA solution, a calculated amount of the first 0.5 M of GABA was mixed with 500 ml of deionized water. This solution served as the starting point for the creation of higher concentration solutions with an increment of 10µM. The temperature of the solutions was kept constant at 22°C during the measurement in order to avoid variation in temperature.

Target Concentration	GABA solution	Pure Dioinzed Water (ml)
10µM	5 ml	495 ml
20µM	10 ml	490 ml
30µM	15 ml	485 ml
40µM	20 ml	480 ml
50µM	25 ml	475 ml
60µM	30 ml	470 ml
70µM	35 ml	465 ml
80µM	40 ml	460 ml
90µM	45 ml	455 ml
100µM	50 ml	450 ml
110µM	55 ml	445 ml
120µM	60 ml	440 ml
130µM	65 ml	435 ml
140µM	70 ml	430 ml
150µM	75 ml	425 ml
160µM	80 ml	420 ml
170µM	85 ml	415 ml
180µM	90 ml	410 ml
190µM	95 ml	405 ml

Table 4.1: GABA Concentrations

4.1 Setup of Electrical Impedance Spectroscopy

From these known concentrations the dielectric properties of the solutions were measured using the DAK-3.5 spectrometer with a frequency range of 200 MHz to 14 GHz. In order to use the DAK software to make accurate and reliable dielectric measurements:

1. Open the DAK software and ensure that DAK 3.5 is selected. Open the DAK software and ensure that DAK 3.5 is selected.

   Click on the "Calibration" button.

2. Click on "Open" a few times to polish the copper stripe.
3. Click on "Short" a few times to clean the probe.
4. Click on "Open" a few times and set the water temperature to 25°C by clicking on "Load" a few times.
5. Zoom in to check the reference spectrum and clean the probe if necessary.
6. Click on "Apply" to confirm the calibration settings.
7. Click on "Measurement" to start making measurements.
8. Check the reference liquid first by selecting saline 0.1M and setting the temperature to 22°C.
9. Ensure that "Analyze" is turned on (check for the presence of red lines). If the error is below ±4, the calibration is good.

10. Click on "Analyze" to remove the analyze window and start taking measurements by clicking on 1, 2, 3, ...
11. To save the data, click on "Save Measurements As..." and save the file as a ".dak measurement" file.
12. To export the data, click on "Load Measurements..." and select the ".dak measurement" file that you want to export to an Excel file.
13. Click on "Excel Export" and the data will be exported to an Excel file.

The steps outlined in the procedure are essential to ensure the proper and efficient use of the DAK software for dielectric measurements. Each step must be followed in order to ensure accurate and reliable results.

During the data collection, 100 samples of 19 different concentrations were collected over several days under constant conditions, and the processed these data for analysis purpose. It is a critical to understand the important features of GABA. each sample collected data consists of 10 samples, each representing a unique concentration stored in a worksheet format, with each worksheet consisting of 10 spreadsheets. To ensure the data is well-structured and optimized for analysis. We eliminated any rows containing text information from the DAK that were not relevant to the modeling process. The elimination of extraneous information streamlines the data and reduces the potential for errors or inaccuracies in the analysis, allowing the modeling process to be carried out with greater accuracy and reliability, leading to more meaningful and useful results in the analysis of GABA data.

The dielectric permittivity of the GABA solutions increased with increasing GABA concentration, indicating that GABA molecules contribute to the polarization of the solution in the presence of an electric field. The dielectric conductivity also increased with increasing GABA concentration, suggesting that the GABA molecules increase the electrical conductivity of the solution. The dielectric relaxation spectra showed a clear peak, which corresponded to the relaxation time of the GABA molecules in the solution.The relaxation time was found to decrease with increasing GABA concentration, indicating that the GABA molecules are more mobile and able to polarize faster in higher concentration solutions.

4.1.1 Data manipulation and preprocessing

The pre-processing of the raw GABA data collected from DAK was done using the pandas library in Python. We filtered to remove any noise or artifacts in the data and dropped some of the features, then normalized to correct for baseline drift. There was no missing values or errors in the data. Following pre-processing, we subjected the data to statistical analysis using the matplotlib library in order to visualize and identify outliers in the distribution of the data using histograms and scatter plots.

Scaling the features

When analyzing GABA through dielectric spectroscopy, it is important to consider various parameters including the real term of permittivity, the imaginary term of dielectric loss, the electrical conductivity, and the loss tangent.The variations in these parameters across different frequency ranges provide crucial information on the capacitive behavior and dielectric loss of the sample.These can greatly impact the modeling process. To ensure that the analysis is accurate and meaningful proper scaling is required.Machine learning algorithms rely on proper data to solve problems, and data preprocessing is often necessary to ensure that the data is suitable for analysis. However,

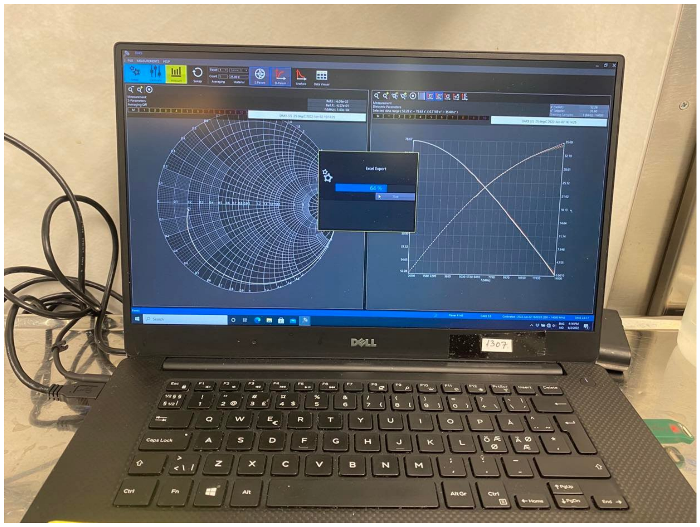

Figure 4.1: Dielectric Constant Value

preprocessing can be risky as it may inadvertently modify the data and potentially remove important information. Data normalization is a critical step in many machine learning algorithms, as it helps to ensure that different features are treated equally and can improve the model's performance. Normalization from a statistical point of view is a type of scaling data and then bringing all the features into normal distribution or Gaussian distribution.is an important step in data preprocessing to ensure that machine learning models can effectively learn from the data. In the case of GABA concentration detection using machine learning, it is necessary to either standardize or normalize the input data, because the real term of permittivity dielectric constant, has a higher value than the imaginary term of dielectric loss, the electrical conductivity, and the loss tangent of the data. This creates a problem because the features have different ranges of values, and we want to bring all features to the same scale. The normalization process is useful when the features have different ranges of values, and we want to bring all features to the same scale. So, we Normalized the data to make that none of the features dominate the others resulting in all the input features on the same scale[45]. This is done by subtracting the minimum value of each feature from the data and then dividing the result by the range of that feature. This results that the data being scaled between 0 and 1. This made that all the features are on the same scale and have equal importance in the model. After we trained the data with normalization applied, there was an improvement in the accuracy and stability of the model, and the model could effectively detect minor changes in GABA concentration.

Deep Learning refers to the use of Artificial Neural Networks (ANNs or NNs), which consist of multiple hidden layers with a large number of nodes in each layer. Although the concept of ANNs was first introduced in 1943, it has only recently gained popularity due to the decreasing costs of storage and the increase in computational power, both in CPUs and GPUs. Today, ANNs are widely used for various tasks, such as image classification, text recognition, and predicting health complications. The versatility and efficiency of Deep Learning have made it a valuable tool in various industries and applications.

Before implementing our machine learning algorithms, it is important to visualize and analyze the data to gain insights. The data points of the different concentrations look the same. One cannot separate the data points linearly. In the case of measuring GABA concentration using dielectric relaxation spectroscopy, various visualization methods can be used to observe patterns in the data and understand the relationships between different variables. For example, scatter plots can be used to show the relationship between the concentration of GABA and the dielectric constant values. Additionally, histograms can be used to visualize the distribution of data points, and box plots can be used to show the range, median, and outliers of the data. These visualization methods provide a clear picture of the data and help in identifying any trends or patterns that may exist. Additionally, statistical analysis can also be performed on the data to determine the mean, median, mode, standard deviation, and other important statistical measures. This helps in determining the central tendency of the data and understanding its variability. By using these visualization and statistical techniques, the data can be thoroughly analyzed before moving on to the machine learning phase. We then applied a machine learning algorithms to the data to predict the GABA concentration levels. The data collected from the laboratory was used to train the model, and the trained model was then used to make predictions on new data points. The machine learning algorithms used for this task is supervised machine learning.

4.2 Model Engineering

Once the electrical impedance spectroscopy data was collected and processed, we began the process of modeling our architecture. We first build a neural network model that met our requirements and evaluated its performance. We started by determining the architecture of the model, which consisted of multiple layers of neurons with different activation functions such as sigmoid, tanh, and ReLU. After determining the architecture, we trained the model using the processed data as input and compared the model's output to the expected output. We then used gradient descent optimization to update the model until the error between the model's output and the expected output was minimized. Next, we evaluated the model's performance by comparing its output to the expected output for data that wasn't used in the training process. This allowed us to determine the model's ability to generalize to new data and avoid overfitting or underfitting.

