Analog spike-based neuromorphic computing for low-power smart IoT applications

Zalfa Jouni

université PARIS-SACLAY

Analog Spike-Based Neuromorphic Computing for Low-Power Smart IoT Applications

Calcul Neuromorphique à Base de Spikes Analogiques pour des Applications IoT Intelligentes à Faible Consommation

Thèse de doctorat de l'université Paris-Saclay

École doctorale n° 575, Electrical, optical, bio-physics and engineering (EOBE)
Spécialité de doctorat : Électronique et optoélectronique, nano et microtechnologies
Graduate School : Sciences de l'ingénierie et des systèmes.
Référent : CentraleSupélec

Thèse préparée dans la unité de recherche **Laboratoire de Génie Electrique et Electronique de Paris** (Université Paris-Saclay, CentraleSupélec, CNRS), sous la direction de **Pietro MARIS FERREIRA**, Maître de Conférence (HDR), et le co-encadrement de **Siqi WANG**, Maître de Conférence

Thèse soutenue à Paris-Saclay, le 9 octobre 2024, par

Zalfa JOUNI

Composition du jury

Membres du jury avec voix délibérative

Damien QUERLIOZ Directeur de recherche, CNRS, Université Paris-Saclay	Président & Examinateur
Florence PODEVIN Professeur des universités, TIMA, Université Grenoble Alpes	Rapporteur & Examinatrice
Sylvain SAIGHI Professeur des universités, IMS, Université de Bordeaux	Rapporteur & Examinateur
Haralampos STRATIGOPOULOS Directeur de recherche, LIP6, CNRS, Sorbonne Université	Examinateur
Patricia DESGREYS Professeur des universités, LTCI, Télécom Paris Tech	Examinatrice

THESE DE DOCTORAT
NNT : 2024UPAST114

université PARIS-SACLAY

ÉCOLE DOCTORALE
Physique et ingénierie : Electrons, Photons, Sciences du vivant (EOBE)

Title: Analog Spike-Based Neuromorphic Computing for Low-Power Smart IoT Applications
Keywords: Edge IoT, Energy-Efficient Localization, Neuromorphic Computing, Spiking Neural Network, Analog Design

Abstract: As the Internet of Things (IoT) expands with more connected devices and complex communications, the demand for precise, energy-efficient localization technologies has intensified. Traditional machine learning and artificial intelligence (AI) techniques provide high accuracy in radio-frequency (RF) localization but often at the cost of greater complexity and power usage. To address these challenges, this thesis explores the potential of neuromorphic computing, inspired by brain functionality, to enable energy-efficient AI-based RF localization. It introduces an end-to-end analog spike-based neuromorphic system (RF NeuroAS), with a simplified version fully implemented in BiCMOS 55 nm technology. RF NeuroAS is designed to identify source positions within a 360-degree range on a two-dimensional plane, maintaining high resolution (10 or 1 degree) even in noisy conditions. The core of this system, an analog-based spiking neural network (A-SNN), was trained and tested on a simulated dataset (SimLocRF) from MATLAB and an experimental dataset (MeasLocRF) from anechoic chamber measurements, both developed in this thesis. The learning algorithms for A-SNN were developed through two approaches: software-based deep learning (DL) and bio-plausible spike-timing-dependent plasticity (STDP). RF NeuroAS achieves a localization accuracy of 97.1% with SimLocRF and 90.7% with MeasLoc at a 10-degree resolution, maintaining high performance with low power consumption in the nanowatt range. The simplified RF NeuroAS consumes just over 1.1 nW and operates within a 30 dB dynamic range. A-SNN learning, via DL and STDP, demonstrated performance on XOR and MNIST problems. DL depends on the non-linearity of post-layout transfer functions of A-SNN's neurons and synapses, while STDP depends on the random noise in analog neuron circuits. These findings highlight advancements in energy-efficient IoT through neuromorphic computing, promising low-power smart edge IoT breakthroughs inspired by brain mechanisms.

université PARIS-SACLAY

ÉCOLE DOCTORALE
Physique et ingénierie : Electrons, Photons, Sciences du vivant (EOBE)

Titre: Calcul Neuromorphique à Base de Spikes Analogiques pour des Applications IoT Intelligentes à Faible Consommation

Mots clés: IoT en périphérie, Localisation écoénergétique, Informatique neuromorphique, Réseau de neurones à impulsions, Conception analogique

Résumé: Avec l'expansion de l'Internet des objets (IoT) et l'augmentation des appareils connectés et des communications complexes, la demande de technologies de localisation précises et économes en énergie s'est intensifiée. Les techniques traditionnelles de machine learning et d'intelligence artificielle (IA) offrent une haute précision dans la localisation par radiofréquence (RF), mais au prix d'une complexité accrue et d'une consommation d'énergie élevée. Pour relever ces défis, cette thèse explore le potentiel de l'informatique neuromorphique, inspirée par les mécanismes du cerveau, pour permettre une localisation RF basée sur l'IA et économe en énergie. Elle présente un système neuromorphique analogique à base d'impulsions (RF NeuroAS), avec une version simplifiée entièrement implémentée en technologie BiCMOS 55 nm. Ce système identifie les positions des sources dans une plage de 360 degrés sur un plan bidimensionnel, en maintenant une haute résolution (10 ou 1 degré) même dans des conditions bruyantes. Le cœur de ce système, un réseau de neurones à impulsions basé sur l'analogique (A-SNN), a été formé et testé sur des données simulées (SimLocRF) à partir de MATLAB et des données expérimentales (MeasLocRF) provenant de mesures en chambre anéchoïque, tous deux développés dans cette thèse. Les algorithmes d'apprentissage pour l'A-SNN ont été développés selon deux approches: l'apprentissage profond (DL) et la plasticité dépendante du temps des impulsions (STDP) bio-plausible. RF NeuroAS atteint une précision de localisation de 97,1% avec SimLocRF et de 90,7% avec MeasLoc à une résolution de 10 degrés, tout en maintenant une haute performance avec une faible consommation d'énergie de l'ordre du nanowatt. Le RF NeuroAS simplifié consomme seulement 1.1 nW et fonctionne dans une plage dynamique de 30 dB. L'apprentissage de l'A-SNN, via DL et STDP, a démontré des performances sur les problèmes XOR et MNIST. Le DL dépend de la non-linéarité des fonctions de transfert post-layout des neurones et des synapses de l'A-SNN, tandis que le STDP dépend du bruit aléatoire dans les circuits neuronaux analogiques. Ces résultats marquent des avancées dans les applications IoT économes en énergie grâce à l'informatique neuromorphique, promettant des percées dans l'IoT intelligent à faible consommation d'énergie inspirées par les mécanismes du cerveau.

Pour maman, étoile des maths, avec un rêve si grand, stoppé dans son élan, à elle, je dédie ma thèse.

⋆ ⋆ ⋆

To my mother, the star of mathematics, with a dream so big, cut short in its path. To her, I dedicate this thesis.

Acknowledgement

This may be the easiest part of my thesis to read, yet it is undoubtedly the hardest to write. How can I express the immense gratitude I feel toward everyone who has been an integral part of this journey in just a few lines?

First and foremost, I would like to thank Pietro Maris Ferreira, my thesis director. While this title fits you perfectly, you have been so much more. From my very first day to the final moments, you have been my steadfast companion. You believed in my potential, guided me through my work, and offered invaluable advice on every challenge, big and small. You nurtured not only the researcher in me but also the educator. I would also like to extend my heartfelt thanks to Siqi Wang, my thesis co-supervisor. You were always there, even across distances, providing support and insightful discussions—particularly when it came to mathematics.

My sincere appreciation goes to my thesis reviewers, Sylvain Saighi and Florence Podevin, for accepting such a pivotal role in this process. I am deeply grateful to all the members of the jury for honoring me with their presence at my defense. Your time and thoughtful feedback have had a lasting impact.

Special thanks go to Aziz Benlarbi-Delai, whose scientific greetings always brightened my days, and to the entire GeePs laboratory team—from the director to the researchers and technicians. Your support in providing both scientific and technical resources has been fundamental to my work. In particular, I want to thank Philippe Bénabès, Philippe Ramos, Jérôme Juillard, and Caroline Lelandais-Perrault for their invaluable assistance.

To my family—my parents and my four brothers, Hassan, Elias, Ali, and Houssein—your unwavering support and faith in me have been the foundation of my strength and perseverance. Every step of this journey was made possible because of the love and belief you gave me. To my father Mohamad, your constant faith in my abilities and your quiet strength have always been my anchor. Thank you for being my pillar of support in everything I do. To my mother Maryam, the sight of your eyes during our daily video calls gave me the courage and determination to see this thesis through. Your presence, even from afar, was my guiding light.

A special thanks to my partner, Abbas Yassine. Abbas, you deserve a chapter of your own in this story. Your incredible companionship, understanding, patience in listening, reviewing my work, sometimes teasing me, but always loving me, has made all the difference.

I must also express my deepest thanks to my dear friends. Sirine, you have been like a sister, standing by me through the toughest moments. I am so grateful for your unwavering love. Zulal, you were my comfort, whether through conversations or hugs. Hiba, you shared the bittersweetness of being far from home—your presence made every morning at the university special. Safaa, true to your name, your calm and serenity were essential to my journey. And to my wonderful friends—Mirna, Amar, Lara, Maryam, Soha, Maria, Sandy—along with many others whose names I may not have mentioned, know that your presence in my life is cherished beyond words.

I would also like to thank the brilliant students I had the pleasure of working with: Joao, Thomas, Théo, and Yannael. Your dedication and enthusiasm were a constant source of inspiration, and it has been a privilege to work with such bright minds.

I want to extend my gratitude to those who inspired me throughout this journey, even if they may not realize it: my uncle Houssein Akhdar and my professor at the Lebanese University, Faculty of Engineering, Ibrahim Ismail. Finally, to everyone who has contributed to this journey—whether mentioned here or not—thank you from the bottom of my heart. In writing these acknowledgments, I am reminded that the journey of a PhD is never a solitary endeavor but a collective voyage enriched by each of you. Thank you for being part of my story.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 . Context and Motivation

Imagine a world where machines communicate seamlessly, factories operate independently, and intelligent systems anticipate our needs before we even express them. Imagine vehicles that autonomously navigate bustling city streets, and robotic arms perform complex surgeries with unparalleled precision. Imagine a watch that does more than just count minutes, it counts your steps, monitors your health, handles your payments, and keeps you updated with notifications. Imagine a virtual assistant so intuitive it feels human, one that handles your daily tasks and engages in any conversation you need with deep understanding. This vision, which once seemed like science fiction, is now becoming a reality thanks to the monumental advances of the 21st century in the fourth industrial revolution, Industry 4.0, fueled by the Internet of Things (IoT) and Artificial Intelligence (AI).

From the spark of the first electric light by Thomas Edison in 1879 to the transistor's invention by Bardeen, Brattain, and Shockley in 1947, the trajectory of human advancement has been marked by significant electrical innovations. Alan Turing's theoretical formulation of the digital computer during WWII in the 1940s laid the foundation for modern computing. This was revolutionized by John von Neumann's architecture in 1945, which reshaped computer design and established its role as a fundamental component of contemporary technology. The development of the Internet in the 1960s, followed by Tim Berners-Lee's creation of the World Wide Web in 1989, revolutionized global communication and information sharing. Despite these advancements, devices remained constrained by their inability to communicate autonomously, heavily relying on human intervention for control and data interpretation. However, the advent of IoT in the late 20th century marked a significant paradigm shift. IoT enabled devices to interconnect, facilitating autonomous communication and data sharing without human intervention [1]. This interconnectivity has reshaped daily life by enhancing efficiency, and functionality across various domains, from smart homes and wearables to industrial automation and healthcare [2].

Building on advancements in AI and cloud computing, cloud-based AI enables centralized data processing within the IoT framework. This integration significantly enhances the IoT landscape and broadens its applications, as everyday objects become increasingly connected to the Internet, leading to a proliferation of devices and sensors [3]. However, this rapid expansion places substantial pressure on cloud computing resources due to the enormous volumes of data processed [4]. This situation underscores the limitations of centralized cloud systems, particularly concerning latency and bandwidth constraints. To keep pushing the boundaries and support the rapid expansion of IoT, edge AI emerges as a solution by bringing AI processing closer to the data source, rather than being relayed to the cloud [5, 6]. This approach reduces latency and bandwidth demands, lowers energy requirements for data transmission, and enhances privacy

and security within the interconnected world.

In the context of Industry 4.0, the current trend in AI is centered on Deep Neural Networks (DNNs) to address complex problems with high performance, extending AI's application across various domains [7]. These AI models comprise networks with multiple densely connected layers, using deep learning techniques, primarily the backpropagation algorithm, which requires extensive parameter tuning. Training these advanced AI models necessitates large datasets and significant computational resources, typically on classical computing systems based on the von Neumann architecture. As AI capabilities expand, the associated challenges also increase. The computational demands of AI are growing at a rate that outpaces Moore's Law, with capabilities approximately doubling every five to six months [8]. This rapid growth results in high power consumption, inefficiencies of the von Neumann architecture, and extensive storage requirements for large data centers [9, 10].

To address these challenges, the AI field is shifting towards more energy-efficient solutions. A notable development is edge AI, which enhances power efficiency by processing data directly at the edge, reducing the need for frequent data transfers. However, edge AI requires hardware beyond the capabilities of traditional front-end sensing equipment, currently limited by low-capacity computing resources. This shift necessitates the development of specialized hardware, efficient algorithms for resource-constrained devices, and innovative neural network designs leveraging the low power characteristics of edge devices [11]. These goals can be achieved through neuromorphic computing, which draws inspiration from the brain, the most intricate and powerful system known, to develop more energy-efficient AI solutions.

Neuromorphic computing has emerged as a promising approach to manage the computational demands of AI and deep learning, as well as meeting the energy efficiency needs of IoT applications. By emulating the human brain's efficient, adaptive, and rapid processing capabilities, neuromorphic computing is prioritized for several reasons. Neuromorphic systems achieve high energy efficiency and reduced latency by performing computations in memory, diverging from conventional von Neumann architectures [12]. These systems excel in parallel processing and real-time operations by simulating biological neural networks through event-driven computation [13]. Specifically, they rely on Spiking Neural Networks (SNNs), which activate neurons only when necessary, thus optimizing efficiency.

In recent years, hardware implementations of neuromorphic computing have advanced significantly, with developments spanning digital, analog, and mixed-signal platforms using both traditional CMOS and emerging technologies. Most common implementations have focused on replicating human sensory functions [14, 15], such as vision and auditory recognition, thus diverging from the specific needs of IoT applications. IoT devices operate differently in their sensor functions, including device localization, pressure, and temperature detection. This gap underscores the critical need for neuromorphic systems specifically designed to address the requirements of IoT applications. Moreover, few neuromorphic implementations are designed as complete end-to-end systems, often lacking integral pre-processing procedures and requiring digitization. For these reasons, including AI challenges and neuromorphic promising solutions, this thesis is motivated to explore the potential of neuromorphic computing to meet the energy efficiency demands of IoT applications, with a particular focus on Radio Frequency (RF) localization.

1.2 . Thesis Contributions

This thesis has a multidisciplinary contribution, connecting microelectronics, electromagnetism, and artificial intelligence. It focuses on addressing the need for low-power smart IoT applications through an analog spike-based neuromorphic approach. The primary application targeted by this thesis is the RF localization of an IoT transmitter with precision comparable to existing AI-based solutions, while being significantly more energy-efficient and environmentally friendly. This is a critical issue, as the demand for energy-efficient and precise solutions in smart IoT intensifies. The main contributions of this thesis can be summarized as follows:

- **Design of an end-to-end RF neuromorphic system (RF NeuroAS)**: This system uses an analog spike-based approach to emulate brain-like capabilities in addressing RF localization challenges. It identifies IoT transmitter positions within a 360-degree range on a two-dimensional plane, maintaining high resolution (10 or 1 degree) even in noisy conditions. This work includes the RF environment setup, data extraction involving the generation of datasets, pre-processing, and analog-based spiking neural network design. Scientific communications, highlighting this contribution, are available in [16] and [17].
- **Implementation of a fully analog circuit design for a streamlined version of the RF NeuroAS system**, using BiCMOS 55 nm technology. This validates the feasibility of hardware-based RF NeuroAS solutions for localization, proving ultra-low power consumption in post-layout simulations. Scientific communications, highlighting this contribution, are available in [18], [19], and [20].
- **Feasibility study of learning techniques on analog-based spiking neural networks**, considering the physical design constraints, the random noise, and the process variability of neuron and synapse components. The learning techniques explored include deep learning and spike-timing-dependent plasticity. Scientific communications, highlighting this contribution, are available in [21], [22], [23],[24], [25], and [26].

The following manuscript is organized as follows. Chapter 2 provides an overview of the development and evolution of AI, deep learning, and neuromorphic computing in the literature, and outlines AI-based research efforts in the field of localization. Chapter 3 thoroughly presents the RF NeuroAS system and each of its components. Chapter 4 details the feasibility analysis of learning techniques from a theoretical perspective. Chapter 5 presents the detailed results derived in this thesis, corresponding to the performance of the RF NeuroAS and the feasibility results. Finally, thesis conclusions and perspective challenges of future works are drawn.

1.3 . List of Publications

During my PhD, I have sent 6 journal papers for publication (4 are still under review), 7 papers to international conferences with proceedings, 4 papers to national conferences or workshops without proceedings, and 2 open source papers.

Journal Articles

[J1] **Jouni, Z.**, Wang, S., Benlarbi-Delai, A., Ferreira, P. M., "Energy-Efficient Neuromorphic RF Localization using Analog Spiking Neurons", *IEEE Transactions on Circuits and Systems for Artificial Intelligence*, Status: **Submitted**, 2024.

[J2] Ferreira, P. M., **Jouni, Z.**, Wang, S., Klisnick, G., Benlarbi-Delai, A., "Neuromorphic-Enhanced Wake-up Radio for a Context-Aware IoT", *IEEE Design & Test*, Status: **Submitted**, 2024.

[J3] Bossard, Y., **Jouni, Z.**, Wang, S., Ferreira, P. M., "Analog Spiking Neuron Model for Unsupervised STDP-based learning in Neuromorphic Circuits", *Journal of Integrated Circuits and Systems*, Status: **Submitted**, 2024.

[J4] Prats, T., Ramos, P., **Jouni, Z.**, Wang, S., Benlarbi-Delai, A., Ferreira, P. M., "Energy-Efficient and Robust-Trained Analog Neural Network Synthesis: a Design Framework for Multilayer Perceptron Problem", *IEEE Open Journal of Circuits and Systems*, Status: **Submitted**, 2024.

[J5] **Jouni, Z.**, Soupizet, T., Wang, S., Benlarbi-Delai, A., Ferreira, P. M., "RF Neuromorphic Spiking Sensor for IoT Devices", *Analog Integrated Circuits and Signal Processing*", vol. 117, pp 3-20, 2023, doi: 10.1007/s10470-023-02164-w

[J6] Soupizet, T., **Jouni, Z.**, Wang, S., Benlarbi-Delai, A., Ferreira, P. M.," Analog Neural Network Synthesis for the MNIST", *Journal of Integrated Circuits and Systems*", vol. 18, n. 1, pp 1-12, 2023, doi: 10.29292/jics.v18i1.663

International Conferences Papers

[C1] **Jouni, Z.**, Wang, S., Benlarbi-Delai, A., Ferreira, P. M.,"Analog Spiking-Rate Neuromorphic System for Efficient RF Source Localization", *IEEE 31th Conference on Circuits and Systems (ICECS)*, Status: **Accepted**, 2024.

[C2] **Jouni, Z.**, Prats, T., Wang, S., Benlarbi-Delai, A., Ferreira, P. M., "Jitter Noise Impact on Analog Spiking Neural Networks: STDP Limitations", *IEEEE 36th SBC Symposium on Integrated Circuits and Systems Design (SBCCI)*, Rio de Janeiro, Brazil, 2023, pp. 1-6, doi: 10.1109/SBCCI60457.2023.10261661

[C3] Prats, T., **Jouni, Z.**, Wang, S., Benlarbi-Delai, A., Ferreira, P. M., "Revisiting the Ultra-Low Power Electronic Neuron Towards a Faithful Biomimetic Behavior", *IEEE 36th SBC Symposium on Integrated Circuits and Systems Design (SBCCI)*, Rio de Janeiro, Brazil, 2023, pp. 1-6, doi: 10.1109/SBCCI60457.2023.10261961

[C4] Sulzbach, J. F., Wang, S., **Jouni, Z.**, Benlarbi-Delai, A., Klisnick, G., Ferreira, P. M., "Sub-nJ per Decision Schmitt Trigger Comparator for Neuromorphic Spike Detection in 55 nm Technology", *IEEE 7th Forum on Research and Technologies for Society and Industry Innovation (RTSI)*, Paris, France, 2022, pp. 1-6, doi: 10.1109/rtsi55261.2022.9905228

[C5] **Jouni, Z.**, Soupizet, T., Wang, S., Benlarbi-Delai, A., Ferreira, P. M., "1.2 nW Neuromorphic Enhanced Wake-Up Radio", *IEEE 35th SBC Symposium on Integrated Circuits and Systems Design (SBCCI)*, Porto Alegre, Brazil, 2022, pp. 1-6, doi: 10.1109/SBCCI55532.2022.9893247

[C6] Soupizet, T., **Jouni, Z.**, Sulzbach, J. F., Benlarbi-Delai, A., Ferreira, P. M., "Deep Neural Network Feasibility Using Analog Spiking Neurons", *IEEE 35th SBC Symposium on Integrated Circuits and Systems Design (SBCCI)*, Porto Alegre, Brazil, 2022, pp. 1-6, doi: 10.1109/SBCCI55532.2022.9893216

[C7] Juillard, J., **Jouni, Z.**, Bourgois, L., et al., "MEMS Resonator Parameter Estimation from Fast Frequency Sweeps", *IEEE International Frequency Control Symposium*, Paris, France, 2022, pp. 1-5, doi: 10.1109/EFTF/IFCS54560.2022.9850663

National Conferences and Workshops

[W1] **Jouni, Z.**, Ferreira, P. M., "An Analog Spike-Based Neuromorphic Sensor for Efficient RF Localization", *IEEE CASS-France Workshop on Hardware Solutions for AI*, Paris, France, 2024, *Oral Presentation.*

[W2] **Jouni, Z.**, Ferreira, P. M., "Système Neuromorphique pour un radio-réveil à 1.2 nW", *IEEE CAS-France Workshop on Machine Learning and Artificial Intelligence: from circuit to system level*, Bordeaux, France, 2022, *Poster Presentation.*

[NC1] **Jouni, Z.**, Wang, S., Ferreira, P. M., "RF Neuromorphic Spiking Sensor for Smart IoT Devices", *17ème Colloque National du GDR SoC2*, Strasbourg, France, 2022, *Poster Presentation.*

[NC2] **Jouni, Z.**, Wang, S., Ferreira, P. M., "1.2 nW Neuromorphic Enhanced Wake-Up Radio", *16ème Colloque National du GDR SoC2*, Strasbourg, France, 2022, *Poster Presentation.*

Open Source Papers and Libraries

[S1] **Jouni, Z.**, Wang, S., and Ferreira, P. M. , "Analog Neuromorphic System for Low-Power RF Localization in IoT," Open Softw., 2024, doi: 10.5281/zenodo.12808148.

[S2] Bossard, Y., **Jouni, Z.**, Wang, S., and Ferreira, P. M. , "Analog Spiking Neuron Model for Unsupervised STDP-based Learning in Neuromorphic Circuits," Open Softw., 2024, doi: 10.5281/zenodo.12839336.

Chapter 2

State of the Art

2.1 . The Need of Localization for Internet of Things

The Internet of Things (IoT) marks a revolutionary shift, redefining traditional lifestyles with advanced technology [2]. Originally introduced by Kevin Ashton, the term *Internet of Things* was meant to increase the utility of radio frequency identification into broader connectivity through the Internet [1]. However, IoT has evolved to extend beyond its initial scope, linking the Internet with the physical world through an expansive network of connected devices and sensors. As of now, billions of these devices populate the globe, with forecasts from IoT Analytics predicting that by 2027, the number of IoT connections will exceed 29 billion [3]. This expansive growth supports a wide array of applications, impacting various sectors such as healthcare, transportation, wearables, home automation, industrial operations, and agriculture.

> *"The problem is people have limited time, attention, and accuracy. And that's a big deal [...] The Internet of Things has the potential to change the world, just as the Internet did. Maybe even more so."*
>
> — *Kevin Ashton*

In the ever-evolving realm of IoT, localization and tracking have become essential, finding widespread use in sectors like sonar, radar, seismic, mobile communications, and Wireless Sensor Networks (WSN) [30]. Furthermore, the demand for precise localization is becoming increasingly critical to boost efficiency, security, and functionality across various industries. In security applications, accurate localization is essential for identifying the source of unauthorized or rogue signals, thereby protecting system integrity [31]. In urban planning, precise localization facilitates the seamless coordination of extensive sensor and actuator networks, ensuring efficient operation. Additionally, in the healthcare sector, precise localization plays a key role in effective resource management, shortening response times, and improving patient care [32], [33].

As the number of IoT devices increases, they operate in environments crowded with overlapping signals. Effective source localization simplifies this complexity by reducing signal interference and boosting the reliability of wireless communications. Moreover, localization allows devices to determine their geographical position from their signals, which is essential for optimizing network logistics and managing resources efficiently [34]. These enhancements not only improve service quality but also extend device lifespan by reducing energy consumption through efficient signal processing.

Localization within WSN-based IoT applications is a thoroughly researched topic, with various approaches and measurement techniques proposed in the literature. These methods span multiple technologies, and their classifications are detailed in Fig. 2.1.

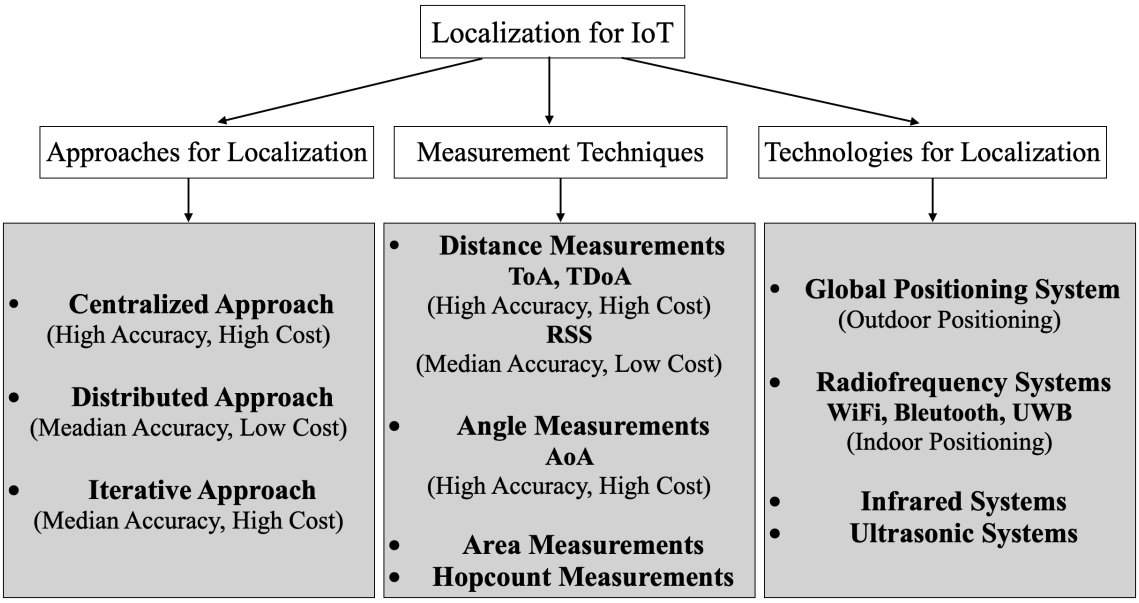


Figure 2.1: Overview of localization strategies in IoT: This diagram categorizes the different approaches, measurement techniques, and technologies used for localization [30].

2.1.1 . Approaches for localization

As illustrated in Fig. 2.1, three principal approaches to localization are used: centralized, distributive, and iterative.

Centralized Localization means that IoT devices, such as sensors and nodes, collect environmental data and transmit it to a central server for processing. Centralized localization takes advantage of cloud computing platforms [4], which offer powerful processing capabilities to manage complex algorithms and large datasets from many devices. While this approach enhances processing capabilities and analytical depth, it also introduces significant challenges. These include increased network traffic, higher latency from extensive data transmission, and rising costs as the network expands, all of which can be detrimental in time-sensitive applications.

Distributed Localization means that devices communicate directly with their neighboring devices to exchange information and determine their positions without relying on a central server. The primary advantage of distributed localization is its efficiency in environments with limited connectivity to a central server, reducing network congestion and speeding up decision-making processes. This method enables data processing directly at the edge, closely aligning with edge computing [4]. This alignment addresses the limitations of cloud computing in scenarios that require real-time data processing and low-latency applications.

Iterative Localization means that the location estimation is refined through repeated computations, enhancing accuracy with continuous feedback and corrections. This method is precise, adaptable, and aligns well with both cloud and edge computing. Cloud environments manage the heavy computational loads, while edge computing supports real-time, low-latency processing. However, its complexity and high processing demands pose challenges in practical

implementation [30].

2.1.2 . Measurement Techniques

As shown in Fig. 2.1, localization relies on various measurement techniques, broadly classified into four types: angle measurement, distance measurement, area measurement, and hopcount measurement. Here are some examples of interest:

Angle of Arrival (AoA) technique falls under angle measurement and is based on the triangulation method [34]. It uses arrays of antennas to precisely determine the direction from which a signal is received. While AoA offers high precision, it often requires regular calibration of the antenna arrays, leading to higher costs.

Time of Arrival (ToA) and Time difference of arrival (TDoA) techniques, categorized under distance measurement, rely on the timing of signal arrivals to determine location [30]. ToA calculates the travel time of a signal from a transmitter to a receiver, measuring distance based on the signal's known speed. TDoA compares the arrival times of the same signal at different receivers to pinpoint the signal source's location. Both methods offer high accuracy under ideal conditions but necessitate precise synchronization between transmitters and receivers, making them complex and potentially costly.

Received Signal Strength (RSS), classified under distance measurement, estimates a device's location based on the power levels of received signals [35]. By measuring signal strength, which decreases with distance, RSS can infer how far away a source is from the receiver. While RSS is sensitive to interference, it is a straightforward and cost-effective method that does not require complex hardware or high computational resources.

2.1.3 . Technologies of localization

Several technologies have been developed for localization in various IoT applications, whether in indoor or outdoor environments. Here are the prominent ones:

Global Positioning System (GPS) is the most recognized satellite navigation system, providing accurate location and time information worldwide when there is an unobstructed line of sight to its satellites. It is extensively used in IoT applications such as vehicle telematics, smartphones, personal wearables, and emergency response systems [31]. However, GPS signals are significantly weakened or blocked by roofs, walls, and other structures, making it ineffective for indoor environments [30].

Radiofrequency (RF) Systems, including Wi-Fi, Bluetooth, and Ultra-Wideband (UWB), are essential for indoor localization where GPS signals are weak [31]. These technologies operate on different parts of the RF spectrum. Wi-Fi typically uses 2.4 GHz and 5 GHz bands, while Bluetooth operates in the 2.4 GHz band. Wi-Fi and Bluetooth can be used for localization through techniques such as RSS triangulation and fingerprinting, which measure the signal strength from multiple access points to estimate a device's location. UWB spans a much wider spectrum, typically between 3.1 and 10.6 GHz, allowing for precise location tracking with minimal interference. They are cost-effective and ubiquitous, offering easy integration, but are susceptible to interference from physical obstructions and electronic devices.

Infrared (IR) systems use IR light to detect objects and determine their location by measuring the reflection of IR beams from a known emitter. IR technology is commonly used in

controlled indoor environments such as smart homes and industrial automation systems where precise short-range detection is needed [36]. IR systems are limited by the requirement for a line-of-sight between the emitter and receiver, which can be easily obstructed.

Ultrasonic systems use sound waves at frequencies above the human hearing range to measure distances and map environments [35]. It is extensively used in robotics for navigation and obstacle avoidance, as well as in automotive applications for parking assistance systems. While ultrasonic localization is highly effective in detecting objects in cluttered environments where visual methods are insufficient, its accuracy can be compromised by environmental changes. Fluctuations in temperature and humidity affect the speed of sound, thereby impacting localization precision.

2.1.4 . Localization Algorithms

Extensive research in the field of localization for modern IoT applications has charted the progression from traditional to more sophisticated techniques and algorithms. Among the conventional algorithms, linear least squares (LLS) and maximum likelihood estimation (MLE) stand out for their accuracy in precise position estimation [37], [38]. LLS effectively uses RSS values to estimate positions by minimizing the discrepancies between the measured signal strengths and those projected from assumed locations. It provides especially beneficial in settings where a distinct correlation between signal strength and distance is evident, such as indoor networks.

On the other hand, MLE estimates parameters that maximize the likelihood of the observed measurements in the model. It uses various measures such as RSS, ToA, or AoA, and is renowned for its precision in handling complex environments prone to multipath interference. However, despite their effectiveness, both methods come with performance limitations and present significant implementation challenges.

The rapid growth of machine learning and artificial intelligence (AI) across various sectors facilitates their integration with IoT, significantly boosting the intelligence and performance of IoT devices [39]. Incorporating these advanced techniques in localization represents a major improvement over traditional methods. This enhancement enriches localization approaches connected with both cloud and edge computing, enabling better support for IoT applications through advanced analytics, enhanced decision-making, and more efficient resource management.

Additionally, the capacity of neural networks to learn and adapt from vast datasets greatly increases the accuracy and efficiency of localization efforts, paving the way for more autonomous and intelligent IoT systems. Before delving into the advancements of AI-based localization, the next Sec. 2.2 will introduce the basics of artificial intelligence, including neural network architectures and learning mechanisms.

2.2 . Artificial Intelligence and Deep Learning: The Heart of Industry 4.0

2.2.1 . Brief History of Artificial Intelligence

The fascination with creating intelligent systems has been shared by scientists, artists, and philosophers for centuries. In the mid-20th century, British mathematician Turing posed a profound question: "Can machines think?", which sparked initial interest in what would later become known as artificial intelligence (AI) [40]. The term "artificial intelligence" itself was officially coined by McCarthy six years after Turing's query. Since then, the field of AI has experienced cycles of intense enthusiasm and periods of disillusionment, often triggered by the limitations of expert systems and the high costs associated with AI research.