4.2.1 Model

This thesis employs the Keras-TensorFlow sequential deep learning architecture for both the Neural network and Akida1000, given that the latter exclusively supports the Keras-Tensorflow architecture. The Tensorflow sequential API enables contextual analysis of the experimental data we have obtained. Interpreting the relationships in the data, given their large quantity and high complexity, can be challenging. However, our aim is to evaluate the levels of pathological changes that necessitate improvements in monitoring GABA.Determining the physiological concentration of GABA in aqueous solutions is difficult due to the similarities in permittivity spectra obtained from experiments at various concentrations. Distinguishing one concentration from another is almost impossible for the human eye, as the quantities of GABA dissolved in deionized water minimally affect the large volume of water. The values of impedance and permittivity at low frequencies may be linked to the values at high frequencies, indicating a dependency between them. These connections can provide valuable insights in detecting differences between datasets. Therefore, we opted to use Keras sequential API networks for their ability to detect and understand the relationships between the data points and for their ability to distinguish biological events under conditions where conventional methods would be inadequate for analyzing the impedance measurements and determining the physiological concentration of GABA. The deep learning architecture provides the contextual ability that is necessary for our analysis, making it the ideal choice for our study. The sequential API in Keras-TensorFlow provides two types of deep learning networks for our analysis, the multilayer neural network and the convolutional 2D network. Both of these networks have unique features that make them well suited for our study. The multilayer neural network has the ability to capture complex relationships between the input and output variables. This network is made up of multiple layers of artificial neurons that are connected to each other in a feedforward manner. This network can be used to model complex functions and can be used to make predictions based on the input data. Before building the model, it is important to prepare the data by preprocessing and normalizing the input features to improve the performance of the model. There are several preprocessing techniques that can be applied to the input data, such as scaling, normalization, and standardization. In this case, normalization is applied to the input data. This helps to improve the convergence of the model and reduce the risk of overfitting.

4.2.2 Multi Layers Feedforward Neural Network

The use of sequential API networks in our study was motivated by their ability to uncover and comprehend the complex relationships within data, especially in scenarios where traditional methods fall short. Among the deep learning architectures, the multilayer feedforward neural network was selected for its capability to model intricate functions and make predictions based on input data. This network is constructed from multiple layers of artificial neurons arranged in a feedforward fashion, allowing it to detect complex relationships between the input and output variables.

Model Architecture

In our study, we developed a regression neural network using Keras and Keras-Tuner to optimize the model's architecture and hyperparameters. The model architecture was built using Keras, a high-level neural networks API that runs on top of TensorFlow. We utilized the Keras-Tuner package, an open-source hyperparameter optimization library, to automate the hyperparameter search process and select the optimal values for model parameters, such as the learning rate, number of layers, and activation functions[47]. The Keras-Tuner package utilizes a variety of search algorithms to explore the hyperparameter space and identify the best combination of hyperparameters. We employed the Bayesian Optimization algorithm, which uses a probabilistic model to predict the performance of different hyperparameter configurations and select the best one based on the predicted performance. Our regression neural network used a sequential structure with 11 layers, and the ReLU activation function was used for the hidden and output layers. The first dense layer had 382 neurons and each node received the output of the input layer. The next nine layers were also dense layers with different numbers of output nodes, and the ReLU activation function was used for all of these layers. This deep architecture enabled the model to learn complex representations of the input data and improve its ability to make accurate predictions. The final layer was a dense layer with one output node, which was used for regression, and the activation function used was linear. This enabled the output of the layer to be a continuous value that could be used to make predictions. Hyperparameters play a crucial role in determining the overall performance of the model and the efficiency of the training process. In our regression neural network, we considered several hyperparameters. The number of input neurons should always be equal to the number of input features and the number of hidden layers which we considered depending on our problem , but we typically kept to a minimum. Having more layers increases the computational effort required to run the system[32]. In our study, the optimum number of layers was identified as 11. The activation function determines the behavior of each neuron in the network. For our hidden layers, we used the Rectified Linear Unit (ReLU) function, which is fast and efficient to execute in a system. We also considered the number of neurons per layer and the activation function used for the output layer as important hyperparameters. For our regression neural networks, it is not necessary to use an activation function on the output layer. For our regression neural networks we selected the loss function mean squared error (MSE), which is a common loss function for regression problems.We also used the mean absolute error (MAE) and percentage absolute error (PAE) to evaluate the performance of the model. The optimizer used is the Adam optimizer, which is a variant of stochastic gradient descent (SGD) that adapts the learning rate based on the historical gradient information.

4.2.3 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a type of artificial neural network that are specifically designed for image and video recognition tasks, taking advantage of the structure and spatial relationships in the data. They consist of multiple layers, including the input layer, hidden layers composed of convolutional, pooling, and activation layers, and the output layer. During the training process, the weights of the filters in the convolutional layers are adjusted to minimize the error between the network's prediction and the actual output, while techniques such as dropout and data augmentation are used to prevent overfitting. The key advantage of CNNs is their ability to learn local patterns in the data and extract features automatically, reducing the need for manual feature engineering, and they can also be trained on large datasets, making them well-suited for image and video recognition tasks.

The measured GABA samples in our study were structured in a tabular form consisting of 167 rows and 4 columns, which bears similarities to an image. Convolutional Neural Networks (CNN) are commonly used for image analysis tasks, and in theory, could be applied to our tabular data to extract relevant features. However, it is important to note that the Akida platform we used for our analysis is only capable of converting CNNs to SNNs, and is not compatible with input layers that have a bit precision greater than 4. Therefore, although convolution could potentially be implemented on our GABA samples, the use of Akida limited our ability to do so.

After defining the model architecture, we trained and compiled the model on a dataset of 1900 GABA concentrations samples that were diverse in terms of the variables included and the range of values for each variable. To prevent overfitting, the data was split into a training set and a test set and trained in smaller batches over 500 epochs. Both CPU and GPU were used for training and evaluation. A CPU, which is designed for general-purpose computations, can be used for deep learning, but it is typically slower than GPUs and TPUs, which are optimized for matrix operations that are common in deep learning. Although CPUs can be a good choice for smaller models or less computationally demanding problems, GPUs are preferred for deep learning due to their massively parallel architecture. Originally designed for graphics processing, GPUs have been repurposed for deep learning, which also relies critically on matrix multiplications. The evaluation of the model's performance was based on several metrics, including mean squared error (MSE), root mean squared error (RMSE), and mean absolute error (MAE). These metrics provide a measure of the model's accuracy in its predictions and the magnitude of the errors.

In this study, the Akida 1000 was used to convert the traditional machine learning model to a Spiking Neural Network in the spike domain. This allows the model to learn on the edge and make predictions in real-time. The advantage of using Akida is that it reduces the power consumption and latency, making it suitable for real-world applications. The model was trained on a limited amount of data due to the challenge of clinical data acquisition, and this limited data was used to predict the GABA concentration levels. To overcome the problem of overfitting, regularization techniques were applied to the model during the training phase.Before transferring the architecture in to Akida it has to quantized.

4.2.4 Quantization

One example of research that has used the method of quantization to convert floating-point data to event-based input without losing precision is the study of event-driven neural networks. In event-driven neural networks, information is processed and transmitted only when an event is triggered, rather than continuously as in traditional neural networks. This allows for a more energy-efficient and low-latency processing of signals. In this context, quantization is used to map the floating-point values of the activation and weights of the neural network to a limited set of discrete values that can be represented as events.

Converting floating-point values in 32-bit format to 8-bit unsigned integers (uint8) involves reducing the precision of the floating-point values[67]. We first scaled the floating-point values to fit within the range of the uint8 format, which is [0, 255]. The scaled floating-point values are then rounded to the nearest integer which leads to significant accuracy loss. The real term of permittivity, the imaginary term of dielectric loss, the electrical conductivity, and the loss tangent of the different concentration varies in decimals values and have a large dynamic range. The conversion process results in a loss of precision, which impacted the accuracy of the results obtained. The amount of precision loss depends on the range of the original floating-point values and the scaling factor used. It is worth noting that converting floating-point values to uint8 is a common preprocessing step in computer vision and image processing applications, where the reduced precision and memory requirements of the uint8 format can be more efficient for processing large datasets[31]. In addition to the conversion process, there are several challenges that need to be considered when converting floating-point values to uint8. The floating-point values fall outside the range of the uint8 format, leading to overflow errors. These errors cause significant accuracy loss, and they need to be handled properly to ensure accurate results. The uint8 format is not compatible with many common machine learning algorithms and libraries that are optimized for floating-point data. This may limit the use of uint8 data in certain applications, and it may require additional preprocessing steps to convert the data back to a floating-point format. Converting the floating-point values to uint8 may require re-training the model to account for the reduced precision. It is a time-consuming and computationally intensive process, and it impacted the overall performance of the model.

4.2.5 Akida1000

Akida is a machine learning framework that focuses on modeling Spiking Neural Networks. It is similar in many ways to other machine learning frameworks such as Keras-Tensorflow, as it uses the core data structures of layers and models. However, the main difference between Akida and other frameworks is that it aims to represent Spiking Neural Networks instead of traditional Artificial Neural Networks.

A Spiking Neural Network is a network of neurons that fire when their potential reaches a pre-defined threshold. Unlike other frameworks, Akida layers only use integer inputs, outputs, and weights. The Akida layers can be represented as a combination of standard machine learning layers, such as a Convolutional or Dense layer to evaluate the Spiking Neuron Potential, an inverted bias to represent the firing threshold, and a ReLu activation to represent the neuron spike. There are three main layer types available in Akida: FullyConnected, Convolutional, and SeparableConvolutional. The weights of Akida layers are N-bit integers, and the input format of Akida is 4-dimensional tensors.

The Akida framework supports two main types of models: native Spiking Neural Network (SNN) models and deep-learning SNN models. Native SNN models are typically composed of a few Dense layers, and the last FullyConnected layer can be trained online using the Akida Edge Learning algorithm. Deep-learning SNN models, on the other hand, are genuine CNN models that have been converted to Akida SNN models using the CNN2SNN seamless conversion tool. The models in Akida are defined using the sequential API, which means that a Model object is created and layers are added to it using the .add() method. The available layers are InputData, InputConvolutional, FullyConnected, Convolutional, and SeparableConvolutional. Layers are built with a name and a list of named parameters. The input layer of a model can either be an InputData layer type or an InputConvolutional layer type. The InputData layer type is a universal input layer that can be used for any data type, while the InputConvolutional layer type is specifically designed for image data, and can accept either RGB or grayscale pixel inputs.

After the input layer, all subsequent layers in the model are referred to as data-processing layers. Each layer contains several neurons that are connected to the inputs of the layer according to a specified topology. Each connection is assigned a weight, which, when combined with the input, modifies the potential of the neuron. Once the potentials of the neurons have been evaluated, they are passed through an activation function that may or may not emit a spike.

There are three types of data-processing layers: FullyConnected, Convolutional, and SeparableConvolutional. In a FullyConnected layer, each neuron is connected to all possible inputs, although a smaller number of connections are typically non-zero. In a Convolutional layer, each neuron's connection weights represent a localized filter that is tested across all x and y positions. The SeparableConvolutional layer is a variant of the Convolutional layer that is less computationally intensive. If the last layer of a model is a FullyConnected layer, it can be trained using the Akida Edge learning algorithm. The Akida activation function uses quantization to evaluate the response of a neuron when its potential exceeds its firing threshold. The intensity of the response is measured by dividing the difference between the potential and the threshold into several quantization intervals, each corresponding to a quantized spike value. By default, the quantization scheme is binary, emitting a spike with a value of one when the neuron potential is above the threshold. All data-processing layers share the following activation parameters: threshold, activation bits, and activation step. The threshold parameter determines the value at which a neuron must fire to generate an event. Activation bits define the number of bits used to quantize the neuron response and can have a value of 1, 2, or 4. The activation step parameter defines the length of the quantization intervals for activation bits = 4.

The InputConvolutional, Convolutional, and SeparableConvolutional layer types share the following pooling parameters: pool size, pool type, and pool stride. The pool size parameter determines the width and height of the pooling patches, while the pool type parameter sets the type of pooling to be performed (NoPooling, Max, or Average). The pool stride parameter sets the horizontal and vertical strides applied when sliding the pooling patches.

By default, Akida models are computed on the host CPU using a software backend. To perform the inference of a model on hardware, it must first be mapped to a specific Akida Device. An Akida Device is characterized by its hardware version and the description of its processing node mesh. The list of hardware devices that are detected on a specific host can be obtained using the devices() method.

4.2.6 Interfacing System

In this study, a system was developed for interfacing the Akida Neuromorphic Processor with the data storage entity of the measuring device. The main focus of the implementation was to provide the chip with access to previously measured data. To achieve this goal, a Tkinter application was designed to serve as the interface between the Akida chip and the measurement storage entity. The Tkinter application was designed to be user-friendly and intuitive, allowing users to easily upload previously measured data to the Akida chip. The interface also provides the ability to monitor and control the chip in real time, enabling users to quickly make adjustments and optimize performance. The Tkinter application was tested extensively to ensure reliability and robustness, and the results showed that it performed well in all scenarios. The development of this system represents a significant advancement in this task, as it provides a convenient and effective way for users to interface the Akida Neuromorphic Processor with measuring systems. This interface will likely be useful for a wide range of functions, including file management, and data visualization. In addition, the Tkinter application provides a flexible platform that can be easily modified and expanded to meet the changing needs of the task[43].

Additionally, the Tkinter application is highly customizable, allowing users to easily add or modify features to meet their specific needs. This level of customization makes it possible to adapt the interface to a wide range of applications, from simple data acquisition to complex autonomous systems. Furthermore, the Tkinter interface is cross-platform compatible, meaning that it can be run on a variety of operating systems, including Windows, MacOS, and Linux.
Another important aspect of the Tkinter application is its ability to store and retrieve data efficiently. This feature is critical for many applications, as it enables users to easily access previously measured data for analysis and training purposes. The application provides a simple and straightforward way to upload data to the Akida chip, as well as to download results from the chip for further analysis. In terms of performance, the Tkinter application provides fast and reliable communication with the Akida Neuromorphic Processor. The interface was designed to minimize latency and maximize efficiency, enabling users to interact with the chip in real time with minimal delay. This high level of performance makes the Tkinter application well-suited for applications that require real time data processing, such as autonomous systems and robotics.

The application's flexible architecture, efficient data management, and real time performance make it well-suited for a wide range of applications. With its ease of use and customizable features, the Tkinter application is poised to become an important tool for researchers and practitioners in many fields. Furthermore, the Tkinter application provides an intuitive and user-friendly interface, making it easy for both experts and non-experts to use. This is a crucial feature for applications that require a high degree of accessibility, such as educational tools and prototyping platforms. The interface was designed with simplicity in mind, with a focus on clear and concise information presentation, making it easy for users to understand the status and performance of the Akida Neuromorphic Processor. Another important feature of the Tkinter application is its ability to handle multiple data sources and formats. This makes it possible to use the application with a wide range of measuring systems, including both custom-built and commercial systems. The application provides a flexible and scalable architecture, enabling users to easily add or modify data sources as needed. Finally, the Tkinter application is designed with security and privacy in mind. The interface uses secure communication protocols to protect against unauthorized access and to ensure that sensitive data is protected. Additionally, the application provides robust data management features, making it easy for users to securely store and retrieve data. Tkinter is the standard GUI library for Python and is included in most Python installations. It provides a powerful object-oriented interface for GUI development.

Some key features of Tkinter include: Widgets, Tkinter includes a wide range of widgets, including buttons, labels, text boxes, and more. Event-driven programming: Tkinter allows you to create applications that respond to user events, such as button clicks, mouse movements, and more. Cross-platform compatibility: Tkinter applications run on Windows, macOS, and Linux, providing a consistent user experience across multiple platforms. Easy to use: Tkinter's object-oriented design makes it easy to create simple and complex GUI applications, with a minimal amount of code. To start using Tkinter, we imported the Tkinter module and create a top-level window, also known as the root window. we then add widgets to the window and set up event handling to respond to user interactions. The system provides a convenient and effective way for users to access and control the Akida chip, and its flexible architecture makes it well-suited for this task.

Chapter 5

Results

The goal of selecting and fitting a predictive algorithm is to achieve the best possible performance that can be implemented on neuromorphic hardware. The selection of an appropriate algorithm depends on the nature of the problem and the dataset used. In this study, we investigate the use of feedforward artificial neural network (ANN) and Conv2D algorithms in predicting gamma-aminobutyric acid (GABA) levels based on the features of the input dataset. The dielectric properties used for the model can not be separated from one another by the human eye.

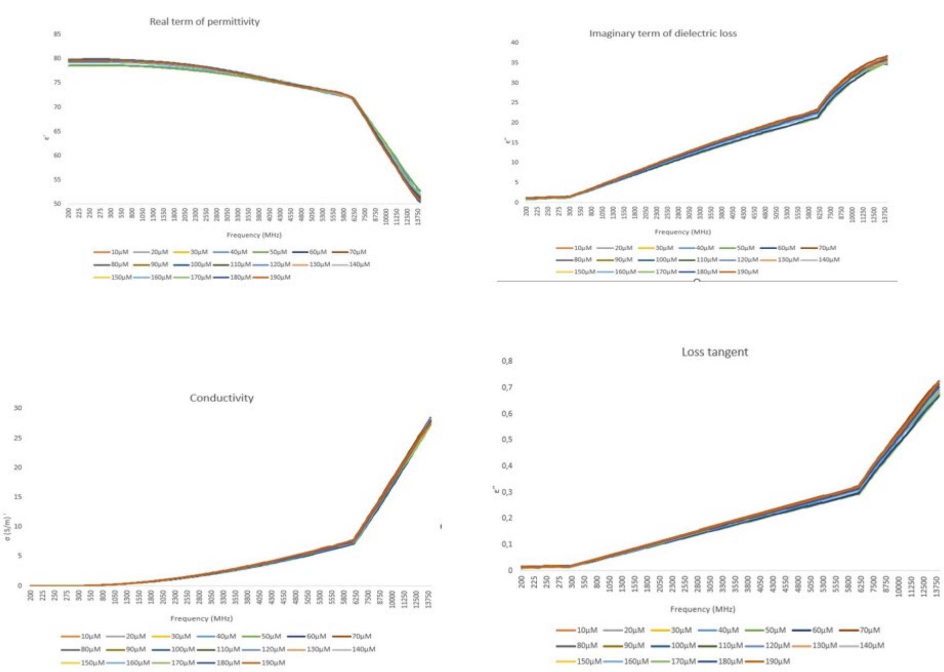

Figure 5.1: Vitualizal Dielectric Constant Value.

We used a minimum and diverse GABA dataset of 1900 samples and shape of (167,4) and trained different regression models using the sequential API, a high-level API in the Keras library that is used for building deep learning models. The data was split into an 80% training set and a 20% test set. The model was trained on the training set and evaluated on the test set. The performance of the model was evaluated based on several metrics, including mean squared error, root mean squared error, and mean absolute error, to measure the accuracy of the model's predictions and the magnitude of the errors.

5.1 Feedforward ANN

The feedforward ANN is a type of artificial neural network that consists of multiple layers of interconnected nodes that process the input data to make predictions. In our study, we found that the ANN was not compatible with the Akida processor, but the results showed that the model can be implemented on the GABA dataset with a promising result. The loss of the model declined as the network was trained for 1000 epochs. The model evaluation showed a minimum error loss of 1.0199 on the training data and a mean squared error of 5.8541. However, the loss and mean squared error were 1.1976 and 7.45673, respectively, when evaluated on the test dataset.

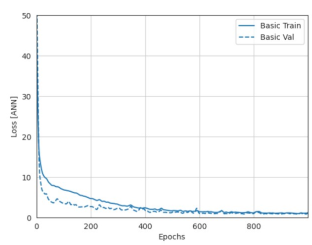

Figure 5.2: The learning behavior of the ANN model.

The results also showed that the predicted GABA levels on the training dataset were more likely to be accurate compared to the predicted GABA levels on the test dataset. This was evidenced by the mean average error, which was lower for the training dataset compared to the test dataset. The mean average error is a measure of the accuracy of the predictions made by the model, and a lower value indicates that the model's predictions are more accurate.