The advancement of machine learning in the 1990s, driven by improved algorithms and more powerful computers, led to significant progress [41]. This era encouraged AI researchers to concentrate on learning-based and probabilistic methodologies and spurred renewed interest in neural networks. These developments marked a pivotal shift at the start of the 21st century, as AI transitioned from traditional machine learning to a focus on neural networks [42].

Following initial progress, the introduction of convolutional neural networks (CNNs) [43] and deep learning [7], coupled with advancements in data storage and the rise of big data, have significantly enhanced AI's capabilities. This progress has been demonstrated in AI's ability to solve complex tasks and achieve breakthroughs in speech recognition, visual recognition, and handwritten text recognition.

Now, in the 2020s, AI has begun to profoundly reshape society and stands as the cornerstone of Industry 4.0, serving as a transformative force across all facets of life. Its integration into various sectors not only enhances operational efficiencies but also fosters innovation, particularly in healthcare, smart manufacturing, and autonomous systems, redefining the potential of human-

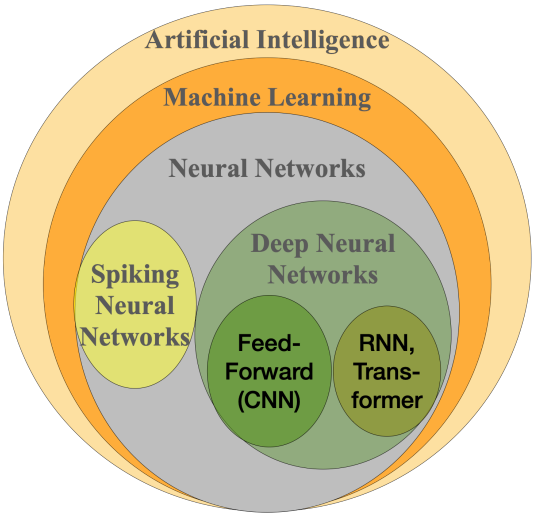


Figure 2.2: Schematic representation of the hierarchical relationship between artificial intelligence, machine learning, and neural networks.

machine collaboration.

As illustrated in Fig. 2.2, the field of AI covers a broad spectrum of technologies and methodologies. At its core, AI branches into machine learning, with further specializations into neural networks. Deep neural networks are a predominant subset of neural networks and represent a critical area of AI advancement, significantly enhancing the capabilities of intelligent systems. Spiking neural networks, another distinct subset inspired by biological processes, have recently gained attention for their energy efficiency in intelligent devices. Next sections will explore the evolution from initial to advanced neural networks (Sec. 2.2.2), focus on deep learning (Sec. 2.2.4), delve into the training process (Sec. 2.2.3), and discuss related challenges (Sec. 2.2.5).

2.2.2 . Evolution of Neural Networks

First Neural Networks

McCulloch-Pitts Neuron — The concept of a neural network was first introduced in 1943 when McCulloch and Pitts proposed a highly simplified model of the brain's neurons [42]. Their model represented neurons as binary units that emit a single output based on multiple inputs. According to this model, the McCulloch-Pitts neuron receives inputs, and fires (outputs a 1) if the sum of the inputs exceeds a predefined threshold, or remains inactive (outputs a 0) if this sum does not exceed this threshold. This fixed threshold underpins the "all-or-nothing" response characteristic of their model, which is defined by the following equation

$$y = \Theta(\sum_i w_i x_i - \text{threshold}), \tag{2.1}$$

where Θ is the step function, w_i are weights (commonly set to $+1$ for excitatory and -1 for inhibitory), x_i are the binary inputs, and threshold is a predetermined threshold value. Their work formed the basis for later developments in artificial neural networks and computational neuroscience. However, the original McCulloch-Pitts model suggests fixed weights, meaning that the connections between neurons did not change, preventing any learning or adaptation. It was static, designed to execute specific logical operations like AND, OR, and NOT, but lacked the capability to learn from data.

Rosenblatts Perceptron — Rosenblatt proposed the perceptron in 1958, a significant advancement that featured adjustable weights and the concept of learning, elements absent in the McCulloch-Pitts model. Initially developed for pattern recognition studies, the perceptron was among the first algorithms capable of learning weights from labeled training data and making decisions by simulating a single layer of neurons. The perceptron learning rule is defined as follows [44]

$$w_i \leftarrow w_i + \eta(t - y)x_i, \tag{2.2}$$

where w_i is the learnable weight, η is the learning rate, t is the desired output, y is the calculated output, and x_i are the inputs. This learning rule is iteratively applied to all weights within the network, and it effectively guides the perceptron towards minimizing the prediction errors, thus learning from the training data. This capability marked a significant advancement as it allowed the model to learn optimal weight configurations for data classification, establishing it as an early example of supervised learning. This leads to the formulation of the perceptron neuron

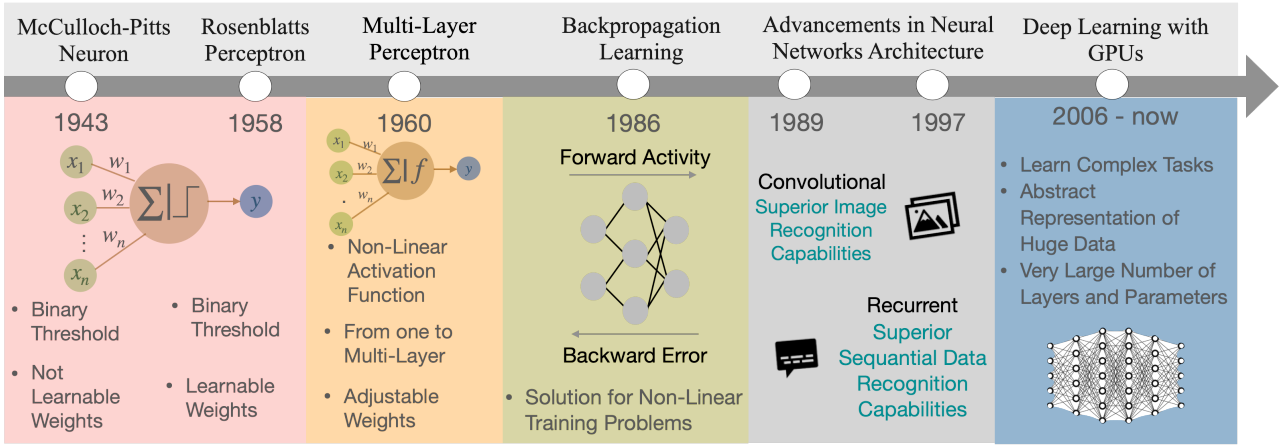


Figure 2.3: Timeline of neural network development: from early concepts to modern deep learning.

model as follows

$$y = \Theta(\sum_i w_i x_i + b). \tag{2.3}$$

In this model, b is the bias, adjusting the threshold at which the perceptron fires, adding an element of flexibility compared to the original McCulloch-Pitts model. Rosenblatt demonstrated the convergence of the learning algorithm in a simple single-layer perceptron through iterative weight adjustments to achieve the desired results. According to the perceptron convergence theorem, the model can only solve linearly separable tasks. Minsky and Papert's work in 1969 highlighted this limitation, notably demonstrating the perceptron's inability to solve the non-linear XOR problem [45].

Multi-layered Perceptrons — The limitations of single-layer perceptrons underscored the need for more complex architectures, leading to the development of multi-layer perceptrons (MLP). Recognized as feedforward neural networks, MLPs consist of multiple layers of neurons, with continuous-valued activation functions to effectuate non-linear transformations of inputs [46]. Each neuron in one layer is fully connected to all neurons in the subsequent layer, ensuring unidirectional data flow from the input to the output layer. Initially, training MLPs relied on heuristic adjustments or rudimentary learning rules, which struggled with the complexities of non-linearities and layered structures. Consequently, their practical applications were initially constrained by the absence of effective training methodologies for multi-layered networks [47].

The introduction of Backpropagation Learning

Backpropagation, introduced in 1986 by Rumelhart, Hinton, and Williams, revolutionized the training of MLPs by enabling effective weight updates across multi-layer fully connected networks [48]. This technique, short for "backward propagation of errors," applies the chain rule of calculus within an optimization algorithm to efficiently calculate the gradient of a loss function with respect to all network weights. Primarily aimed at minimizing the loss function (described in Sec. 2.2.3), backpropagation quantifies the discrepancy between the network's actual output

and the target output. It then adjusts the weights to gradually reduce this loss, facilitating iterative learning from the data.

Advanced Neural Networks

Since the advent of backpropagation learning, neural networks have evolved significantly in the 1980s. In 1989, Yann LeCun's groundbreaking work on convolutional neural networks (CNNs) enhanced the ability of neural networks to process spatial data hierarchies [43], [49], establishing CNNs as a cornerstone architecture for image and video processing tasks. In contrast to feedforward networks like MLPs and CNNs, where data flows linearly from input to output, recurrent neural networks (RNNs), introduced in 1997, effectively manage sequential data [50]. They allow outputs from previous steps to influence the current state, making them ideal for tasks such as language modeling and speech recognition. The field reached a further milestone in 2017 with the introduction of transformers [51], which revolutionized sequence processing through self-attention mechanisms that assess the relevance of different inputs regardless of their sequential order.

Deep Neural Networks

The term "deep" began to be widely used to describe neural networks as they evolved to effectively learn complex patterns through multiple hidden layers [7]. These networks, known as deep neural networks (DNNs), now include a variety of complex architectures designed for diverse applications. Some DNNs use a straightforward feedforward data flow, while others, known as non-feedforward networks, incorporate cycles or loops in their structure.

Apart from transformers, which use a distinct mathematical model, the most common neuron model used in DNNs is

$$y = f(\sum_{i}^{n} w_i x_i + b), \tag{2.4}$$

where f is a non linear activation function, w_i re the learnable weights that can be positive or negative real numbers, associated with inputs x_i, and b is the bias term. The bias allows for an adjustment of the activation function, improving the network's capability to accurately fit the data.

2.2.3 . Training the Network

Training Process

The objective of training a neural network is to minimize the error between the predicted outputs and the actual target values. In the training process, the available dataset is divided into three subsets: training, validation, and test sets. The training set updates the model's weights and biases iteratively using the backpropagation algorithm or its advanced variants. The validation set aids in tuning hyperparameters (described below) and mitigating overfitting by assessing the model's performance on unseen data during training. After training, the test set evaluates the final performance of the model. Throughout the training phase, data encoded as input signals is processed from the input neurons through successive layers to the output layer. Key elements of the training include selecting suitable hyperparameters, activation functions, loss functions, and optimization algorithms to improve model robustness.

Hyperparameters

Hyperparameters are variables that are not learned from the training process itself but are set prior to training [52]. They are optimized using the validation set to achieve the best performance, including settings such as learning rate, batch size, number of epochs, dropout rate, and optimizer, detailed in Tab. 2.1.

Hyperparameter	**Objective**
Learning Rate	Controls the speed of weight updates during training.
Batch Size	Defines the number of samples used to process the model at once.
Number of Epochs	Defines the number of times the entire training dataset is passed through the network.
Dropout Rate	Prevents overfitting by adding constraints during training.
Optimizer	Governs how the network updates its weights based on gradients.

Table 2.1: Key Hyperparameters and Their Objectives

During each training epoch, the model involves iterative adjustments to its weights and biases to reduce the loss function, which quantifies the difference between the network's predicted outputs and the actual target values. These adjustments, essential for enhancing model accuracy, are driven by optimizers such as Stochastic Gradient Descent (SGD) and Adaptive Moment Estimation (Adam) [52]. These optimizers use gradients computed during backpropagation to strategically update model parameters and reduce loss. SGD updates the model parameters by calculating the gradient of the loss function for each mini-batch of data and adjusting parameters in the reverse direction of the gradient. Adam, on the other hand, adjusts the learning rate for each parameter using the running averages of the gradients and their squared values, promoting faster convergence and better performance in complex scenarios [53].

Alongside the hyperparameters used for training, the choice of the loss function is crucial as it evaluates how closely the model's predictions match actual values [54]. For regression, Mean Squared Error (MSE) and Mean Absolute Error (MAE) effectively measure the discrepancies between predictions and actual outcomes. In binary classification, Binary Cross-Entropy is prevalent as it measures the accuracy of probability predictions for two categories. For multi-class classification, Categorical Cross-Entropy assesses the alignment between predicted probabilities and one-hot encoded true labels.

Activation Functions

Several non-linear activation functions have been adopted and played a critical role in enhancing the training and testing performance of DNNs, enabling them to learn faster and reduce gradient problems [55]. The commonly utilized activation functions, as depicted in Fig. 2.4, are standard logistic function (sigmoid), Hyperbolic Tangent (Tanh), and Rectified Linear Unit (ReLU). Each function is selected based on the specific requirements and characteristics of the task being addressed.

The sigmoid function is often used in binary classification models to generate probabilities between 0 and 1 [56, 57, 58, 59], while softmax is preferred for multi-class classification [60, 61, 62]. The Tanh function, with outputs ranging from -1 to 1 and being zero-centered, is frequently used in RNNs to help maintain balanced gradient flow [63, 64, 65]. The ReLU function is commonly used in CNNs for its computational efficiency, which significantly accelerates the training process [66], [67].

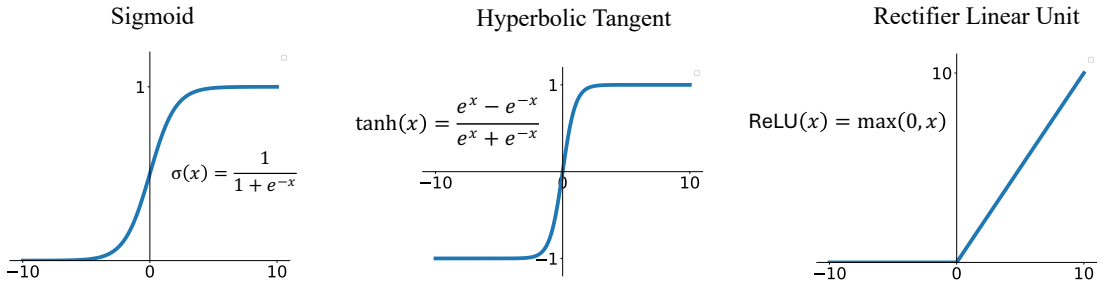


Figure 2.4: Common activation functions in neural networks: Sigmoid, Hyperbolic Tangent, and Rectified Linear Unit

Training Platforms

A variety of platforms are available for neural network training. TensorFlow [68, 69], developed by Google, and PyTorch [70], created by Facebook, stand out as popular open-source frameworks due to their robustness and flexibility. Keras [71], which operates on top of TensorFlow, simplifies model building with its user-friendly Application Programming Interface (API). Additionally, cloud-based solutions such as Google Colab [72], Amazon SageMaker [73], and Microsoft Azure Machine Learning [74] offer scalable resources for large-scale training, making them accessible for both beginners and professionals.

2.2.4 . Deep Learning: Today's Dominant Trend

Deep learning has profoundly transformed the field of AI, especially with the advancement of DNNs, broadening the scope and complexity of addressable problems. This transformative era kicked off in 2006 when Hinton's team demonstrated that deep networks could be effectively trained using a pre-training strategy [75]. The subsequent incorporation of graphical processing units (GPUs) and distributed computing further accelerated this revolution [76], [77], providing the necessary computational power and access to extensive datasets. As a result, deep learning

architectures for CNNs [79] and RNNs [78], have significantly broadened the applications and challenges that neural networks can tackle.

One of the most transformative applications of deep learning has been in image recognition, highlighted by the introduction of AlexNet in 2012 by Hinton's team [80]. By using CNNs and GPU acceleration, AlexNet significantly enhanced performance in the ImageNet challenge, a large-scale repository of annotated images. This breakthrough led to the rapid adoption of deep learning across the field of computer vision, establishing new standards for image recognition tasks. In the realm of strategic games, AlphaGo was developed by Google's DeepMind in 2016 [81], which is an intelligent program capable of mastering the complex board game Go. AlphaGo's victory over world champion Lee Sedol in 2016 marked a significant milestone for deep reinforcement learning, illustrating its potential to solve problems requiring complex strategic thinking. This achievement underscored deep learning's ability to not only match but surpass human expertise in high-dimensional problems.

The field of natural language processing has seen substantial progress with the introduction of the ChatGPT series by OpenAI. GPT-3, launched in 2020, emerged as an advanced language model distinguished by its remarkable capabilities in tasks such as text generation and question answering, facilitated by few-shot learning [82]. In 2023, the release of GPT-4 marked further advancements, leveraging an expanded transformer architecture with more parameters and refined training techniques to improve upon its predecessor's capabilities [83]. By 2024, GPT-4 Optimized (GPT-4o) was developed to boost efficiency and performance, particularly suited for real-time applications and environments with limited resources [84]. These developments illustrate the rapid evolution of deep learning, increasingly hinting at the emergence of general artificial intelligence capabilities in the 21st century.