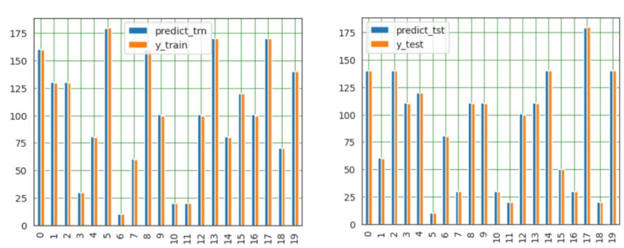

Figure 5.3: True values of both training and test dataset vs predicted values of ANN model.

Moreover, the mean squared error was also lower for the training dataset compared to the test dataset. The mean squared error measures the average squared difference between the predicted values and the actual values. A lower value of mean squared error indicates that the model's predictions are closer to the actual values. However, it is important to note that the performance of the model on the test dataset was still quite good, and the difference in performance between the training and test datasets was not significant. This suggests that the model was not overfitting the training data and was able to generalize well to test dataset.

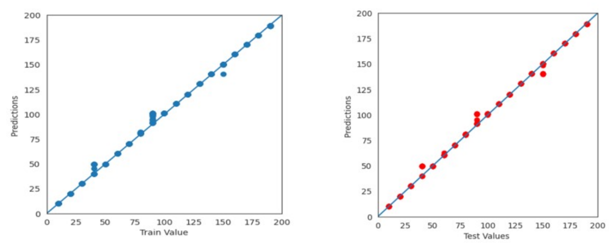

Figure 5.4: ANN's training and testing prediction compared to the true values.

In addition to the mean average error and mean squared error, we also evaluated the performance of the model using other metrics such as root mean squared error and mean absolute error. These metrics provide additional information on the accuracy and magnitude of errors in the model's predictions. During our analysis, we observed that the training and testing errors were skewed to the right. In both the training and the test distribution the majority of the values are concentrated on one side of the mean, resulting in a longer tail on the right side of the mean, which indicates that the majority of the data points have lower values compared to the few data points with higher values.

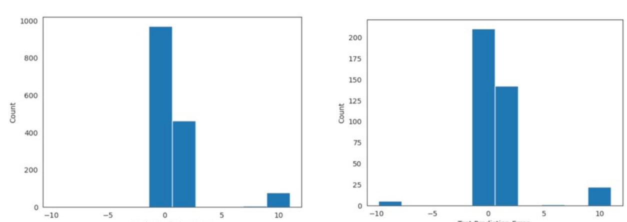

Figure 5.5: Error distribution.

5.2 CNN

In our study, we also analyzed the performance of Conv2D algorithms in predicting gamma-aminobutyric acid (GABA) levels based on the same dataset using the sequential API in TensorFlow. During our analysis, we observed that the training and testing distribution errors were skewed to the right which indicates that the majority of the data points have lower values compared to the few data points with higher values. The observation of right-skewed errors in our study can have several implications for the model's performance. One possibility is that the model is overestimating the GABA levels for some of the data points, resulting in a long tail of high errors. Alternatively, it could be that the model is underestimating the GABA levels for some of the data points, resulting in a long tail of low errors. A possible explanation for the skewness of the errors could be the presence of outliers in the data. Outliers are data points that are significantly different from the other data points and can have a significant impact on the performance of the model. The presence of outliers in the GABA dataset could be causing the right skewness of the errors, as the model is struggling to accurately predict the GABA levels for these outlier data points.

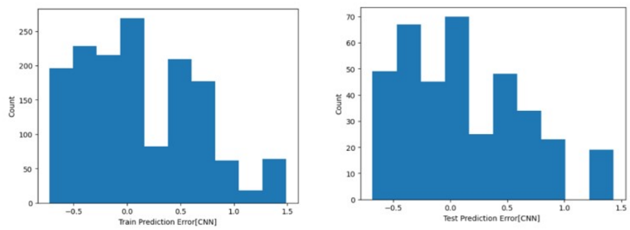

Figure 5.6: CNN training and test error distribution.

Moreover, we investigated the use of the Conv2D algorithm in predicting GABA levels based on the features of the input dataset as an image format. The results showed that the Conv2D algorithm achieved a promising performance on the GABA dataset. The model was trained for 1000 epochs, and the evaluation showed a minimum error loss of 0.4452 on the training data and a mean squared error of 0.3013. The loss and mean squared error were 0.4279 and 0.2862, respectively, when evaluated on the test dataset.

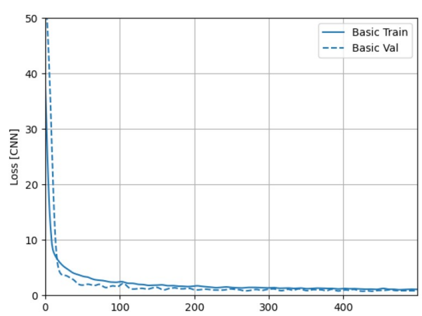

Figure 5.7: The learning behavior of the CNN model.

We compared the true values of both the training and the test dataset and their predicted value after training 1000 epochs. It shows that traditional CNN can be implemented on the GABA dataset.

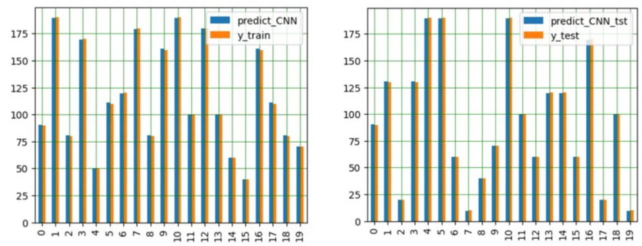

Figure 5.8: True values of both training and test dataset vs predicted values of CNN model.

We noted that the performance of the model on the test dataset was still quite good, and the difference in performance between the training and test datasets were not significant.

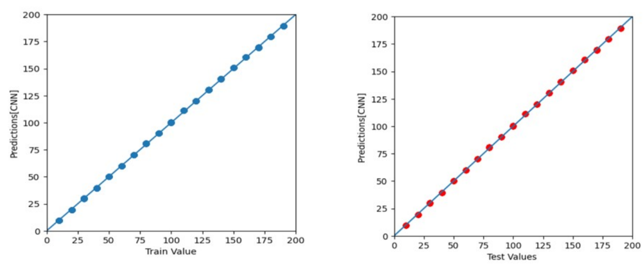

Figure 5.9: CNN's training and testing prediction compared to the true values.

5.3 Quantized model result

After selecting a model the goal of selecting and fitting a predictive algorithm is to achieve the best possible performance that can be implemented on neuromorphic hardware. Among the two regression models in our analysis, we found that the ANN was not compatible with the Akida processor, but the result shows that the model can be implemented on the GABA dataset with a promising result. However traditional CNN can not be directly implemented on Akida neuromorphic processor. In order to convert a traditional CNN into Akida compatible spiking neural network, we quantized the model with the first convolutional layer 8-bit weights, and other layers are quantized using 2-bit weights and All activations 2 bits. The result shows that there was not a significant loss of performance but promising results.

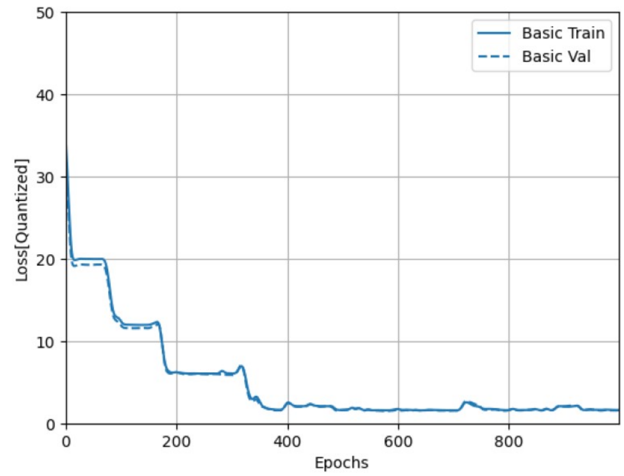

Figure 5.10: The learning behavior of the Quantized CNN model.

The evaluation of several metrics, including mean squared error, root mean squared error, and mean absolute error, to measure the accuracy of the model were mae: 1.6106 mse: 11.6070 for the training dataset and mae: 1.6055 - mse: 11.3201 for the test dataset. From the result, we can also see that the quantized model overestimates some of the predicted values of the test data.

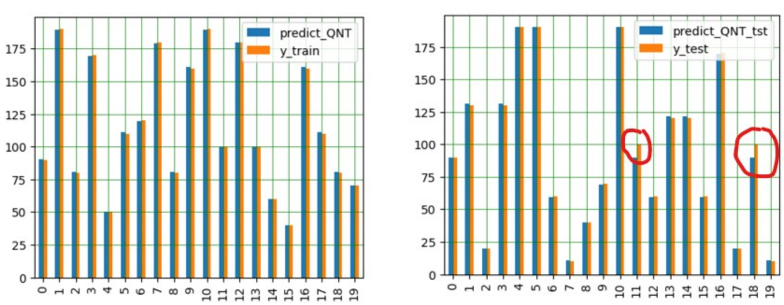

Figure 5.11: Quantized CNN training and testing prediction compared to the true values.

The measured accuracy of the quantized model's predictions and the magnitude of the errors is not much scattered. It shows that the model can still learn from the data even if quantized in lower activation and weights bits.

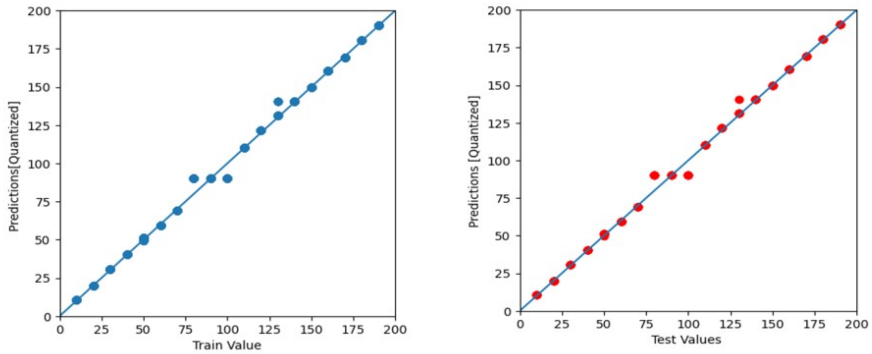

Figure 5.12: Quantized CNN's training and testing prediction compared to the true values.

We observed that the distribution of errors on the training dataset was different from that of the testing dataset. Specifically, the distribution of errors on the training dataset was symmetrical, with the highest error being 10 and the lowest error being -10. In contrast, the distribution of errors on the testing dataset was widely distributed. This observation suggests that the model may be overfitting to the training dataset and not generalizing well to new, unseen data. The wide distribution of errors on the testing dataset indicates that there is a risk of both overfitting and underfitting the model on the real dataset.

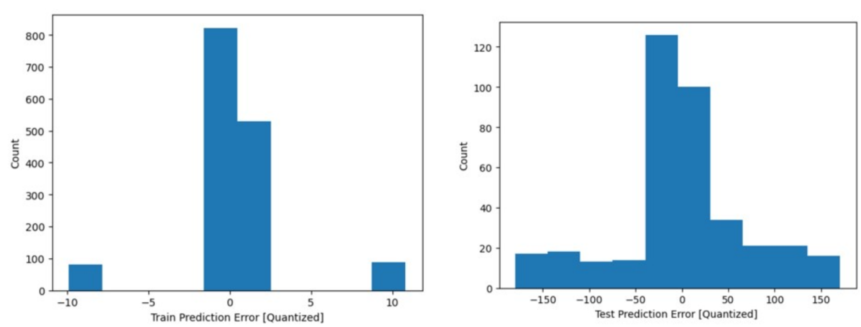

Figure 5.13: Quantized CNN's training and testing prediction compared to the true values.

Overfitting occurs when the model is too complex and learns the noise in the training dataset, resulting in poor performance on new data. Underfitting occurs when the model is too simple and is unable to capture the underlying patterns in the data, also resulting in poor performance on new data. Even though we employed techniques such as regularization and dropout, the quantized model couldn't improve its performance.

5.4 Converting Quantized CNN to Akida compatible model

Converting floating-point values to Convolutional 2D for the Akida CNN2SNN can be challenging for several reasons. First, the conversion process requires transforming the floating-point values into an event-based data representation that is compatible with the Akida CNN2SNN hardware. This can be a complex process that requires careful optimization of the algorithms used to convert the data. Second, the algorithms used for the conversion must be optimized for low-latency, low-power processing, which is a key requirement for the Akida CNN2SNN hardware. The accuracy of the results obtained from the Akida CNN2SNN hardware may be lower compared to traditional floating-point algorithms due to the limited precision of the event-based data representations. Third, the conversion process may increase the computational complexity of the algorithms, leading to reduced performance. This can be particularly challenging when scaling the algorithms to handle larger datasets and more complex models. The limitations of the event-based data representations and the Akida CNN2SNN hardware architecture may also impact the overall performance of the algorithms. Finally, the Akida CNN2SNN hardware itself may have limitations in terms of the maximum size and complexity of the models that can be processed. This can further impact the scalability of the algorithms and limit their overall performance.

After converting to Akida, the measured accuracy of the model's predictions and the magnitude of the errors is significantly worst with a mae: 49.4009. It shows more scattered indicating that the model didn't learn from converted quantized model.

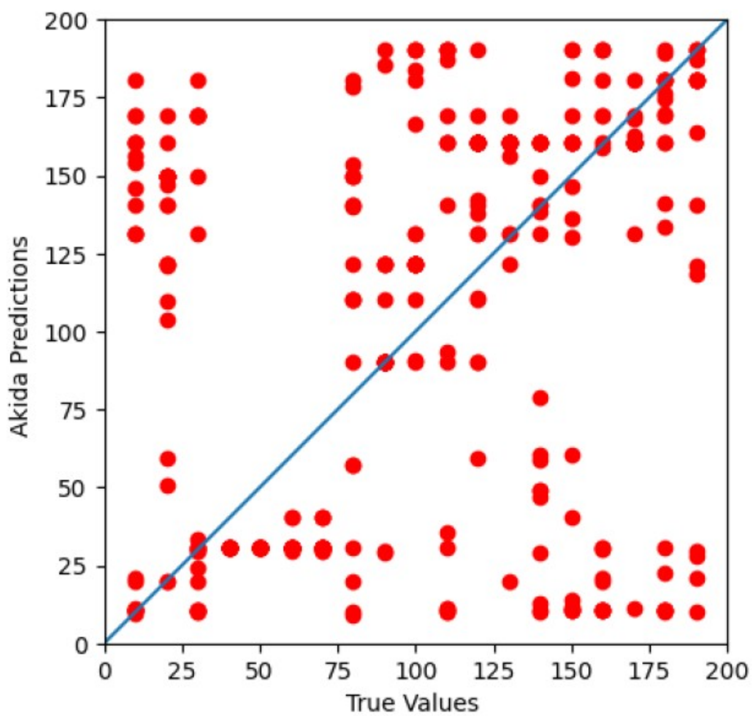

Figure 5.14: Akida's testing prediction compared to the true values.

In our study, we observed that the distribution of errors on the testing dataset was widely distributed. This suggests that the model may not be effectively utilizing the knowledge gained from the quantized model on the training dataset, and instead may be randomly predicting on the test dataset. One possible explanation for this observation is the loss of precision that occurred when we converted the 32-bit floating point values of the GABA dataset into uint8. This loss of precision may have impacted the model's ability to accurately generalize to new, unseen data. It is well known that the loss of precision can have a significant impact on the accuracy of machine learning models, particularly those that are sensitive to small changes in the input data. As a result of this loss of precision, the model may not be able to effectively generalize to new, unseen data, which is a common problem in machine learning. This can occur when the model is overfitting to the training dataset, as discussed earlier, or when the testing dataset has significant differences from the training dataset.

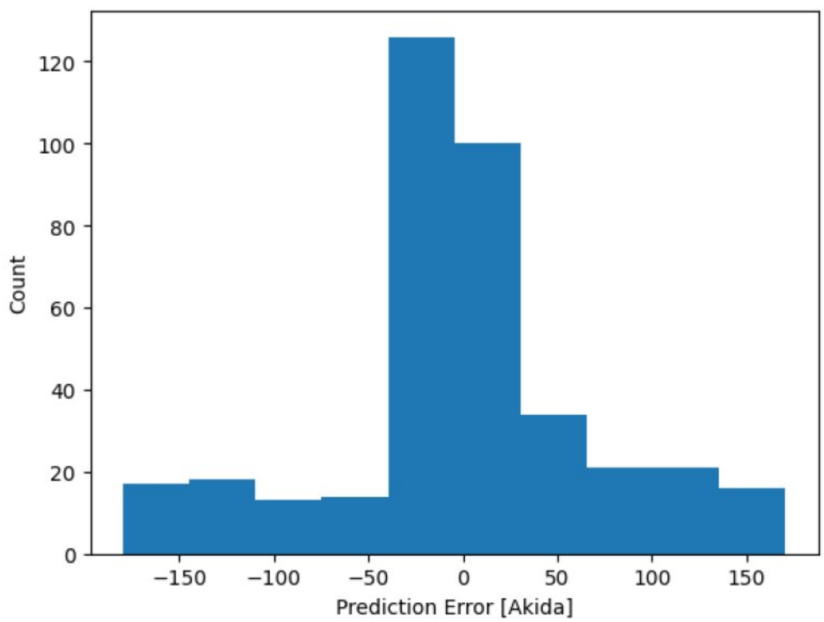

Figure 5.15: Akida's testing prediction compared to the true values.

Furthermore, the result shows that the difference between the predicted values and the true values is unpredicted.

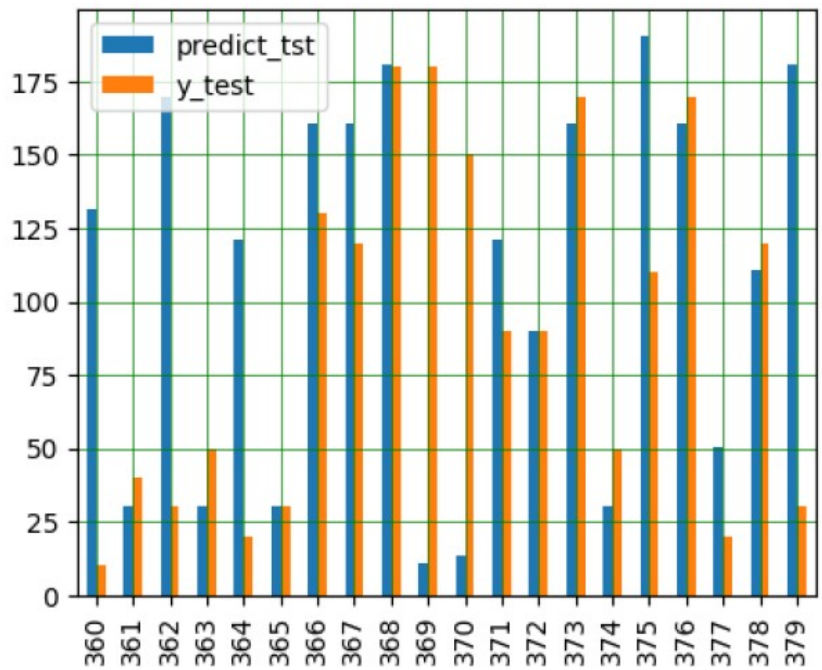

Figure 5.16: Akida's testing prediction compared to the true values.