2.2.5 . The challenges of AI and Deep Learning

The rapid advancement of AI and the growth of deep learning technologies come with significant challenges. The drive to capture increasingly complex data patterns and enhance performance on demanding tasks has led to more complex neural network architectures. These networks often feature an exponential growth in the number and size of parameters. Training these expansive models requires vast amounts of data and computational resources, which poses considerable impacts due to high energy consumption [9, 10, 85]. Moreover, while employing software and algorithmic strategies like feature selection and pruning can reduce the model's parameter count and power usage, these methods typically come at the expense of diminished accuracy [86], [87].

An example of the high costs associated with advanced AI models is ChatGPT-4, which boasts a trillion parameters and consumes substantial energy measured in MWh during training [82]. This training typically takes place on powerful cloud servers, using extensive computing resources to manage its vast parameter set. In contrast, even deep networks models designed for edge devices, which aim for efficiency, still maintain a large number of parameters but manage to keep a moderate power consumption. For instance, Google's MobileNet, optimized for real-time processing on resource-constrained devices, is designed for tasks like image classification and object detection [88]. MobileNet uses CNNs and comprises approximately 4.2 million parameters, demonstrating a balance between capability and efficiency suitable for edge computing.

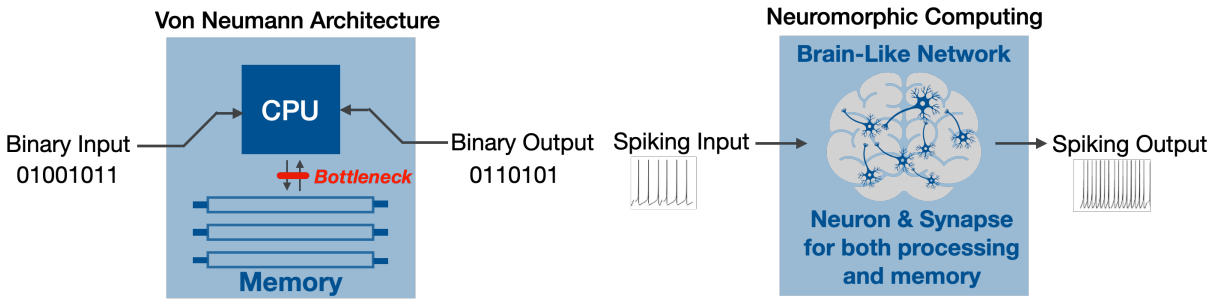


Figure 2.5: Comparative overview of computing paradigms: traditional Von Neumann architecture versus neuromorphic computing

Currently, DNNs are trained on classical computers that operate under the von Neumann architecture (shown in Fig. 2.5), characterized by a distinct separation between memory and processing units [91]. The primary processing units used are the Central Processing Unit (CPU) and the Graphics Processing Unit (GPU). The computational needs of deep learning are diverse, involving various operations that necessitate specialized hardware capabilities. At the core of deep learning processes, tasks like matrix multiplications and vector additions are predominantly performed on GPUs, which are suited for parallel processing [89]. CPUs, on the other hand, manage data loading, preprocessing, and input/output operations during training. Additionally, the storage of large datasets and extensive model parameters, such as weights and biases, highlights the critical role of memory in deep learning computations [90].

Von Neuman Bottleneck

The computational demands of AI, particularly deep learning, are increasing at a rate that surpasses Moore's Law, with capabilities doubling roughly every 5 to 6 months [8]. Contemporary AI models require vast data and extensive parameter tuning for training, often surpassing the memory capabilities of even advanced GPUs like the NVIDIA A100 GPU, launched in 2020 for AI applications [90], [92]. Additionally, the frequent data transfers between memory and processors, a challenge known as the von Neumann bottleneck shown in Fig. 2.5, lead to substantial energy use and latency [93]. This issue is especially pronounced during training, when model parameters are continuously read from and written back to memory for gradient computations.

The high power consumption, the inherent inefficiencies of the von Neumann architecture, and many other problems have become increasingly apparent, which paves the way to new trends, such as the neuromorphic computing and the edge AI. Neuromorphic computing is the alternative promising of the von Neuman bottleneck as it mimics the brain's neural architecture by integrating memory and processing units, enabling parallel and event-driven computation [13], [11]. Edge AI, on the other hand, mitigates AI problems by processing data locally on devices [5], [6], reducing the need for frequent data transfers between the device and cloud-based servers. It leverages specialized AI hardware, such as AI accelerators with integrated memory, to achieve improved computational efficiency. The most benefit from all that is to enable the energy-efficient Edge AI which opens the question of: What if we benefit from both by implementing neuromorphic computing on Edge AI applications?

2.3 . Neuromorphic Computing: Towards Brain-Inspired AI

The concept of neuromorphic computing was first introduced by Carver Mead in the 1980s [94]. Mead described systems that mimic biological neural systems using mixed analog-digital implementations as "neuromorphic" [95]. Over the years, neuromorphic systems have evolved from traditional digital approaches, drawing inspiration from the structure and function of the human brain, particularly neurons and synapses. Recently, neuromorphic architectures have gained significant attention as potential solutions for addressing the computational complexity of AI and deep learning, and for meeting the demands for enhanced energy efficiency in large-scale edge devices within the IoT landscape. The advantages offered by neuromorphic computing, making it a promising solution for overcoming current AI limitations, include [11, 12, 13, 96, 97]:

Energy Efficiency — Neuromorphic systems are designed to replicate the human brain's neural architecture, renowned for its exceptional energy efficiency in handling complex computations. Information can be encoded based on the timing, magnitude, and shape of neural spikes, enhancing computational efficiency.

Reduced Latency — Unlike the von Neumann architecture, where data must be transferred back and forth between memory and the CPU, neuromorphic systems often use in-memory computing. This reduces latency and power consumption by performing computations directly where data is stored [98]. This is crucial for applications where timing and quick responses are critical, such as in neuromorphic sensors and edge computing devices.

Highly Parallel Operation — While parallel von Neumann architectures have existed, neuromorphic systems excel in performing simultaneous computations using simple processing components (neurons and synapses).

Real-time Processing — By mimicking the computation of biological neural systems, neuromorphic systems leverage event-driven computation (spikes), meaning they compute only when data is available and use temporally sparse activity for extremely efficient computation.

Adoption of the Spiking Neural Networks — Neuromorphic systems are specifically engineered to implement Spiking Neural Networks (SNNs), the third generation of artificial neural networks. SNNs were developed to provide a more biologically accurate model of how real neurons in the human brain operate, adopting bio-inspired learning mechanisms. Unlike traditional artificial neural networks like DNNs, which process data continuously, SNNs operate on a bio-inspired, event-driven basis, using time-based information. This enhances efficiency by activating neurons only as needed. Implementing SNNs on neuromorphic hardware is beneficial, as the hardware can be designed to operate in an event-driven or asynchronous manner, aligning well with the temporal dynamics of spiking neurons and synapses.

These advantages suggest that neuromorphic computing could address some of the critical challenges faced by traditional AI, particularly in terms of power consumption, speed, and adaptability to real-world environments [99]. Moreover, these advantages translate into further benefits such as scalability, allowing neuromorphic systems to expand without significant increases in power or space requirements. The small footprint, facilitated by integrated processing and memory, makes these systems ideal for space-constrained environments. Additionally, the inherent stochastic nature of spiking neurons introduces randomness into computations, enhancing robustness and reliability.

The neuromorphic computing field draws from a diverse array of disciplines, including materials science, neuroscience, electrical and computer engineering, and computer science. Materials scientists innovate by creating new substances that replicate the functionality of biological neural systems for use in neuromorphic devices. Neuroscientists contribute by applying their research to enhance computational models and emulate biological neural systems with neuromorphic technology. Electrical and computer engineers are involved in designing devices, using various circuit technologies to develop cutting-edge systems. Meanwhile, computer scientists craft network models and algorithms that are inspired by both biological processes and machine learning principles, producing software that supports the real-world deployment of neuromorphic computing systems. Neuromorphic systems have captured significant interest from both industry and academia:

In industry — Recent advancements in neuromorphic hardware like IBM's TrueNorth and Intel's Loihi 1 and 2 highlight the potential for highly efficient computing. IBM's TrueNorth [100, 101], unveiled in 2014 at the IBM Almaden Research Center, boasts a million programmable neurons and 256 million programmable synapses. It excels in efficient pattern recognition and sensory processing, consuming only 70 milliwatts of power. Intel's Loihi 1 [102] debuted in 2017, succeeded by Loihi 2 [103] in 2021. The Loihi 2 is part of the Hala system, introduced in 2022, which stands as one of the largest neuromorphic platforms globally, featuring 1.15 billion neurons and enhancing AI performance with superior energy efficiency.

In academic research — The European Union's Human Brain Project has funded the development of the BrainScaleS-2 and SpiNNaker neuromorphic systems. Developed at Heidelberg University in Germany, BrainScaleS-2 builds on its predecessor launched in 2013, using analog electronic circuits to accelerate neural simulations and achieve high computational efficiency [104], [105]. SpiNNaker [106], designed at the University of Manchester in the UK by Professor Steve Furber, was introduced in 2014. It uses a million ARM processors to simulate large-scale neural networks in real time. The advanced SpiNNaker 2, introduced in 2024, is equipped with 152 ARM-based cores per chip, capable of scaling to emulate 5 billion neurons [107]. This system enhances energy efficiency and performance, meeting the needs of hybrid AI models.

2.3.1 . Inspiration from Brain Functionality

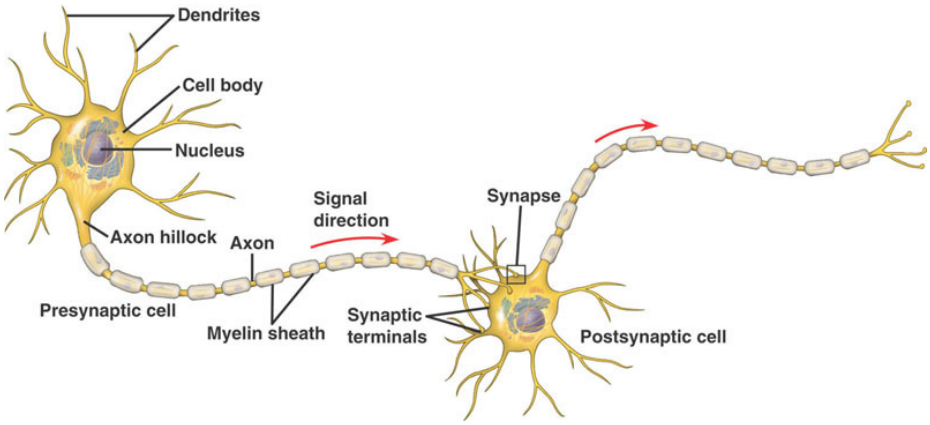
The human brain, with its intricate structure and exceptional capabilities, serves as an inspiring model for advancing artificial intelligence, computing technologies, and efficient hardware development [108]. Notably characterized by its adaptability in learning, decision-making, and pattern recognition, the brain enhances daily interactions with the world and solves problems through intuitive and dynamic responses. Beyond handling routine tasks, the brain's capacity for self-improvement and adaptation in response to new stimuli underscores a level of sophistication that AI aspires to replicate. Despite its densely packed, complex structure and ability to handle intricate tasks, the brain stands out for its energy efficiency, consuming only about 20 watts of power [109]. Therefore, taking inspiration from the brain continues to push the boundaries of what AI can achieve for creating more advanced, low power autonomous systems that can learn from their environments and continuously enhance their performance without human intervention.

A .Basic Neural Units of the Brain

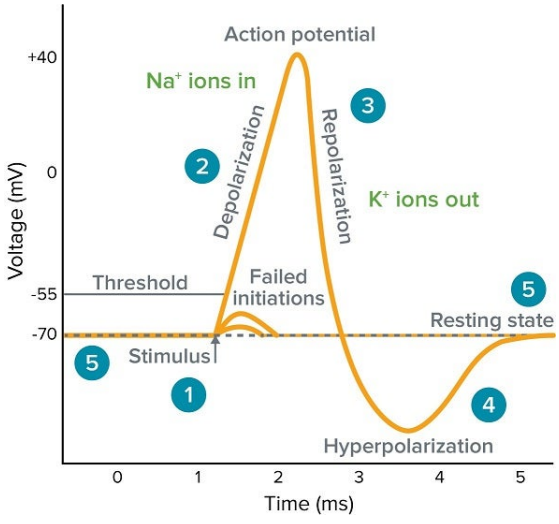
The human brain contains about 50 to 100 billion neurons, along with an even larger number of glial cells that provide support and protection for these neurons [110, 111, 112]. Each neuron can form connections with up to 10,000 other neurons, resulting in over 100 trillion synapses that enable signal transmission across the brain.

Neurons

Neurons, the fundamental cellular units of the brain's nervous system, are central to its function [113], [114]. Each neuron, as illustrated in Fig. 2.6(a), has a cell body called the soma, which contains the nucleus and serves as the metabolic center, where proteins and some neurotransmitters are synthesized. The neuron has distinct input and output sections. Inputs



(a)



(b)

Figure 2.6: Simplified structure of a neuron from [115], (b) Graph of an action potential depicting changes in membrane voltage over time from [116].

are managed by dendrites, branching extensions from the soma that receive signals from other neurons. These inputs are integrated in the soma, and if sufficient depolarization occurs, it triggers electrical impulses. Outputs are managed by the axon, a long, slender projection that transmits electrical impulses away from the neuron's cell body. The axon may be insulated by a myelin sheath, which enhances the speed of signal transmission. The axon hillock, located at the junction between the axon and the cell body, plays a crucial role in initiating the neuron's signal. This signal, known as an action potential or spike, is illustrated in Fig. 2.6(b).

This graph illustrates the stages of an action potential in a neuron [117]. At the **resting state**, the neuron maintains a negative membrane potential, around -70 mV, which is the difference between the electric potential inside the cell and its surroundings. This state is characterized by a higher concentration of potassium ions (K^+) inside the cell and sodium ions (Na^+) outside. When the neuron receives a sufficient stimulus, the membrane potential reaches a threshold that triggers an action potential. This causes a rapid influx of Na^+ ions as voltage-gated sodium channels open, resulting in a sharp rise in membrane potential (**depolarization**).

Following this, the sodium channels close and potassium channels open, allowing K^+ ions to flow out of the neuron, bringing the membrane potential back toward the negative resting level (**repolarization**). The voltage-gated potassium channels close slowly, causing a brief period of **hyperpolarization** where the membrane potential becomes more negative than the resting potential. Eventually, the potassium channels close, and the neuron returns to its resting potential, reset by the sodium-potassium pump and other ion channels that restore the initial ion concentration gradient. After an action potential, the neuron enters a **refractory** period during which it is less likely to fire another action potential. This period allows the neuron to reset before it can generate another action potential.

For any given neuron, the amplitude, duration, and shape of an action potential remain constant once it is triggered. Variability in response to stimulus strength is primarily reflected in the rate or timing of firing. Neurons can exhibit a wide range of spiking behaviors, which will be explored in the next section.

Synapses

As shown in Fig. 2.6(a), neurons connect via synapses, which are junctions where the axon tip of the presynaptic neuron (transmitting neuron) meets a dendrite or the cell body of a postsynaptic neuron (receiving neuron) [118]. Figure 2.7 illustrates that the axon terminal of the presynaptic neuron contains synaptic vesicles filled with neurotransmitters. When an action potential reaches the axon terminal, it triggers voltage-gated calcium channels to open, allowing Ca^{2+} ions to flow in. This influx of calcium causes the synaptic vesicles to fuse with the presynaptic membrane and release neurotransmitters into the synaptic cleft [119, 120]. These neurotransmitters then bind to receptors on the postsynaptic density of the dendrite of the receiving neuron, initiating a response. Recent developments in neuroscience provide evidence that synapses are not merely interfacing elements for transmitting signals between neurons but also play a crucial computational role in biological neural networks. Whether a synapse is excitatory or inhibitory depends on the type of neurotransmitter released and the nature of the receptors on the postsynaptic neuron. Excitatory neurotransmitters increase the likelihood of firing an action potential in the postsynaptic neuron, while inhibitory neurotransmitters decrease this likelihood.

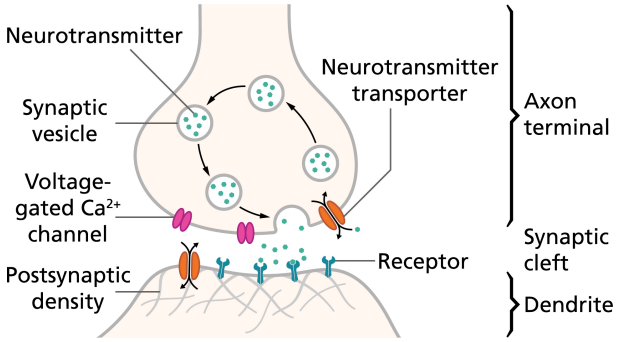


Figure 2.7: Synapse diagram showing neurotransmitter release, synaptic cleft passage, and receptor binding on the postsynaptic neuron, from [121]

B .Neuron Models

The study of neuronal models has been pivotal in advancing the understanding of biological neural mechanisms and the development of computational architectures for SNNs and neuromorphic systems. Neurons in the mammalian cortex exhibit inherently spiking behavior, a fundamental aspect that distinguishes them from artificial neurons. According to [122], neurons can exhibit different firing behaviors, also known as neuro-computational features. Over the years, a variety of spiking neuron models have been proposed in scientific literature, varying in complexity, the number of neuro-computational features, and the biological inspiration they offer. Among the most notable, shown in Fig. 2.8, are:

Leaky Integrate-and-Fire Model (LIF), introduced in 1907, is one of the earliest models of neuronal behavior [123]. It provides an abstraction of a biological neuron, simplifying the neuron's membrane potential dynamics using a single differential equation. This model can represent three neuro-computational features [122]. Despite its simplicity, it effectively captures the essential behavior of a neuron: integrating inputs, exhibiting membrane leakiness (the passive movement of ions through the neuron's membrane), and generating action potentials or spikes once the membrane potential reaches a specific threshold.

$$\frac{dV_m}{dt} = \frac{1}{C_m}\left(I_{syn} - g_L(V_m - V_{rest})\right), \tag{2.5}$$

where V_m is the membrane voltage; C_m is the membrane capacitance; I_{syn} is the synaptic current applied to the neuron; g_L is the leak conductance; and V_{rest} is the resting membrane potential, the voltage of the neuron when it is not excited.