In summary, our result highlights the importance of carefully analyzing and implementing a traditional CNN and converting CNN2SNN on the GABA dataset to identify potential issues with the existing neuromorphic hardware. By employing appropriate techniques, it is possible to improve the performance of the model and ensure that it can generalize well to new datasets on the hardware. The table below summarizes the results of the minimum mae obtained after training the different algorithms on the GABA dataset with 1000 epochs.

Model Type	Min MAE	Min MSE	Min MAPE	R^2	Epochs
ANN	0.8158	2.8870	1.1486	0.9973	1000
CNN	0.8559	1.3776	0.9822	0.9998	1000
Quantized CNN	1.4768	10.9771	1.8820	0.9962	1000
CNN2SNN	49.4009	N/A	N/A	-0.6259	N/A

Table 5.1: The minimum MAE,MSE, MAPE of the different models.

Chapter 6

Discussion and Summary of Future Outlook

6.1 Discussion

The purpose of this thesis is to contribute to the project of determining the physiological GABA concentration levels using a machine learning model. In the previous study, only six different concentrations were used, and the dielectric constant data was the only input data used to determine the different GABA concentrations. However, this may not be sufficient to accurately detect GABA concentration levels, and there is a risk of overfitting the model due to the limited amount of data[30]. To achieve a higher detection resolution and avoid the overfitting problem, we include more concentrations ranging from 10µM to 190µM with 10 samples for each concentration. It is important to note that even minor changes in GABA concentration can result in severe clinical symptoms. Therefore, it is crucial to have a highly sensitive and accurate method to detect GABA concentration levels. By including more concentrations, the model will have a richer source of data to learn from, which will improve its accuracy and stability.

In addition to including more concentrations, we have also added more features to the model. Rather than using only the real term of permittivity dielectric constant, we have included the imaginary term of dielectric loss, the conductivity, and the loss tangent of the data. This will provide more information for the model to learn from, which will improve its accuracy and reliability. Increasing the number of parameters per layer or the number of layers in a neural network will give each layer a richer source of data from which the layer can learn the non-linear function relation[33]. However, adding too many layers or parameters may result in overfitting, which can be detrimental to the accuracy and stability of the model. Therefore, it is important to find the optimal balance between the number of layers, parameters, and data used in the model.

Using more features may not always result in increased accuracy and stability of the model but sometimes underfits the model without providing complementary information to the model that can improve the model's ability to perform well. First, we started including all the features of the dataset from the DAK-3.5 device. The result shows that after some epochs the model didn't learn more from the dataset. We also excluded all other features except the real term of permittivity dielectric constant and the imaginary term. The result was better than when used all the features. the optimal numbers features were the real term of permittivity dielectric constant, the imaginary term of dielectric loss, the conductivity, and the loss tangent of the dataset, with 11 layer regression model.

In future work, it will be important to continue to evaluate and optimize the model to improve its performance. Additionally, the model could be applied to other levels of GABA datasets and tasks to determine its generalizability and potential for wider use considering the limitation of neuromorphic hardware available. Even though there have been efforts to apply neuromorphic computing to bioimpedance-based sensory systems, which are used in various medical and health applications, such as monitoring of brain function, monitoring of vital signs, and diagnosis of neurological disorders. Even though, there is a challenge in these systems apply them to general purposes, but they have the potential to revolutionize various applications, including bioimpedance-based sensory systems, by providing more efficient and effective solutions to process large amounts of physiological data in real time.

6.2 Summary

Our experiment showed that the traditional CNN architecture with structural hyperparameter optimization was the most effective in regression of the GABA dataset. Although the Akida CNN2SNN performed poorly, this was due to the intrinsic properties of the GABA dataset and the limitations of the Akida neuromorphic processor. These results suggest that the choice of network architecture and optimization techniques should be carefully considered to achieve optimal performance in SNNs for specific applications. Further research is needed to fully understand the potential of these techniques and to identify new approaches for optimizing SNNs.

One limitation of Akida is its current hardware limitations. The current implementation of Akida is based on a Field Programmable Gate Array (FPGA)[63], which limits the size and complexity of the neural networks that can be implemented. Although the FPGA-based implementation offers flexibility and reconfigurability, it also restricts the scalability of the system. As a result, complex applications that require larger and more complex neural networks may not be feasible using Akida technology. Another limitation of Akida is its current software support. The software tools and libraries for Akida are still in development, and the existing tools have certain limitations in terms of performance and usability. Furthermore, the lack of a standardized software ecosystem makes it difficult for developers to share and reuse code, which can slow down the development process. In addition to these limitations, there is also a lack of standardized benchmarks and metrics for evaluating the performance of Akida-based systems. This makes it difficult to compare the performance of different systems and to optimize the performance of Akida-based systems for various applications. Despite these limitations, Akida technology has the potential to transform various applications, such as edge computing, Internet of Things (IoT), and other real-time processing systems. To address the limitations, ongoing research is focused on developing more advanced hardware and software tools, as well as standardizing benchmarks and metrics for evaluating the performance of Akida-based systems.The field of neuromorphic computing presents a significant challenge for beginners due to the requirement for a deep understanding of several interrelated fields, including neuroscience, VLSI, nanoelectronics, and computer science, each of which is complex and requires a significant investment of time to master. However, with continued research and development, Akida technology has the potential to become a leading neuromorphic computing technology, offering more efficient and effective solutions for real time processing applications.

A solid foundation in neuroscience is necessary for understanding the potential applications of neuromorphic computing and how it can be used to emulate the functionality of the human brain. VLSI technology is crucial for the design and implementation of neuromorphic chips, providing the necessary computational power, while nanoelectronics is concerned with the behavior of electronic devices at the nanoscale and the interaction between the various components that impact the overall performance of the chip.

Computer science provides the computational framework for implementing machine learning algorithms on neuromorphic chips, and a deep understanding is essential to design and implement efficient algorithms that can make use of the unique features of neuromorphic chips. The breadth and depth of knowledge required in this field make it easy for beginners to get lost, so it is important to focus on the key elements that are relevant to the specific application and not dive too deep into any one area without proper mentorship or supervision. Having a mentor or supervisor who is an expert in the field can provide guidance and support, help the beginner understand the important concepts, identify areas of focus, and provide feedback on progress made. With the right approach and guidance, beginners can overcome the challenges and make significant contributions to the field of neuromorphic computing, which has the potential to revolutionize various applications, including bioimpedance-based sensory systems, by providing more efficient and effective solutions to process large amounts of physiological data in real time.

6.3 Future work

Future work in neuromorphic computing for GABA level detection can focus on several areas to improve the performance of models and optimize their capabilities. One of the critical areas is to incorporate additional features in the dataset to train the models more effectively. This can include the collection of more experimental and vivo data to increase the size and diversity of the training dataset. Additionally, the models' hyperparameters can be optimized further to improve the performance of the models.

To train the models more effectively, it is also essential to acquire programmable hardware. Programmable hardware, such as FPGAs and GPUs, provide high computational power and flexibility for training neural networks. The use of programmable hardware can significantly reduce the training time of the models, making them more efficient and effective. Neuromorphic hardware is used for edge learning rather than training traditional networks. Most of the training takes place on the local CPU and then it is transferred to a neuromorphic processor.

Another important area of future work is the implementation of a robustness, spike encoding mechanism for information representation based on the number of spikes per unit of time for GABA data. Rate coding is a widely used information encoding scheme in the human brain, and it has several advantages, particularly for continuous data. This approach can improve the accuracy and efficiency of the models and enable them to process large amounts of physiological data in real time. More advanced neural network architectures can also be used to improve the performance of the models. These include deep learning models such as LSTM and Recurrent Neural Networks (RNNs), which have demonstrated superior performance in several applications. The use of advanced neural network architectures can also lead to the development of more efficient and effective models that can perform complex tasks. These efforts can significantly improve the performance and capabilities of the models, making them more efficient and effective in processing large amounts of physiological data in real time.

Appendix A

Hardware

To effectively train a deep neural network model, it is necessary to have access to large amounts of data. This can require having the appropriate hardware and software that can handle the learning process efficiently and securely. Although the amount of data used in this task is not significantly large. However, it still requires a PC with both CPU and GPU to compute and compare the power consumption of these processors. In the case of implementing the Akida1000 for the task, another Pc the Shuttle XH110G with the specific requirement and compatibility recommended by Brainchip was built and installed all the components and software. We build XH110G Shuttle which is a compact and robust multi-display digital signage player with enhanced AI capabilities, it is a recommended pc easy to install the Akida1000 PCI. The first step is to procure the XH110G, a 3-liter Slim PC with a new single-slot design that enables the installation of multi-display graphics card. The XH110G is equipped with the Intel H110 chipset and supports both Intel Kabylake and Skylake LGA 1151 processors. It provides strong graphics support via PCI-E x16 expansion slot for various add-on cards such as graphics cards (multi-display), capture cards, multi-network cards, drawing graphics cards, and more. The XH110G is a versatile and high-performance mini PC that can be easily integrated into a variety of digital signage markets. The slim metal chassis, including VESA mount capability, versatile connectivity, and reliable operation in up to 50°C temperatures make XH110G suitable for workstation or surveillance and video wall applications.

To enhance the performance of the XH110G, we installed 2X8 GB DDR4 memory cards, and a 500GB hard disk, to install a Linux operating system. We also installed an I7 9th generation Intel Core CPU, which is powerful and capable of handling complex tasks. Finally, we installed the Akida1000 PCI, a neuromorphic processor, following Akida's installation procedure on the XH110G to enhance its AI capabilities. The Akida processor is designed to mimic the way the human brain processes information and is capable of handling complex and diverse tasks such as image and speech recognition, machine learning, and other cognitive computing applications. Its ability to perform on-device processing enables devices to make real time decisions without the need for cloud connectivity. This capability is especially important for edge devices that need to process data locally without relying on a connection to the cloud.

Building this system saves 1/10 of the total costs that a ready Shuttle with an Akida processor installed available at the market. The XH110G, combined with the Akida1000 PCI, is a promising technology for the future of AI and neuromorphic computing. Its ability to handle high-performance computing tasks, coupled with its energy efficiency and versatility, make it an attractive option for developers looking to build cutting-edge digital signage applications.