Hodgkin-Huxley Model is a cornerstone in computational neuroscience, developed in the 1950s [124]. It uses a set of four differential equations and ten parameters to describe the changes in the conductances of Na and K channels over time, and their effects on the membrane potential. Although it is the most biologically accurate neuron model, it is also complex and computationally intensive. This model can capture up to 17 neuro-computational features, making it one of the most versatile models available [122].

FitzHugh-Nagumo Model, introduced in 1962, the FitzHugh-Nagumo model was designed to simplify the complexities of the Hodgkin-Huxley model [125], [126]. While it is less biologically inspired and captures fewer firing patterns [122], it retains the essential characteristics of action potential generation and propagation. This streamlined model uses a pair of nonlinear differential equations to achieve its simplicity [127].

Morris-Lecar Model (ML), introduced in 1981, is highly regarded in the computational neuroscience community for its biophysically meaningful and measurable parameters [128]. It can represent 13 neuro-computational features, according [122]. The ML model, specifically designed to simulate the electrical properties of muscle fibers in barnacle giant neurons, uses calcium (Ca) for depolarization instead of Na (typically used in human brain neurons). The ML model achieves a balance between simplicity and biological accuracy through two coupled first-order differential equations [129]. The first equation (2.6) describes changes in membrane potential, incorporating the instantaneous activation of Ca ion conductance, while the second equation (2.7) addresses the activation dynamics of K ion conductance.

$$\frac{dV_m}{dt} = \frac{1}{C_m}\left(I_{syn} - G_{Ca} m_{ss}(V_m)(V_m - V_{Ca}) - G_K n(V_m - V_K) - G_L(V_m - V_L)\right), \tag{2.6}$$

$$\frac{dn}{dt} = \lambda(V_m)\left(n_{ss}(V_m) - n\right), \tag{2.7}$$

where

$$\begin{aligned} m_{ss}(V_m) &= \frac{1}{2}\left(1 + \tanh\left(\frac{V_m - V_1}{V_2}\right)\right), \\ n_{ss}(V_m) &= \frac{1}{2}\left(1 + \tanh\left(\frac{V_m - V_3}{V_4}\right)\right), \\ \lambda(V_m) &= \lambda_0 \cosh\left(\frac{V_m - V_3}{2V_4}\right), \end{aligned} \tag{2.8}$$

where V_m is the membrane voltage of the neuron; C_m is its membrane capacitance; I_{syn} is the synaptic current applied to the neuron; G_{Ca}, G_K, and G_L are the maximal conductances for the Ca, K, and leak channels, respectively; V_{Ca}, V_K, and V_L are the reversal potentials for the Ca, K, and leak channels; m_{ss} and n_{ss} are the Ca and K gating variables; λ is the rate of change for the K channel; and V_1, V_2, V_3, and V_4 are adjustable parameters used to calibrate the steady-state and time constants of the model.

Izhikevich Model, introduced in 2003, provides a computationally efficient method for modeling spiking neurons [130]. It captures a wide range of neuro-computational features (approximately 20 [122]) while maintaining a lower level of biological inspiration compared to models like Hodgkin-Huxley or Morris-Lecar. The model is characterized by a system of only two differential equations

$$\frac{dV_m}{dt} = 0.04V_m^2 + 5V_m + 140 - U_m + I_{syn}, \tag{2.9}$$

$$\frac{dU_m}{dt} = a(bV_m - U_m)), \tag{2.10}$$

where V_m is the membrane potential; U_m is the membrane recovery variable, which influences the firing rate by accounting for the activation of K currents and inactivation of Na currents; a is the time scale of the recovery; and b is the sensitivity of the recovery to the subthreshold fluctuations of the membrane. After a spike reaches its maximum, V_m and U_m are reset to the resting values c and $U_m + d$, respectively.

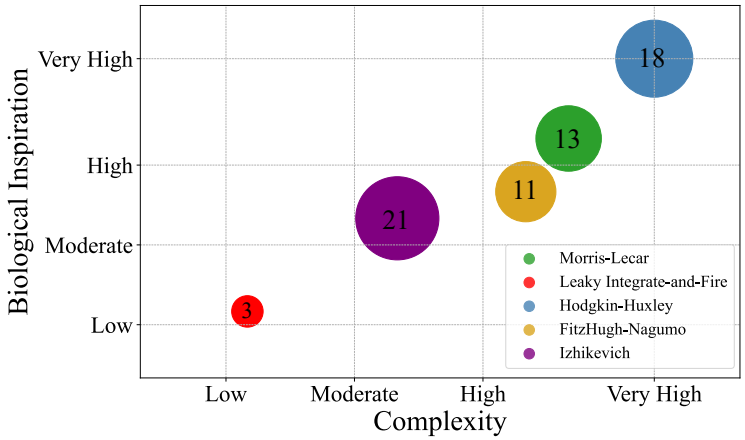


Figure 2.8: Comparison of neuron models based on biological inspiration and complexity, with bubble sizes (and numbers inside) representing the number of neuro-computational features. Inspired by [122] and [13].

2.3.2 . Neuromorphic Hardware

Neuromorphic computing seeks to replicate the brain's exceptional learning efficiency and low energy consumption. Central to this field is the development of SNNs, where neurons communicate through discrete spikes, closely mimicking the behavior of biological neurons. This discipline focuses on understanding and emulating the brain's event-driven dynamics, are fundamentally different from the static architectures commonly used in deep learning. Developing SNNs requires extensive research into energy-efficient hardware (discussed in this section) and bio-inspired learning algorithms (covered in the next section).

Standard CMOS or Emerging Technology?

Neuromorphic hardware primarily comprises two components: neurons and synapses. Some devices and circuits use silicon-based CMOS (Complementary Metal-Oxide-Semiconductor) technology, which has been the industry standard for decades. Others push the boundaries to explore beyond silicon-based CMOS to explore emerging technologies [131], [132]. This thesis focuses on devices based on fully analog CMOS technology for designing neurons and synapses, chosen for their high reliability, low cost, and efficiency [13], [133]. However, the following provides an overview of the most prominent emerging devices developed for neuromorphic computing and their limitations:

Memristors or memory resistors, are passive devices that control the flow of electrical current in a circuit while retaining a memory of past voltages or currents [134], [159]. They

come in two types: non-volatile and volatile. Non-volatile memristors retain their resistance without power, mimicking biological synapses and long-term potentiation, making them ideal for stable synaptic models. Volatile memristors, however, reset their resistance when power is off, resembling the transient behaviors of neurons.

Spintronics or spin transport electronics, use the spin of electrons in addition to their charge to perform computations. In neuromorphic computing, spintronic devices offer the potential for low-power operation and high-density integration. [136], [137] These devices mimic the function of neurons and synapses through magneto-electric phenomena, such as spin-transfer torque and spin-orbit torque, enabling non-volatile memory and state-dependent behavior [138].

Despite their promise, practical implementations of emerging devices face significant challenges due to their higher cost, limited compatibility with existing CMOS technology, and variabilities that lead to inconsistencies. These issues can result in drift and aging, further complicating their use.

A .Electronic Spiking Neuron (eNeuron)

Spiking neurons implemented in CMOS technology have recently become highly popular in research, emerging as a preferred choice for the development of hardware-based neuromorphic computing. This focus is driven by their ability to emulate neural behavior directly in hardware, enabling event-driven computation. Additionally, they offer significant improvements in energy efficiency over general-purpose computer simulations and advance real-time, large-scale neural emulations. Developers can choose analog, digital, or mixed-signal approaches for CMOS neuron design [139], [140]. This thesis specifically focuses on fully analog CMOS neuron (eNeuron) circuits due to their natural, continuous emulation of biological processes, superior power efficiency, and real-time signal processing capabilities [13]. These benefits make analog circuits ideal for creating compact, efficient, and effective neuromorphic systems.

Current neuron circuit designs primarily use the Izhikevich, Morris-Lecar (ML), or Leaky Integrate-and-Fire (LIF) models (described in Sec. 2.3.1). These models are preferred for hardware implementations due to their balance of biological accuracy and computational simplicity. Unlike the complex Hodgkin-Huxley model, which requires extensive computational resources for its nonlinear equations, the Izhikevich and ML models simplify neuronal dynamics, making them more feasible for CMOS integration. Additionally, the LIF model, resembling a basic RC circuit, is ideal for large-scale simulations where minimizing power consumption and minimizing chip dimensions are crucial.

This thesis focuses on the neuro-computational behavior known as Class 1 excitable firing patterns, as identified in the mammalian cortex and detailed in [122]. These patterns can be implemented using the Izhikevich, LIF, or ML models. Class 1 excitability is characterized by a neuron's ability to modulate its spiking rate based on the strength of the input it receives; as input strength increases, so does the neuron's firing rate [141], [142]. This firing pattern is particularly relevant for applications requiring the encoding of a wide range of stimulus intensities with high precision, which is common in sensory processing systems such as vision and hearing.

The literature highlights the wide variety of performance metrics for eNeurons, including the neuron model, technology used, occupied area, spiking rate, power consumption, and energy efficiency. Each of these factors is critical: **(1) the neuron model** determines the fidelity

and complexity of the emulated neural behavior, **(2) the technology** impacts integration, scalability, and system compatibility, **(3) the area occupied** influences the potential for large-scale implementations, **(4) the spiking rate** dictates the responsiveness and operational speed of the circuits, **(5) the power consumption** is pivotal for using these neurons in resource-intensive SNNs, and **(6) the energy efficiency** ensures minimal energy use during spike generation and propagation. Table 2.2 presents a comparison of eNeurons in the state-of-the-art.

In 2006, Giacomo Indiveri and his team introduced a significant progress with their publication of a LIF eNeuron [143]. This eNeuron featured capabilities for spike-frequency adaptation, adjustable refractory periods, and modifiable voltage thresholds. However, the design was complex, using 22 transistors and a capacitor, which resulted in a relatively high power consumption of up to 100 microwatts. Indiveri's research continues to evolve, with efforts to make the neuron circuits more compact and improve their spike timing learning capabilites [144], [145]. His work also aims to achieve success with large-scale network integrations on chips and to diversify the firing patterns of neurons. At the same time, efforts to implement circuits for the Izhikevich neuron model, introduced in 2003, have developed to showcase various firing patterns in fully analog circuits [130]. Later, the focus shifted towards maintaining low power consumption and high energy efficiency to support large-scale networks [146, 147, 148, 149, 150, 151, 152].

In 2017, the pioneering work of Sourpikopoulos and his team introduced ultra-low power biomimetic and simplified eNeurons based on the biologically meaningful ML model [153]. Their success in implementing the ML model involved the configuration of transistors to operate in deep subthreshold conditions, with a supply voltage below 200 mV, to accurately handle the non-linear gating variables that control the currents. The biomimetic version achieved remarkable energy efficiency at 78.3 fJ/spike, significantly lower than Indiveri's less biologically oriented LIF neuron, which has 900 pJ/spike. This achievement was attributed to the minimal supply voltage allowing for low drain-source voltage and deep subthreshold operation. Furthermore, the simplified version of the eNeuron, shown in Fig. 2.9(a) as s-ML eNeuron, greatly increased energy efficiency to 4 fJ/spike. This improvement was made possible by reducing the membrane capacitance, which

Table 2.2: Summary of State-of-the-Art eNeurons and their characteristics.

Ref	eNeuron Model	CMOS Tech. (nm)	Core area (μm^2)	Spike Rate (Hz)	Energy Efficiency (fJ/spike)	Nb Trans / Nb Capa
[143]	LIF	350	2,573	200	900	22/1
[153]	Bio. ML	65	200	1.2×10^3	78.3	8/2
	Simp. ML	65	35	25×10^3	4	6/2
[155]	LIF	65	31	15.6×10^3	2	5/1
[156]	ML	55	98.61	400×10^3	1.95	8/2
[149]	Izhikevich	180	472	NA	58.5	12/2
[157]	LIF	28	32	343×10^3	1.2	8/2
[150]	Izhikevich	65	1050	NA	40.1	19/2
[158]	ML	45	3.91	6×10^3	NA	6/0

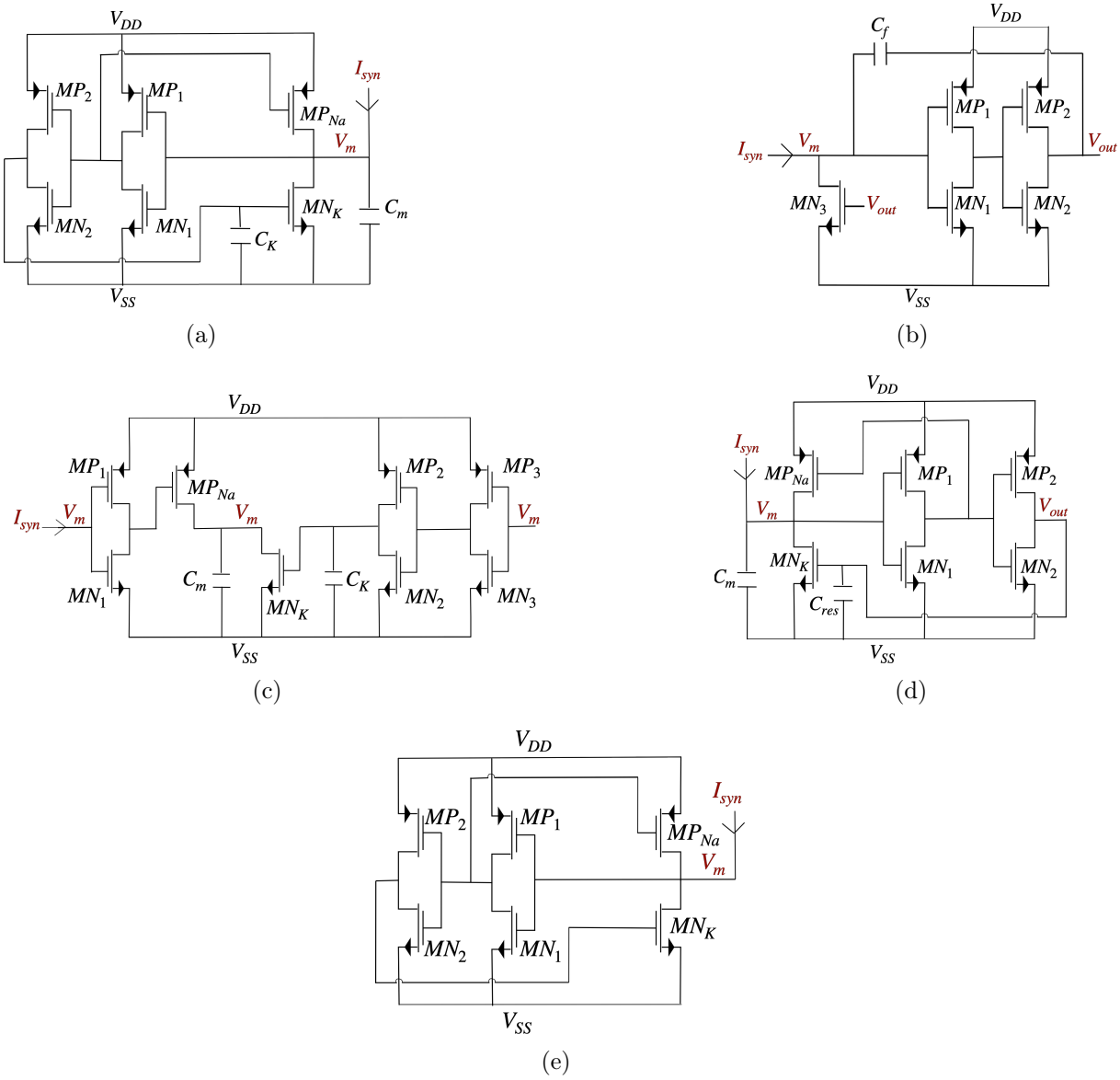


Figure 2.9: Redesigned eNeurons circuits in BiCMOS 55 nm technology: (a) simplified ML (s-ML) from [153], (b) Axon Hillock LIF (AH-LIF) from [155], (c) biomimetic ML (b-ML) from [156], (d) typical LIF (t-LIF) from [157], and (e) parasitic-capacitance based ML (p-ML) from [158].

enhances the charging and discharging rate of the membrane, thereby increasing the spike rate and enhancing overall energy efficiency measured by power consumption per spike rate.

In 2019, Danneville and his team [155] introduced a simplified version of the LIF model known as Axon Hillock [154], and is shown in Fig. 2.9(b) as AH-LIF eNeuron. They achieved an energy efficiency of 2 fJ/spike, prioritizing a low maximum firing rate and exceptionally

low power consumption, measured at just 30 pA. This was achieved by eliminating membrane capacitance and incorporating only a minimal feedback capacitance. Simultaneously, Ferreira and his team [156] developed a biomimetic version of the ML model, shown in Fig. 2.9(c) as b-ML eNeuron. They reach an energy efficiency of 1.95 fJ /spike, representing a 97% improvement over Sourpikopoulos's earlier work. Their success stemmed largely from efforts to minimize substrate leakage and reduce the silicon area in the circuit layout.

In 2022, Besrour and his team [157] developed a LIF eNeuron, shown in Fig. 2.9(d) as typical LIF (t-LIF) eNeuron. It achieves just 1.2 fJ/spike and maintains a small footprint suitable for integration into large-scale neuromorphic circuits. They accomplished this using the advanced TSMC 28 nm technology, a significant upgrade from other eNeurons. More recently, in 2023, Takaloo introduced an ML eNeuron in a capacitance-free circuit, relying only on parasitic capacitances [158]. It is shown in Fig. 2.9(e) as parasitic-capacitance based ML (p-ML). Therefore, this design reduces the circuit area dramatically by 87% compared to the Sourikopoulos version.

All the circuits presented in Fig. 2.9 share a similar design principle centered around the eNeuron's membrane node, which typically includes a membrane capacitance C_m and two transistors, MP_{Na} and MN_K. These two transistors mimic the activity of ions that enter or leave the neuron membrane, functioning as electronic charge pumps regulated by feedback loops to create a series of inverters for Potassium (K) and Sodium (Na) ions. Synaptic current I_{syn} charges C_m through MP_{Na} (pull-up) and discharges it through MN_K (pull-down), causing a significant but brief change in the membrane potential , which triggers the neuron to spike. In the AH-LIF and p-ML eNeurons, C_m corresponds to the feedback capacitance C_f and the parasitic capacitances, respectively.

All the circuits presented in Fig. 2.9 have been redesigned in this thesis for in-depth analysis developed in the subsequent Chap. 3 and Chap. 4. The redesigns were made using the same BiCMOS 55 nm technology (detailed in Appendix C) to ensure that the results are not skewed by technology improvements. The layouts of these circuits are also shown in the Appendix D.

B .Electronic Synapse (eSynapse)

eSynapses play a crucial role in SNNs by linking eNeurons through diverse pre- and post-synaptic signals. They adjust their strengths (synaptic weights) based on neuronal spiking activities, a process known as synaptic plasticity, which is vital for learning and memory. While extensive research exists on eNeurons, there is fewer studies focusing on analog CMOS synapse designs. Currently, the predominant focus in synaptic research has shifted towards emerging technologies, notably memristors, which are preferred due to their inherent ability to mimic biological synaptic plasticity [159], [160].

State-of-the-art eSynapses typically respond to either the spike rate or the spike timing of pre- and post-eNeurons, adjusting their weights accordingly. These eSynapses can be excitatory or inhibitory, thereby increasing or decreasing the firing probability of the post-synaptic eNeuron. Among various designs, the work of Indiveri and his team stands out; they have been pioneering eSynapse development since 2006 [143], [161], focusing on synapses that support spike-timing-dependent plasticity (STDP) learning, a bio-inspired learning detailed in Sec. 2.3.3. In 2017, their team introduced a novel approach that uses arrays of current mirrors to represent synaptic weights through the gains of these current mirrors [162]. This sparked interest in developing

spike-time-based eSynapses that support STDP, leading to various circuit designs capable of adjusting synaptic weights based on different pre- and post-synaptic scenarios [163].

In 2021, Danneville and his team proposed an ultra-low-power eSynapse designed for both excitation and inhibition based on spike-rate detection, using 65 nm technology [164]. This design features transistors operating in the subthreshold region for minimal power consumption, often in the nW range. It employs an RC integrator to capture spike rate information by varying output voltage based on the charge and discharge cycles of incoming pre-eNeuron spikes. The eSynapse then uses two inverters to extend spiking signal duration and a cascode transconductance amplifier to generate synaptic current. By 2022, a fully spike-time-based eSynapse was developed using 28 nm technology, achieving nW-range power consumption and enhancing STDP capabilities under experimental conditions and variations in process, voltage, and temperature (PVT) [152].

A spike-rate-based eSynapse was proposed in 2022 [24], as shown in Fig. 2.10. This design comprises an RC filter, a transconductance amplifier, and a current mirror. The RC filter, made up of a diode-connected transistor and a capacitor, extracts the spike rate from the pre-eNeuron input spikes. This rate is then converted into an excitatory current by the transconductance amplifier, which is modulated by the synaptic weight set by the PMOS current mirror gain. This eSynapse draws inspiration from cutting-edge research: plasticity is given through the current mirror architecture, where the current mirror gain is a synaptic weighted bias signal [162]; ultra-low power consumption is achieved by using RC and transconductance components [164]. This excitatory eSynapse was integrated into this thesis work, achieving a low power consumption ($\approx$ 0.4 nW) and a compact area. It was designed using BiCMOS 55nm technology, with the layout and appropriate component sizes detailed in Appendix D.

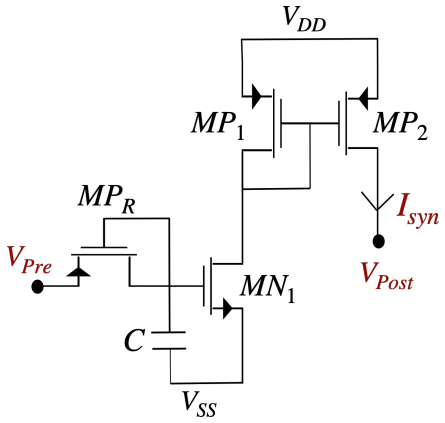


Figure 2.10: Redesigned eSynapse circuit in BiCMOS 55 nm technology from [24].

2.3.3 . Neuromorphic Learning Algorithms

To develop effective neuromorphic learning algorithms, it is essential to emulate the biological learning processes observed in the brain. Learning in the brain is a complex and multifaceted process, continuing to be a dynamic field of research. Most scientists believe that the brain encodes information through specific mechanisms such as the precise timing, rate, and patterns of spikes across extensive neuronal groups. However, the exact algorithms underlying learning in the brain remain not fully understood.

A learning algorithm is considered bio-plausible based on several key criteria, with its bio-plausibility increasing as more criteria are met [12]. First, it should be local, meaning that synaptic weight updates rely solely on the information accessible to a neuron, reflecting the decentralized information processing in biological systems. Second, it should use similar operations or circuitry for both the inference and learning phases, facilitating efficient implementation on neuromorphic hardware. Third, it should support sparse coding and be robust to noise, mirroring the brain's ability to represent information efficiently using only a small number of active neurons at any given time.

Another important criterion is the type of learning. Learning algorithms are traditionally classified into two main categories: supervised learning and unsupervised learning, with additional categories emerging [165], [166]. The comparison between these categories, their roles, and their degree of biological plausibility is presented in Tab. 2.3. Supervised learning algorithms require labeled data, using known input-output pairs to train a model for making predictions or decisions [167]. In contrast, unsupervised learning does not use target labels; instead, it autonomously identifies patterns and structures in the data [168]. Semi-supervised learning is a hybrid approach that uses a small amount of labeled data alongside a larger set of unlabeled data, making it useful in scenarios where acquiring labeled data is costly or impractical [169]. Self-supervised learning generates its own training labels from the input data, allowing models to learn patterns and features without external annotations [170]. Reinforcement learning operates

Table 2.3: Comparison of Learning Types

Learning Type	**Label Requirement**	**Data Requirement**	**Degree of Bio-Plausibility**
Supervised	Labeled	High volume of data	**Low** - Does not closely mimic natural learning processes
Unsupervised	Unlabeled	Any volume	**Medium** - Mimics discovery and adaptation without guidance
Semi-supervised	Partially labeled	Mixed (mostly unlabeled)	**Medium** - Reflects mixed learning scenarios in nature
Self-supervised	Self-generated labels	Any volume	**High** - Closely mimics human learning by self-exploration
Reinforcement	No labels, reward-based	Interaction data	**High** - Very similar to how organisms learn from consequences

differently, focusing on learning optimal behaviors through trial and error in an environment, using rewards to shape an agent's strategy [171]. The more a learning algorithm avoids reliance on labeled data, the closer it aligns with biological learning processes, where labeling is not typically present.

There have been various attempts to adapt deep learning, particularly the supervised back-propagation algorithm, to train deep SNNs directly [172]. Deep learning, heavily influenced by statistical learning theory, primarily uses rate-based neurons that perform continuous non-linear mappings of inputs, often interpreted as firing rates [173], [174]. However, it faces several challenges when applied to in-hardware learning on neuromorphic processors: (a) weight updates are not solely based on locally available information, (b) spiking neuron activities are non-differentiable, (c) errors propagate as real values, and (d) synchronous updates are biologically implausible.

To address these issues, advanced learning algorithms have been proposed that draw inspiration from both the biological plausibility of computational neuroscience and the practical effectiveness of deep learning. The bio-plausible learning algorithms are either in a rate or spike-based approaches, where the difference relies on how the synapses adjust their weights [175]. Rate-based learning algorithms uses the average firing rates of neurons to adjust synaptic weights, offering noise robustness and straightforward implementation, which is well-supported by existing software frameworks. Conversely, spike-based or temporal learning algorithms adjust weights based on the precise timing of spikes, allowing for more efficient encoding of complex information, though with increased complexity and hardware demands.

The following examples are the predominant biologically plausible learning algorithms:

Hebbian Learning is summarized by the phrase "neurons that fire together, wire together", and is introduced by Donald Hebb in 1949 [176]. It is a fundamental principle of synaptic plasticity where the strength of a synaptic connection depends on the simultaneous activity of connected neurons. It is often modeled in a rate-based framework, where the strength of synaptic connections increases with high levels of concurrent neuronal activity over time. This dynamic underlies the formation of memory and learning pathways, mirroring the way organisms learn from repeated experiences and stimuli.

Spike-Timing-Dependent Plasticity (STDP) is a prominent spike-based learning algorithm that emerged from experimental neuroscience research in the late 1990s and early 2000s [177], [178]. It is considered a key mechanism for synaptic learning and adaptation, providing a biologically plausible rule for SNNs. STDP is a form of Hebbian learning that specifically considers the precise timing of spikes. This rule is often summarized by the maxim "fire together, wire together; fire out of sync, lose your link". In STDP, synaptic weight changes depend on the relative timing of pre-synaptic and post-synaptic spikes. As shown in Fig. 2.11, if the pre-synaptic neuron fires just before the post-synaptic neuron, the synaptic weight increases. Conversely, if the pre-synaptic neuron fires just after the post-synaptic neuron, the synaptic weight decreases. The weight change function, Δw_{syn}, can be expressed as follows

$$\Delta w_{syn}(\Delta T_s) = \begin{cases} w_+ e^{-\frac{\Delta T_s}{\tau_+}} & \text{if } \Delta T_s > 0 \\ w_- e^{\frac{\Delta T_s}{\tau_-}} & \text{if } \Delta T_s < 0 \end{cases} \tag{2.11}$$

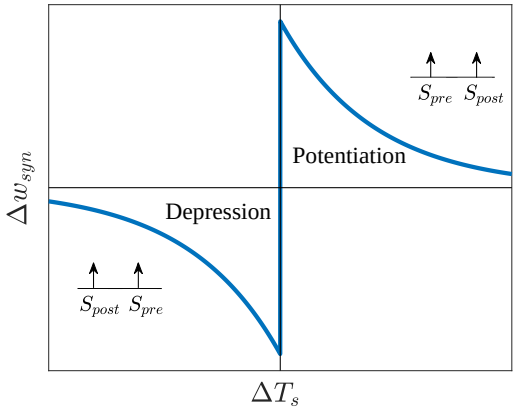


Figure 2.11: Curve shape of the synaptic weight update Δw_{syn} in STDP.

where $\Delta T_s = t_{post}$ - t_{pre} represents the time difference between the postsynaptic spike at t_{post} and the presynaptic spike at t_{pre}; w_+ and w_- are the maximum and minimum values of Δw_{syn}; and τ_+ and τ_- are the time windows that determine the weight update rate for long-term potentiation and long-term depression, respectively.

Equilibrium Propagation (EqProp) is introduced by Benjamin Scellier and Yoshua Bengio in 2017. It combines energy-based model dynamics with neural network learning through two distinct phases [179]. It is modeled as a rate-based learning approach, primarily using continuous neural activation values instead of discrete spikes to compute changes in the network's state. In the free phase, the network naturally settles into a low-energy equilibrium state, similar to the resting state in biological systems. During the nudged phase, a slight perturbation is applied to the output neurons, subtly shifting this equilibrium. The gradient for weight adjustments is calculated from the state differences between these phases, based on local synaptic activities. As a result, EqProp updates synaptic weights using locally available information, eliminating the need for backpropagated error signals from the output to the input layers.

Triplet STDP is an advanced algorithm introduced in 2006 [180]. It refines synaptic adjustments by considering the timing of spike triplets instead of just pairs. Triplet STDP captures the complex and subtle dynamics of synaptic changes in natural neural networks, reflecting the influence of higher-order spike patterns on synaptic plasticity.

EqSpike or Equilibrium Spike, was proposed in 2021 as an extension of the principles of EqProp to SNNs [181]. This algorithm integrates the dynamics of spiking neurons with the energy-based framework of EqProp in a spike-based model. The key advantage of EqSpike is its local computation in both space and time; each neuron uses only its own state and those of its direct connections, eliminating the need to store values during learning.

Training Platforms

Training platforms for SNNs play a pivotal role in the simulation and analysis of these biologically-inspired computational models. Prominent platforms include Brian2, which offers a flexible and user-friendly Python-based interface for simulating SNNs with custom neuron

and synapse models [182]. Another key platform is NEST (Neural Simulation Technology Initiative) designed for large-scale brain simulations and known for its scalability and efficiency [184]. Additionally, SpiNNaker, a specialized hardware platform, supports large-scale, real-time neural simulations of SNNs with high parallelism and energy efficiency [185]. The snnTorch significantly enhances the training of SNNs by offering a platform that integrates smoothly with existing PyTorch workflow [186].

2.3.4 . Advancements in Analog-based SNN Applications

Hardware implementations of neuromorphic computing have seen significant growth in recent years, spanning digital, analog, and mixed-signal platforms [139], [140]. The prevalence of digital platforms in deploying SNNs can largely be attributed to their ease of use and exceptional flexibility. Common digital platforms include Field-Programmable Gate Arrays (FPGAs) and several well-known digital Application-Specific Integrated Circuits (ASICs) such as TrueNorth, Loihi2, and SpiNNaker [12]. Despite the dominance of digital platforms in SNN implementations, analog integrated circuits have consistently been considered prime candidates for neuromorphic systems for several reasons. Initially, the term "neuromorphic" specifically referred to analog designs, highlighting the natural alignment between analog circuits and biological neural processes [154]. Analog systems share key characteristics with biological systems, such as charge conservation, amplification, thresholding, and integration [13]. Moreover, the asynchronous operation of analog circuitry aligns perfectly with the inherent functionality of spiking neural networks, making it well-suited to handle noise and unreliability in signal processing. Additionally, analog circuits can efficiently operate in subthreshold mode, significantly enhancing power efficiency. Given these advantages, analog-based SNNs have been successfully demonstrated in various applications, showcasing their readiness for practical deployment.

Analog-based SNN implementations have been shown in mimicking brain sensory functions [14], [15], even brain's attentional mechanisms [187]. Analog-based SNNs can enhance visual patterns recognitions as they process visual information in a manner akin to the human retina, providing fast response times and low power consumption [188], [189]. Moreover, analog-based SNNs appeared greatly for audio signal processing, as by processing auditory signals in a way that mirrors the biological processes in the human auditory system [190], [191]. These devices can offer a more natural listening experience, improving speech recognition in noisy environments. Implementations were developped in the robotics and biomedical fields, where analog-based SNNs could reflex the highly precision in decision making [192], [193], [194]. They offer promising solutions for treating neurological diseases by providing efficient and adaptive signal processing [195], [196]. Analog SNNs are capable of real-time adaptation to physiological signals, making them well-suited for advanced applications such as prosthetic devices [197], cochlear implants, and neural interfaces.

Recent advancements also highlight the use of FPGAs and SoCs to implement SNNs by modeling the behaviors of neurons and synapses along with bio-plausible learning mechanisms. These implementations have demonstrated high accuracy and energy efficiency in various tasks, including image recognition [198] and communication systems [199], [200], [201] where low latency and power consumption are crucial. In addition to the development of efficient neuromorphic models and chips, there has been intensive work on ensuring their testing and reliability for

fault detection and manufacturing [202]. These advancements collectively push the boundaries of what neuromorphic computing can achieve, paving the way for its applications. The focus on energy efficiency, real-time processing, and robustness ensures that these systems can meet the demands of future technological challenges.