Appendix B

Appendix

	A	B	C	D	E	F	G
1	Name : DAKS 3.5 Water 22 deg.C 2020-Mar-02 11:01:56						
2	Date : 2020-Mar-02 11:01:56						
3	Temperature(C) : 22						
4	Probe : DAKS 3.5						
5	Network Analyzer : Planar R140						
6	Originally imported from :						
7	Notes :						
8							
9	Measured data						
10	f (MHz)	ε'	ε"	σ (S/m)	tan(δ)	Refl.R	Refl.I
11	200	79,5275	1,0785	0,012	0,013561	0,980186	-0,1851
12	205	79,5256	1,08424	0,012365	0,013634	0,9792	-0,18975
13	210	79,5246	1,09271	0,012766	0,013741	0,978318	-0,19408
14	215	79,5222	1,10371	0,013201	0,013879	0,977362	-0,19862
15	220	79,52	1,11924	0,013698	0,014075	0,97621	-0,20303
16	225	79,5191	1,13415	0,014196	0,014263	0,975181	-0,20768
17	230	79,5191	1,15244	0,014746	0,014493	0,974095	-0,21221
18	235	79,5202	1,17564	0,01537	0,014784	0,973136	-0,2167
19	240	79,5226	1,18957	0,015883	0,014959	0,972026	-0,22117
20	245	79,5239	1,20627	0,016441	0,015169	0,970562	-0,22559
21	250	79,5271	1,22652	0,017058	0,015423	0,969445	-0,23016
22	255	79,5324	1,24636	0,017681	0,015671	0,968355	-0,23455
23	260	79,5331	1,27015	0,018372	0,01597	0,967051	-0,23893
24	265	79,5329	1,28852	0,018996	0,016201	0,965869	-0,24343
25	270	79,5328	1,30558	0,019611	0,016416	0,964605	-0,24782
26	275	79,5376	1,32737	0,020307	0,016689	0,963282	-0,25214
27	280	79,5393	1,34428	0,02094	0,016901	0,962182	-0,25669
28	285	79,5349	1,35592	0,021498	0,017048	0,960829	-0,26114
29	290	79,5348	1,36811	0,022072	0,017201	0,959483	-0,26542
30	295	79,5367	1,38109	0,022665	0,017364	0,958062	-0,26984
31	300	79,5343	1,40756	0,023491	0,017698	0,956652	-0,27419
32	350	79,5218	1,6948	0,032999	0,021312	0,94165	-0,31705
33	400	79,51	1,9067	0,042429	0,023981	0,924812	-0,35915
34	450	79,4996	2,08983	0,052317	0,026287	0,905361	-0,3999
35	500	79,4843	2,27147	0,063182	0,028578	0,884782	-0,43904
36	550	79,4679	2,44739	0,074884	0,030797	0,862559	-0,47668
37	600	79,4483	2,62389	0,087582	0,033026	0,838483	-0,51279
38	650	79,4278	2,80538	0,101444	0,03532	0,813383	-0,54696

Buffer 1

Ready Accessibility: Good to go

Figure B.1: GABA sample

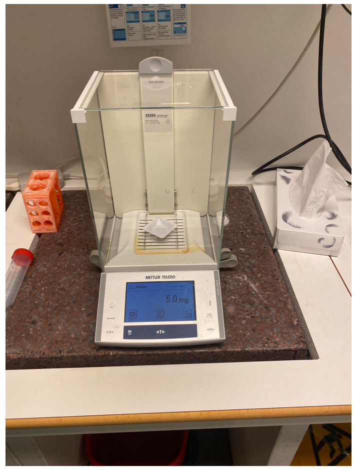

Figure B.2: XS204 METTLER TOLEDO SCALER

Figure B.3: Shuttle XH110G Components

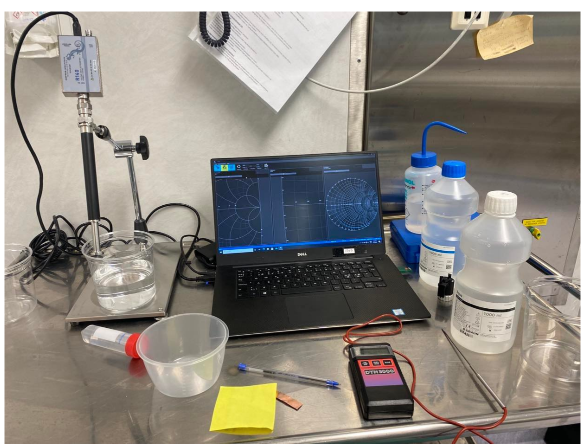

Figure B.4: Full DAK Setup

Bibliography

[1] Kamal Abu-Hassan, Joseph D Taylor, Paul G Morris, Elisa Donati, Zuner A Bortolotto, Giacomo Indiveri, Julian FR Paton, and Alain Nogaret. Optimal solid state neurons. *Nature communications*, 10(1):5309, 2019.

[2] Armin Alaghi, Weikang Qian, and John P Hayes. The promise and challenge of stochastic computing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(8):1515–1531, 2017.

[3] Arnon Amir, Pallab Datta, William P Risk, Andrew S Cassidy, Jeffrey A Kusnitz, Steve K Esser, Alexander Andreopoulos, Theodore M Wong, Myron Flickner, Rodrigo Alvarez-Icaza, et al. Cognitive computing programming paradigm: a corelet language for composing networks of neurosynaptic cores. In *The 2013 International Joint Conference on Neural Networks (IJCNN)*, pages 1–10. IEEE, 2013.

[4] Adarsha Balaji, Prathyusha Adiraju, Hirak J Kashyap, Anup Das, Jeffrey L Krichmar, Nikil D Dutt, and Francky Catthoor. Pycarl: A pynn interface for hardware-software co-simulation of spiking neural network. *arXiv preprint arXiv:2003.09696*, 2020.

[5] Adarsha Balaji, Shihao Song, Anup Das, Jeffrey Krichmar, Nikil Dutt, James Shackleford, Nagarajan Kandasamy, and Francky Catthoor. Enabling resource-aware mapping of spiking neural networks via spatial decomposition. *IEEE Embedded Systems Letters*, 13(3):142–145, 2020.

[6] Ben Varkey Benjamin, Peiran Gao, Emmett McQuinn, Swadesh Choudhary, Anand R Chandrasekaran, Jean-Marie Bussat, Rodrigo Alvarez-Icaza, John V Arthur, Paul A Merolla, and Kwabena Boahen. Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations. *Proceedings of the IEEE*, 102(5):699–716, 2014.

[7] Sander M Bohte, Joost N Kok, and Han La Poutre. Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing*, 48(1-4):17–37, 2002.

[8] Alexander Bryson, Robert John Hatch, Bas-Jan Zandt, Christian Rossert, Samuel F Berkovic, Christopher A Reid, David B Grayden, Sean L Hill, and Steven Petrou. Gaba-mediated tonic inhibition differentially modulates gain in functional subtypes of cortical interneurons. *Proceedings of the National Academy of Sciences*, 117(6):3192–3202, 2020.

[9] Xin-Lu Cai, Gai-Ying Li, Ling-Ling Wang, Wei-Wei Zhao, Yi Wang, Chao Yan, Simon SY Lui, Jian-Qi Li, and Raymond CK Chan. Striatal gaba level is associated with sensory integration ability in individuals with low levels of negative schizotypy. *PsyCh Journal*, 11(2):205–213, 2022.

[10] Andrea Calimera, Enrico Macii, and Massimo Poncino. The human brain project and neuromorphic computing. *Functional neurology*, 28(3):191, 2013.

[11] C Castillo-Botón, D Casillas-Pérez, C Casanova-Mateo, S Ghimire, E Cerro-Prada, PA Gutierrez, RC Deo, and S Salcedo-Sanz. Machine learning regression and classification methods for fog events prediction. *Atmospheric Research*, 272:106157, 2022.

[12] John K Chapin, Karen A Moxon, Ronald S Markowitz, and Miguel AL Nicolelis. Real-time control of a robot arm using simultaneously recorded neurons in the motor cortex. *Nature neuroscience*, 2(7):664–670, 1999.

[13] Hsin-Pai Cheng, Wei Wen, Chunpeng Wu, Sicheng Li, Hai Helen Li, and Yiran Chen. Understanding the design of ibm neurosynaptic system and its tradeoffs: A user perspective. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*, pages 139–144. IEEE, 2017.

[14] Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*, 2016.

[15] Enrico Cherubini et al. Phasic gabaa-mediated inhibition. *Jasper's Basic Mechanisms of the Epilepsies*, pages 127–146, 2012.

[16] André B Cunha, Jie Hou, and Christin Schuelke. Machine learning for stem cell differentiation and proliferation classification on electrical impedance spectroscopy. *Journal of Electrical Bioimpedance*, 10(1):124–132, 2019.

[17] Yang Dan and Mu-ming Poo. Spike timing-dependent plasticity of neural circuits. *Neuron*, 44(1):23–30, 2004.

[18] Anup Das, Yuefeng Wu, Khanh Huynh, Francesco Dell'Anna, Francky Catthoor, and Siebren Schaafsma. Mapping of local and global synapses on spiking neuromorphic hardware. In *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1217–1222. IEEE, 2018.

[19] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, et al. Loihi: A neuromorphic manycore processor with on-chip learning. *Ieee Micro*, 38(1):82–99, 2018.

[20] Anthony S de Leon and Prasanna Tadi. Biochemistry, gamma aminobutyric acid. 2019.

[21] Michael V DeBole, Brian Taba, Arnon Amir, Filipp Akopyan, Alexander Andreopoulos, William P Risk, Jeff Kusnitz, Carlos Ortega Otero, Tapan K Nayak, Rathinakumar Appuswamy, et al. Truenorth: Accelerating from zero to 64 million neurons in 10 years. *Computer*, 52(5):20–29, 2019.