2.4 . Neural Networks for RF Localization

As highlighted in previous Sec. 2.2 and Sec. 2.3, Artificial Intelligence has transformed a variety of applications through the adoption of neural networks. These range from minimally bio-inspired deep neural networks to highly bio-inspired spiking neural networks. In the realm of RF localization, neural networks have significantly changed the ways in which signal data are interpreted and used for positioning and tracking. Moreover, neural networks, with their ability to model complex relationships and learn from vast amounts of data, offer significant advantages over traditional localization techniques. This section presents the prominant works of state of the art in RF localization using both DNNs (in Sec. 2.4.1) and SNNs (in Sec. 2.4.2). This discussion is especially relevant as this thesis addresses the RF localization problem through efficient AI-based solution.

Localization, as discussed in Fig. 2.1, can be achieved through various technologies and measurement techniques. This section will explore these methods, with a particular focus on RF localization using the RSS technique, as adopted in this thesis. RF sensing is chosen for its suitability in indoor environments and easiness of integration. The RSS method is preferred due to its low cost and low power requirements, making it an ideal choice for efficient localization.

2.4.1 . Deep Neural Networks for Localization

Given the constraints of conventional localization techniques discussed in Sec. 2.1, deep learning has gained significant interest in recent years for addressing complex localization problems. This shift has driven the development of extensive datasets and specialized network architectures for localization. Here are some deep networks examples implemented for localization, demonstrating their prominent performance:

In 2016, a feed-forward network was proposed for RSS-based indoor localization, achieving over 96% accuracy [203]. This approach treats localization as a classification problem in deep learning. It has since been widely adopted for its applicability to various forms of localization, whether for objects or sources, by mapping potential positions similarly to image classification. In 2019, a CNN was developed for indoor localization using Wi-Fi fingerprinting, one of the most practical methods for localizing mobile users in multi-building, multi-floor environments [204]. This framework uses dropout layers to prevent overfitting and achieves a high accuracy rate of 94%.

In 2019, a real-time CNN was proposed for sound source localization using Direction of Arrival (DoA) estimation [205]. It used an Android smartphone and its two built-in microphones, performing effectively under noisy conditions with low latency. At a signal-to-noise ratio (SNR) of 0 dB, the model achieved 90% accuracy with simulated data and 88% accuracy with real recorded data. In 2022, a parallel CNN-based real-time sound localization system using microphone arrays was introduced [206]. At 0 dB SNR, it achieved 91% accuracy and a DoA angle error of 7 degrees.

Additionally, a hybrid TDoA-RSS localization algorithm was developed to address multipath fading challenges in two-dimensional space with two base stations [207]. By incorporating both signal amplitude (RSS) and propagation delay time (TDoA) measurements, it achieved minimal deviation error but in a complex model.

In contrast to classification-based approaches, a long short-term memory (LSTM) recurrent neural network was proposed in 2019 to create a regression model between fingerprints and locations for tracking moving targets [208]. This method focuses on enhancing RSS-based localization to overcome the time bottleneck and computational limitations of CPU processing. In 2021, a combined CNN-LSTM network for RSS-based localization was presented in [209]. The system's performance was evaluated by the average error, measured as the Euclidean distance between the estimated and actual source locations. Experimental results demonstrated that the fusion neural network achieved an average location error of 0.03 meters, representing at least an 11.4% improvement over using CNN and LSTM methods independently.

Despite significant progress in achieving high accuracy, deep networks used for localization still face many challenges. These networks often require digital signal processing and involve complex architectures with extensive tuning parameters. Moreover, they are computationally intensive, difficult to integrate into hardware, and result in significant power consumption [9], [10]. All these limitations, discussed in Sec. 2.2.5, can be addressed by neuromorphic computing and its core component, SNNs, offering bio-inspired and highly efficient localization solutions.

2.4.2 . Spiking Neural Networks for Localization

SNNs are designed to emulate the brain's efficient, adaptive, and rapid processing capabilities, making them ideal for implementing complex neural network architectures and learning algorithms across various applications, as detailed in Sec. 2.3. Most research in neuromorphic computing has been geared towards emulating human sensory functions and developing energy-efficient neuromorphic processors. To maximize the benefits of existing neuromorphic systems, they should be integrated directly with event-driven sensors, as currently, few are complete end-to-end systems. Furthermore, few studies have focused on implementing applications beneficial for IoT sensory functions, such as pressure, temperature, ID detection, or device localization, which differ significantly from human sensory functions. Here are some works on spike-based neuromorphic systems dedicated to localization that could be pertinent for edge AI applications in IoT.

In 2022, Moro et al. have proposed in [210] an end-to-end neuromorphic system for ultrasonic object localization, using LIF neurons and resistive memory (RRAM)-based computational maps. This bio-inspired system mimics the auditory localization capabilities of barn owls by measuring time differences in ultrasonic wave propagation using two transducer sensors. The system achieves an angular resolution of 4 degrees and consumes just hundreds of nanowatts. However, its operation in the hundreds of kHz range makes it unsuitable for indoor RF localization.

In 2023, new architectures have introduced for RF signal classification [211] and drone identification [212], achieving accuracies above 90% while consuming power in the milliwatt range. These designs incorporate emerging spintronics technologies to develop nanoscale, fast, and energy-efficient neuromorphic components (neurons and synapses). Despite their advancements, these technologies face challenges such as higher costs and limited reliability and compatibility com-

pared to traditional CMOS technologies. Currently, the potential of analog CMOS-based SNNs for RF localization has not been thoroughly explored in state-of-the-art research.

2.5 . Conclusion

As the IoT continues to expand, with an ever-increasing number of connected devices and progressively complex communications, the demand for precise and energy-efficient localization technologies becomes more critical. Significant advancements in artificial intelligence and deep learning have partly met this need by revolutionizing localization and outperforming conventional technologies. This chapter provided an overview of the development and evolution of AI and deep learning over the years. It detailed how deep neural networks managed complex problems through nonlinear functionalities and training on large datasets. It also addressed the challenges posed by their computational demands, which approximately doubled every 5 to 6 months. The chapter then introduced neuromorphic computing as a bio-inspired alternative, highlighting its advantages over traditional AI. The discussion included an in-depth look at the functionality of biological neurons and synapses, various neuron models, and the current advancements in analog-based spiking neurons and synapses, along with bio-inspired learning algorithms. Lastly, the chapter outlined AI-based research efforts in the field of localization, highlighting the identified gaps. These gaps will be addressed in the subsequent Chap. 3.

Chapter 3

Towards Efficient RF Localization: An Analog Spike-Based Neuromorphic Approach

As the world becomes increasingly interconnected, the ability to locate devices underpins key functions critical to the IoT ecosystem. Efficient RF localization is essential, driving the need for precise and energy-saving solutions. Recent literature showcases significant advancements in highly precise RF localization, propelled by the adoption of advanced techniques like artificial intelligence (AI) and machine learning [209]. However, current approaches still face numerous challenges such as complex architectures, extensive tuning requirements, and computation-intensive processes [10].

This chapter presents the RF NeuroAS, a radiofrequency neuromorphic system using an analog spike-based approach to emulate brain-like capabilities in addressing RF localization challenges. RF NeuroAS exploits the potential of neuromorphic computing through spiking neural networks, to perform complex operations with minimal energy consumption. Concurrently, it incorporates the strengths of CMOS analog technology, known for its low-cost and low-power consumption. The primary aim of RF NeuroAS is to identify the location of a mobile source on a two-dimensional (2D) plane, achieving high precision and energy efficiency.

Figure 3.1 depicts the RF NeuroAS architecture, divided into four key components: RF environment setup, data extraction, neuromorphic pre-processing, and analog spiking neural

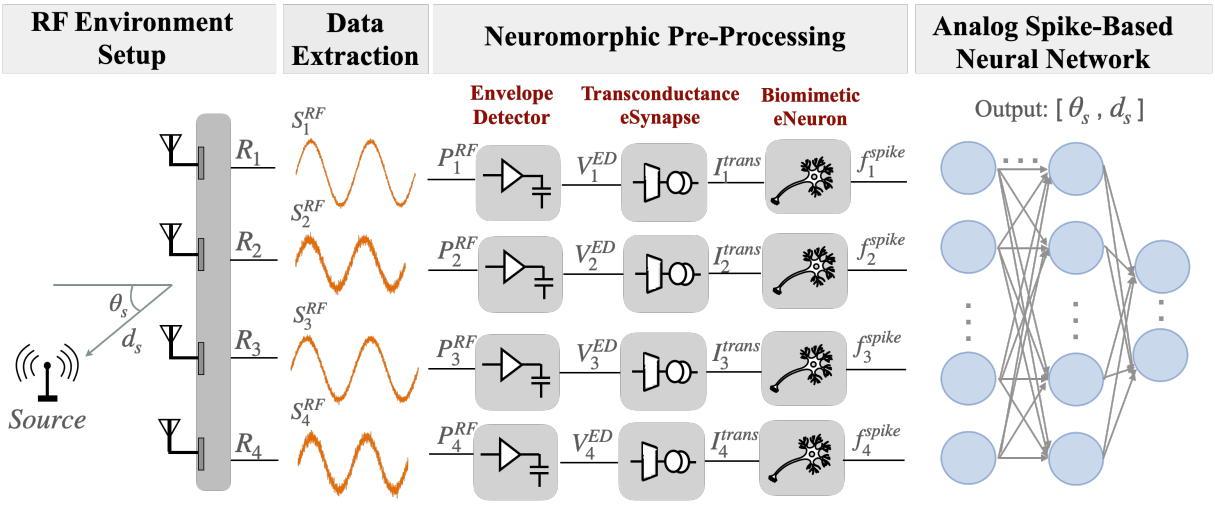


Figure 3.1: System-level architecture of RF NeuroAS for source localization, including four stages: RF environment setup, data extraction, neuromorphic pre-processing, and analog spiking neural network. Outputs include the position of the source, given as angle θ_s and distance d_s.

network. Initially, Sec. 3.1 outlines the arrangement of source and receiver antennas within the RF field, enabling accurate source identification by coordinates angle θ_s and distance d_s. Then, Sec. 3.2 delves into data extraction from the RF environment, leading to the creation of both simulated and measured datasets. Subsequently, Sec. 3.3 details the design and functionality of the neuromorphic pre-processing circuit. Moreover, Sec. 3.4 explores the analog-based spiking neural network, incorporating the functions of analog spiking neurons (eNeurons) and synapses (eSynapses). RF NeuroAS accurately determines the source's coordinates with a **10-degree angular resolution**. A refined version of RF NeuroAS is introduced, designed for enhanced precision in source localization while maintaining low power consumption, achieving a **1-degree angular resolution**. Further details are available in Appendix A. Finally, Sec. 3.5 presents a simplified version of the RF NeuroAS system, fully designed on an analog circuit layout to validate the feasibility of hardware RF NeuroAS solutions. The code used to run the simulated and measured datasets, as well as the neural network training, is available in [28].

3.1 . RF Environment Setup

The RF configuration plays a critical role in defining the mobile source's path, the requisite number of receivers, their locations, and other vital parameters within the RF environment. Figure 3.2 illustrates this setup on a 2D plane, marking the positions of both the source and receivers. In this scenario, the source is a mobile point that can assume any position denoted by red crosses, situated across three concentric circles. These circles denote three distinct distances (d_s) from the origin to the source, specified as 0.1, 0.23, or 0.5 meters. Moreover, they afford complete 360-degree coverage, enabling the source's angle relative to the origin (θ_s) to range from 0 to 360 degrees in increments of 10 degrees.

Regarding the receivers, there are four, each one stationed at predetermined locations marked by blue squares in Fig. 3.2. They occupy the midpoint of each plane boundary to offer full

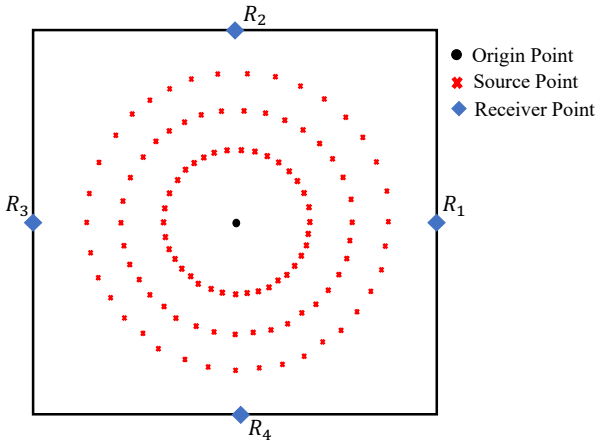


Figure 3.2: A 2D spatial configuration map of the source and receivers. The black dot marks the origin of the plane; red crosses denote the potential positions of the RF source, while blue squares indicate the established locations of the four receivers.

coverage and to improve localization accuracy within the 2D layout. Specifically, Receiver 1 is situated to the right, Receiver 2 at the top, Receiver 3 on the left, and Receiver 4 at the bottom, each maintaining an equal distance (d_r) from the origin at 1 meter. Accordingly, the fundamental parameters settings applied to this environment are presented in Tab. 3.1.

Table 3.1: Main parameters configuration used in the RF environment

Parameters	**Values**
Operating Frequency [GHz]	2.4
Wavelength [cm]	12.5
Transmitted Power [dBm]	10
Antenna Gain [dB]	1.5
Receiver-Origin Distance [m]	1
Source-Origin Distance [m]	0.1, 0.3, 0.5
Source Angle [degrees]	0-360
SNR [dB]	0, 10, 20

3.2 . Data Extraction

In the data extraction stage of RF NeuroAS, RF data are collected for different source locations and under various noise conditions. This data is used to organize a dataset vital for analyzing source localization scenarios. Specifically, this dataset plays a crucial role in evaluating the decision-making capabilities of the RF NeuroAS system. It is used for data pre-processing in the third stage and evaluating the neural network in the fourth stage of the system, as shown in Fig. 3.1. To enhance the learning robustness and performance of RF NeuroAS, two distinct datasets have been developed. The first, a simulated dataset, is generated in MATLAB to model scenarios with the principles of electromagnetic theory. The second, an empirical dataset, is obtained from actual measurements of the RF experimental setup.

The dataset's framework is presented in Fig. 3.3, and it is split into features and labels. Data samples, structured in the rows, capture various source positions (SP) identified by instances of source angle (θ_s) and distance (d_s). To enrich data diversity, each receiver is subject to random noise on 200 different iterations of each source position. The feature columns record the signal power P_{RF} from the four receivers (R_1 to R_4). For labeling, a classification method is adopted to simplify the neural network structure outlined in Sec. 3.4. This method assigns labels based on the source's region, angle within that region, and distance from the origin. In this case, the entire 360-degree range is divided into four regions labeled as r_{s1} to r_{s4}, and each covering 90 degrees ($n_r = 90$). Within the predicted region, the specific angle is then categorized into one of nine possible classes (a_{s1} to a_{s9}), corresponding to a 10-degree increment. Hence, the source's angle θ_s is calculated by $r_s \times n_r + a_s$. The source's distance is designated as one of three pre-defined categories (d_{s1} to d_{s3}). This classification model aims to predict the source's location by learning from the patterns between the received power features and the corresponding source labels.

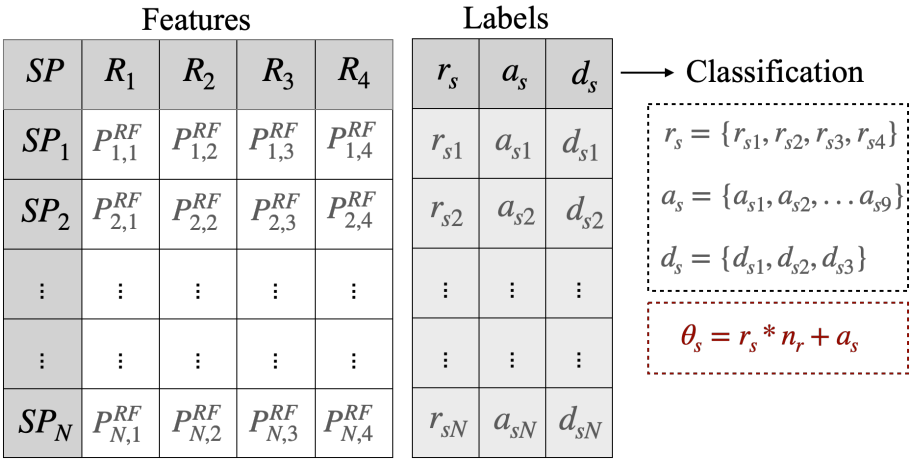


Figure 3.3: Dataset structure for RF source localization. Features comprise power levels from receivers R_1 to R_4 for different source positions SP, and labels are classified by three attributes: the source's region r_s, its angle within the region a_s, and its distance from the origin d_s. The source's angle θ_s is subsequently computed.

3.2.1 . Simulated Dataset

This dataset, named from this point as SimLocRF, simulates a situation in which a source radiates electromagnetic waves in a free space environment. Both the source and receivers are co-polarized antennas, positioned as illustrated in Fig. 3.2. Features are derived by applying the Friis equation to determine the power available at the output terminal of a receiver antenna R_i, described as [213]

$$P_i^{RF} = P_T + G_T + G_{R_i} - PL_i, \tag{3.1}$$

where P_i^{RF} is the received power at receiver R_i; P_T is the power delivered to the source antenna at its input terminal; G_T represents the gain of the source antenna towards the receiver antenna, whereas G_{R_i} is the gain of the receiver antenna towards the source antenna; P_{L_i} denotes the path loss, which is the decrease in power density as the electromagnetic wave travels from source to receiver R_i. The path loss PL_i can be expressed as

$$PL_i = 20log_{10}(d_i) + 20log_{10}(f_{RF}) - 20log_{10}\frac{c}{4\pi}, \tag{3.2}$$

where d_i is the distance between the transmitter and receiver R_i; f_{RF} is the operating frequency of the RF signal (2.4 GHz here); c stands for the speed of light, i.e. 3×10^8 meters per second.