[22] Arash Fayyazi, Mohammad Ansari, Mehdi Kamal, Ali Afzali-Kusha, and Massoud Pedram. An ultra low-power memristive neuromorphic circuit for internet of things smart sensors. *IEEE Internet of Things Journal*, 5(2):1011–1022, 2018.

[23] Sami Gabriel, RW Lau, and Camelia Gabriel. The dielectric properties of biological tissues: Ii. measurements in the frequency range 10 hz to 20 ghz. *Physics in medicine & biology*, 41(11):2251, 1996.

[24] Francesco Galluppi, Xavier Lagorce, Evangelos Stromatias, Michael Pfeiffer, Luis A Plana, Steve B Furber, and Ryad B Benosman. A framework for plasticity implementation on the spinnaker neural architecture. *Frontiers in neuroscience*, 8:429, 2015.

[25] Wulfram Gerstner and Werner M Kistler. *Spiking neuron models: Single neurons, populations, plasticity.* Cambridge university press, 2002.

[26] Samanwoy Ghosh-Dastidar and Hojjat Adeli. Spiking neural networks. *International journal of neural systems*, 19(04):295–308, 2009.

[27] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[29] Jie Hou, Sisay Mebre Abie, Runar Strand-Amundsen, Yuri M Galperin, Joakim Bergli, Christin Schuelke, Sina Hashemizadeh, and Ørjan Grøttem Martinsen. Temperature dependence of the microwave dielectric properties of

\gamma

γ-aminobutyric acid. *Scientific Reports*, 11(1):1–10, 2021.

[30] H Jabbar and Rafiqul Zaman Khan. Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study). *Computer Science, Communication and Instrumentation Devices*, 70:163–172, 2015.

[31] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2704–2713, 2018.

[32] Daniel Justus, John Brennan, Stephen Bonner, and Andrew Stephen McGough. Predicting the computational cost of deep learning models. In *2018 IEEE international conference on big data (Big Data)*, pages 3873–3882. IEEE, 2018.

[33] Zenun Kastrati, Ali Shariq Imran, and Sule Yildirim Yayilgan. The impact of deep learning on document classification using semantically rich representations. *Information Processing & Management*, 56(5):1618–1632, 2019.

[34] Yoo Sung Kim and Bo-Eun Yoon. Altered gabaergic signaling in brain disease at various stages of life. *Experimental neurobiology*, 26(3):122, 2017.

[35] Alessandra La Gioia, Emily Porter, Ilja Merunka, Atif Shahzad, Saqib Salahuddin, Marggie Jones, and Martin O'Halloran. Open-ended coaxial probe technique for dielectric measurement of biological tissues: Challenges and common practices. *Diagnostics*, 8(2):40, 2018.

[36] Mikhail Lebedev. Brain-machine interfaces: an overview. *Translational Neuroscience*, 5(1):99–110, 2014.

[37] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[38] Andrew Lines, Prasad Joshi, Ruokun Liu, Steve McCoy, Jonathan Tse, Yi-Hsin Weng, and Mike Davies. Loihi asynchronous neuromorphic research chip. In *2018 24th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*, pages 32–33. IEEE, 2018.

[39] Shih-Chii Liu, Tobi Delbruck, Giacomo Indiveri, Adrian Whatley, and Rodney Douglas. *Event-based neuromorphic systems*. John Wiley & Sons, 2014.

[40] R Bruce Lydiard. The role of gaba in anxiety disorders. *Journal of Clinical Psychiatry*, 64:21–27, 2003.

[41] Batta Mahesh. Machine learning algorithms-a review. *International Journal of Science and Research (IJSR).[Internet]*, 9:381–386, 2020.

[42] James R Matthie. Bioimpedance measurements of human body composition: critical analysis and outlook. *Expert review of medical devices*, 5(2):239–261, 2008.

[43] Alan D Moore. *Python GUI Programming with Tkinter: Develop responsive and powerful GUI applications with Tkinter*. Packt Publishing Ltd, 2018.

[44] Simon W Moore, Paul J Fox, Steven JT Marsh, A Theodore Markettos, and Alan Mujumdar. Bluehive-a field-programable custom computing machine for extreme-scale real-time neural network simulation. In *2012 IEEE 20th International Symposium on Field-Programmable Custom Computing Machines*, pages 133–140. IEEE, 2012.

[45] K Muralidharan. A note on transformation, standardization and normalization. *IUP J. Oper. Manag*, 9:116–122, 2010.

[46] Sergei Petrenko and Sergei Petrenko. Limitations of von neumann architecture. *Big Data Technologies for Monitoring of Computer Security: A Case Study of the Russian Federation*, pages 115–173, 2018.

[47] Mohamad Zaim Awang Pon and Krishna Prakash KK. Hyperparameter tuning of deep learning models in keras. *Sparklinglight Transactions on Artificial Intelligence and Quantum Computing (STAIQC)*, 1(1):36–40, 2021.

[48] Alberto Prieto, Beatriz Prieto, Eva Martinez Ortigosa, Eduardo Ros, Francisco Pelayo, Julio Ortega, and Ignacio Rojas. Neural networks: An overview of early research, current frameworks and new challenges. *Neurocomputing*, 214:242–268, 2016.

[49] Peter Roberts, Mason Zadan, and Carmel Majidi. Soft tactile sensing skins for robotics. *Current Robotics Reports*, 2:343–354, 2021.

[50] Bihi Sabiri, Bouchra El Asri, and Maryem Rhanoui. Mechanism of overfitting avoidance techniques for training deep neural networks. In *ICEIS (1)*, pages 418–427, 2022.

[51] Catherine D Schuman, Thomas E Potok, Robert M Patton, J Douglas Birdwell, Mark E Dean, Garrett S Rose, and James S Plank. A survey of neuromorphic computing and neural networks in hardware. *arXiv preprint arXiv:1705.06963*, 2017.

[52] Thomas Sharp, Luis A Plana, Francesco Galluppi, and Steve Furber. Event-driven simulation of arbitrary spiking neural networks on spinnaker. In *Neural Information Processing: 18th International Conference, ICONIP 2011, Shanghai, China, November 13-17, 2011, Proceedings, Part III 18*, pages 424–430. Springer, 2011.

[53] Geoff Smith, Alistair P Duffy, Jie Shen, and Cedric J Olliff. Dielectric relaxation spectroscopy and some applications in the pharmaceutical sciences. *Journal of pharmaceutical sciences*, 84(9):1029–1044, 1995.

[54] Shihao Song, Adarsha Balaji, Anup Das, Nagarajan Kandasamy, and James Shackleford. Compiling spiking neural networks to neuromorphic hardware. In *The 21st ACM SIGPLAN/SIGBED Conference on Languages, Compilers, and Tools for Embedded Systems*, pages 38–50, 2020.

[55] Suraj Srinivas, Ravi K Sarvadevabhatla, Konda R Mopuri, Nikita Prabhu, Srinivas SS Kruthiventi, and R Venkatesh Babu. An introduction to deep convolutional neural nets for computer vision. In *Deep learning for medical image analysis*, pages 25–52. Elsevier, 2017.

[56] Narayan Srinivasa and Jose M Cruz-Albrecht. Neuromorphic adaptive plastic scalable electronics: analog learning systems. *IEEE pulse*, 3(1):51–56, 2012.

[57] Daniel Svozil, Vladimir Kvasnicka, and Jiri Pospichal. Introduction to multi-layer feed-forward neural networks. *Chemometrics and intelligent laboratory systems*, 39(1):43–62, 1997.

[58] Zied Tayeb, Emeç Erçelik, and Jörg Conradt. Decoding of motor imagery movements from eeg signals using spinnaker neuromorphic hardware. In *2017 8th International IEEE/EMBS Conference on Neural Engineering (NER)*, pages 263–266. IEEE, 2017.

[59] C Guin Ting Wong, Teodoro Bottiglieri, and O Carter Snead III. Gaba, γ-hydroxybutyric acid, and neurological disease. *Annals of Neurology: Official Journal of the American Neurological Association and the Child Neurology Society*, 54(S6):S3–S12, 2003.

[60] Anup Vanarse, Adam Osseiran, Alexander Rassau, and Peter van der Made. A hardware-deployable neuromorphic solution for encoding and classification of electronic nose data. *Sensors*, 19(22):4831, 2019.

[61] Stefano Vassanelli and Mufti Mahmud. Trends and challenges in neuroengineering: toward "intelligent" neuroprostheses through brain-"brain inspired systems" communication. *Frontiers in neuroscience*, 10:438, 2016.

[62] OA Vedyasova and TE Kovaleva. Role of gaba a receptors of parafacial respiratory group in control of respiration in rats. *Bulletin of Experimental Biology and Medicine*, 165:711–714, 2018.

[63] Bharadwaj Veeravalli, Xuanyao Fong, et al. An fpga-based co-processor for spiking neural networks with on-chip stdp-based learning. In *2022 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 2157–2161. IEEE, 2022.

[64] Gen Yan, Tao Zhang, Zhuozhi Dai, Meizhi Yi, Yanlong Jia, Tingting Nie, Handi Zhang, Gang Xiao, and Renhua Wu. A potential magnetic resonance imaging technique based on chemical exchange saturation transfer for in vivo γ-aminobutyric acid imaging. *PloS one*, 11(10):e0163765, 2016.

[65] Qiang Yu, Huajin Tang, Jun Hu, and Kay Chen Tan. Neuromorphic cognitive systems. *A Learning and Memory Centered Approach*, 2017.

[66] Chao Zhang and Gary C Stevens. The dielectric response of polar and non-polar nanodielectrics. *IEEE Transactions on Dielectrics and Electrical Insulation*, 15(2):606–617, 2008.

[67] Xiandong Zhao, Ying Wang, Xuyi Cai, Cheng Liu, and Lei Zhang. Linear symmetric quantization of neural networks for low-precision integer hardware. 2020.