The simulated dataset contains 72,000 samples, generated with white Gaussian noise at three different SNR levels (0, 10, and 20 dB) to improve model performance and enrich data variability. A MATLAB-based framework used to generate the simulated dataset is available in Appendix B, with the corresponding code in [28].

3.2.2 . Experimental Dataset

The main purpose of the experimental dataset, named from this point as MeasLocRF, is to offer insights into how signals travel and are captured by the receivers. Using measurement data

helps to evaluate the neural network's effectiveness in practical scenarios, beyond theoretical-based predictions. RF parameters are collected from the experimental setup, creating a dataset that aligns with the simulated dataset's format. MeasLocRF is extracted from setups where both source and receivers are positioned within an anechoic chamber to minimize reflections and external interference, as depicted in Fig. 3.5. This dataset, containing 3,000 samples that capture varied angular positions of the source, is processed in MATLAB to incorporate the three levels of noise, adding further diversity to the dataset. The measured dataset is available in [28].

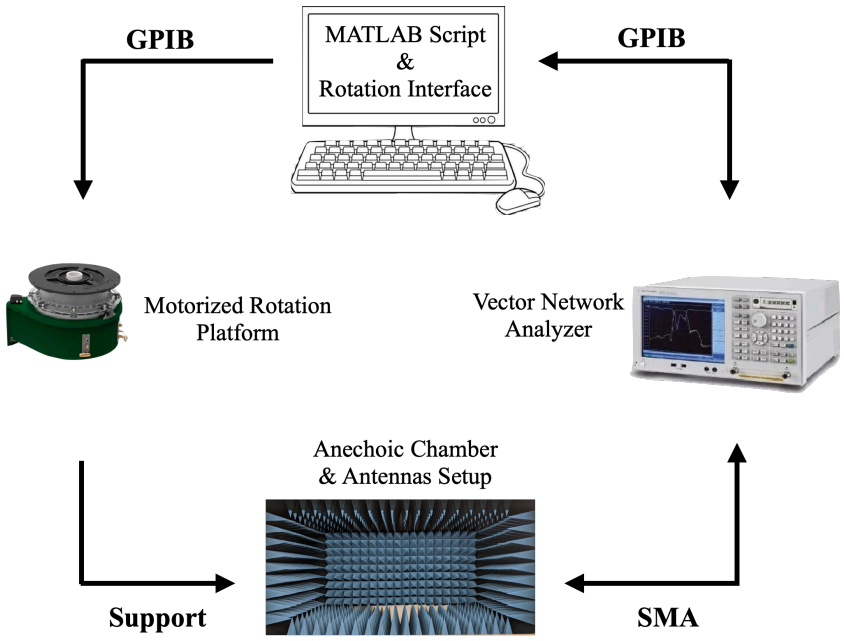


Figure 3.4: Schematic overview of the RF experimental setup: a computer, through GPIB, controls a motorized rotation platform for precise angular positioning, and interfaces with a vector network analyzer (VNA) which is connected to the antennas in the anechoic chamber via SMA cables.

Figure 3.4 displays the components of the experimental setup: an anechoic chamber for a controlled antenna environment, a motorized rotation platform for precise antenna orientation, an Agilent Technologies Vector Network Analyzer (VNA) model E5071B for conducting measurements, and dedicated software scripts for environment setup and parameter configuration. High-quality RF cables were used in the measurements, with a constant 10 dB total loss being accounted for to ensure accurate results.

The position of the mobile source is determined via commands from a rotation interface that controls the high-precision motorized platform. RF parameters are configured and gathered through the VNA, which is interfaced with the MATLAB script-driven computer and connected to the antennas within the anechoic chamber. This configuration guarantees synchronized data collection and allows for accurate measurement of antenna characteristics, including radiation patterns and signal propagation, across different source orientations.

Figure 3.5 depicts the antenna arrangement within the anechoic chamber, designed to ensure a reflection-free environment for RF testing. As per the RF environment setup specified in Fig. 3.2, the source antenna is mounted on a wooden support attached to a motorized rotation platform. This setup is arranged to adjust the antenna's distance (d_s) from a fixed point, referred to as the origin, and to rotate it through a full circle in steps of 10 degrees, covering every angle (θ_s) from 0 to 360 degrees. In parallel, the receiver antenna is sequentially placed at each of the four designated locations, mounted on a plastic support and equally distanced from the origin by a radius (d_r), providing uniform measurements in all directions. Both wood and plastic supports are chosen for their non-interfering properties with RF signals.

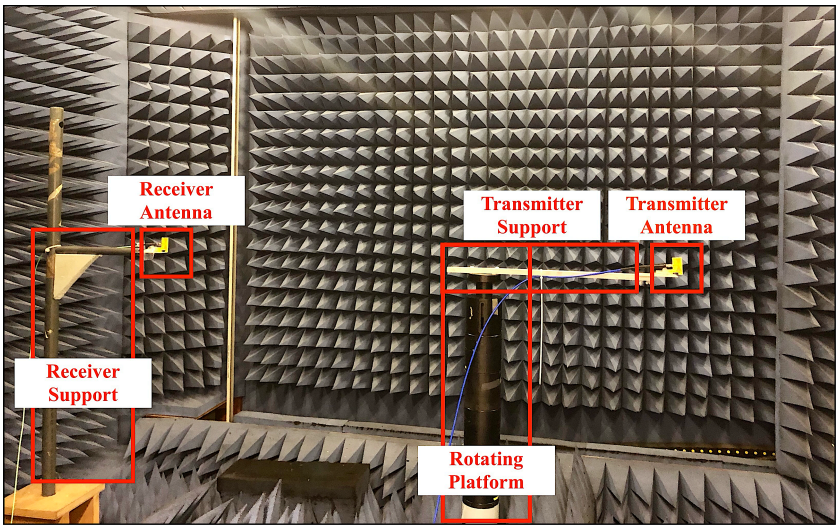


Figure 3.5: Anechoic chamber configuration: source and receiver antennas mounted on supports with the motorized rotation platform for antenna orientation. Here, the source is positioned at $\theta_s = 180°$.

Both transmission and reception leverage identical bow-tie antennas, depicted in Fig. 3.6. Characterized by their efficient dipole design, these antennas span 0.035 meters in length and 0.015 meters in width, ensuring omnidirectional signal radiation with a gain of 1.5 dB. The selection of bow-tie antennas is driven by their wide bandwidth and frequency flexibility, ensuring they perform effectively within the critical range of 2.15 GHz to 2.45 GHz. Furthermore, they are equipped with a 50 Ω impedance, conforming to standard RF system requirements. Within this setup, the source transmits RF signals at 2.4 GHz at a consistent power level of 10 dBm, captured by all four receivers in diverse orientations. To simulate diverse environmental conditions, transmissions and receptions are tested under various noise levels with a signal-to-noise ratio (SNR) set to 0, 10, or 20 dB.

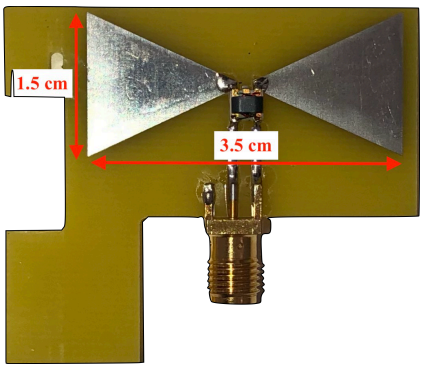


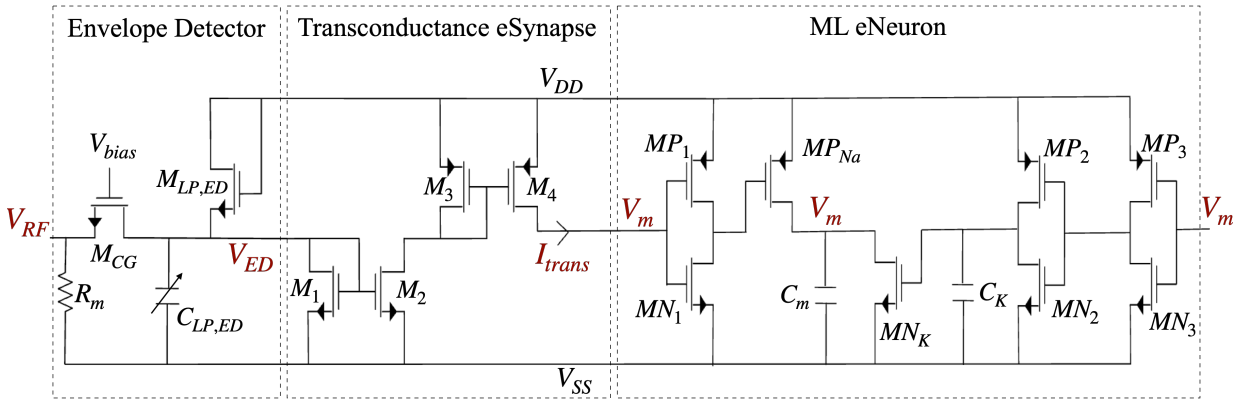
Figure 3.6: Bow-tie dipole antenna used for both transmitter and receiver.

3.3 . Neuromorphic Pre-Processing

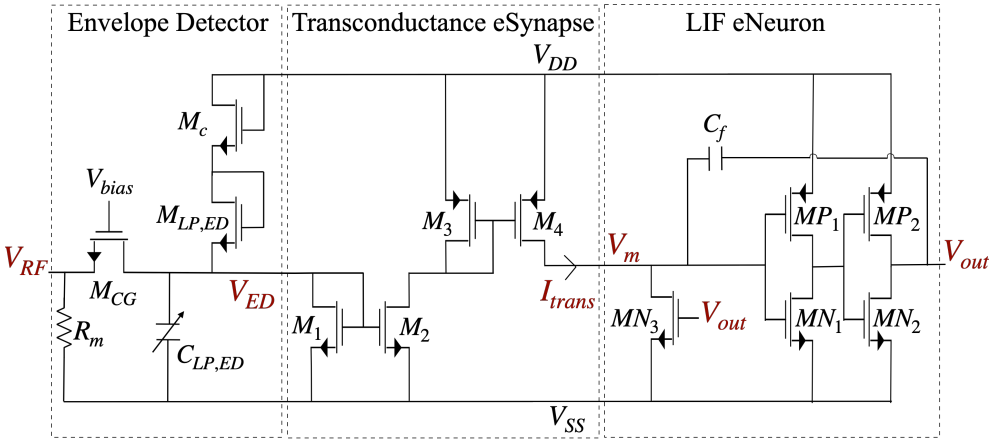
The neuromorphic pre-processing, also defined as neuromorphic-enhanced wake-up radio (NWR), is the third stage of the RF NeuroAS system. Within the scope of this system, NWR's key function is to convert RF signals into spike trains. The rate of these spikes, or the spiking frequency (f_{spike}), correlates with the RF signal's power level, which is determined by the dataset features from the previous stage of the RF NeuroAS system. This rate serves as the input for the next stage of the system, the analog spiking neural network. Furthermore, the NWR can detect and decode bit patterns from the RF signal modulated using On-Off Keying (OOK), with these bits being determined based on the associated spiking frequency.

As depicted in Fig. 3.1, the NWR consists of an envelope detector equipped with a matching network, a transconductance eSynapse, and a biomimetic eNeuron. The RF signal V_{RF} is captured and demodulated by the envelope detector into V_{ED} signal. This latter is then transformed into synaptic current by the transconductance eSynapse. This current activates the eNeuron, triggering it to produce spikes (or action potentials) at a frequency f_{spike}. Within this work, NWR incorporates two types of eNeurons for comparison: the Morris-Lecar (ML) eNeuron from [156], which closely simulates biological neuron behavior as discussed in Sec. 2.3.2, and the Leaky Integrate-and-Fire (LIF) eNeuron from [155], which offers a simplified, less biologically detailed model, also described in Sec. 2.3.2. The performance differences between the ML-based and LIF-based NWR approaches are explored to assess their effectiveness. For the RF NeuroAS system, the ML-based NWR is selected to maintain biomimetic fidelity.

The neuromorphic pre-processing circuit is fully designed using Cadence Virtuoso, a leading software for design kits. This design uses the BiCMOS 55 nm technology from STMicroelectronics, a cost-effective RF solution featuring low-power, low-threshold MOS transistors suited for microwave applications. A detailed description of this technology is presented in Appendix C.



(a)



(b)

Figure 3.7: Circuit level of the neuromorphic pre-processing stage (NWR) composed of an envelope detector, a transconductance eSynapse, and an eNeuron (ML or LIF) (a) ML-based NWR (V_{DD} = 100 mV and V_{SS} = -100 mV), and (b) LIF-based NWR (V_{DD} = 200 mV and V_{SS} = 0 mV).

3.3.1 . Circuit-Level Design

Figure 3.7 presents the circuit-level design of the NWR stage within the RF NeuroAS system. The circuit of the ML-based NWR is depicted in Fig. 3.7(a), while the LIF-based NWR circuit is shown in Fig. 3.7(b). All transistors in these circuits are operating in the subthreshold region offering two significant benefits. Firstly, this operation region enables ultra-low power consumption due to the use of very low supply voltages (in the order of hundreds of millivolts) (See Appendix C). Secondly, in the eNeuron circuit, the requirement for a non-linear functions model is efficiently met when transistors operate in the subthreshold region [153]. The ML-based NWR operates within supply voltages of V_{SS} = -100 mV and V_{DD} = 100 mV, while the LIF-based NWR operates between V_{SS} = 0 V and V_{DD} = 200 mV, according to the specific requirements

of the respective eNeuron models.

A .Envelope Detector

The envelope detector component includes a matching network, the envelope detector itself, and a low-pass filter for both circuit configurations (ML-based and LIF-based NWR). It is optimized for ultra-low power consumption within the neuromorphic pre-processing stage. For this purpose, a passive circuit featuring a metallic resistance ($R_m = 50\ \Omega$) with minimal parasitic capacitance is used to achieve low-cost on-chip impedance matching. However, this circuit may lead to an increased input noise, which could potentially affect the envelope detector's sensitivity to weak signals. Despite this, the eNeuron's design can compensate for this drawback, as it has been proven that the eNeuron is robust to external noise, outlined later in Sec. 4.2.

Conventionally, the RF envelope detector converts the incoming RF signal (V_{RF}) into a baseband output (V_{ED}) that mirrors the envelope of the original signal. The main characteristic of the envelope detector is its conversion gain (CG_{ED}), which is determined by the ratio of the demodulated output signal to the RF input signal. The literature outlines three well-known envelope detector designs: common drain, common source, and common gate. The NWR uses a common gate transistor configuration (M_{CG}), for its capacity to reach the highest conversion gain, followed by a low pass filter.

In the analysis of small signals using a common gate configuration, the conversion gain of the envelope detector is defined as

$$CG_{ED} = \frac{V_{ED}}{V_{RF}} = \frac{i_o r_o}{V_{RF}} = \frac{I_D r_o V_{RF}}{4\phi_T^2}, \tag{3.3}$$

where r_o is the intrinsic output impedance; i_o is the demodulated output current represented by the second-order term of Taylor series expansion as

$$i_o = \left(\frac{\partial^2 I_D}{\partial V_S^2}\right)\frac{V_S^2}{2}. \tag{3.4}$$

The drain current I_D is expressed in the weak inversion region as

$$I_D = I_S \cdot e^{(V_G - V_{T0})/\eta\phi_T}(e^{-V_S/\phi_T} - e^{-V_D/\phi_T}), \tag{3.5}$$

where I_S is the specific current; V_{T0} is the bias-independent threshold voltage for $V_S = 0$; η is the subthreshold slope factor ($\eta \approx 1.34$ for the 55nm BiCMOS technology, as presented in Appendix C); ϕ_T is the thermal voltage ($kT/q \approx 26\ mV$ at 27 °C); and V_G, V_S, V_D are the voltages at the gate, source, and drain of the transistor, respectively.

To design the common gate transistor, it is important to analyze its conversion gain relative to its operating region. One may design the common gate by identifying the operating region where the conversion gain reaches its optimum. Figure 3.8 presents the variations of the conversion gain of the envelope detector against the corresponding g_m/I_D of the common gate transistor. This figure features two distinct curves: a blue line that indicates the theoretical conversion gain calculated from (3.3), and orange star markers that reflect the conversion gain results obtained from periodic steady state (PSS) simulations. The analysis reveals the peak conversion gain,

CG_{ED}, at $g_m/I_D \approx 26\ 1/V$, suggesting an operation in the weak inversion region (see Appendix C for details about the g_m/I_D ratio and the operating region of a transistor). Based on this result, M_{CG} is designed to operate on this specified region, using an external bias voltage V_{bias} = 300 mV, with dimensions L = 60 nm and W = 130 nm.

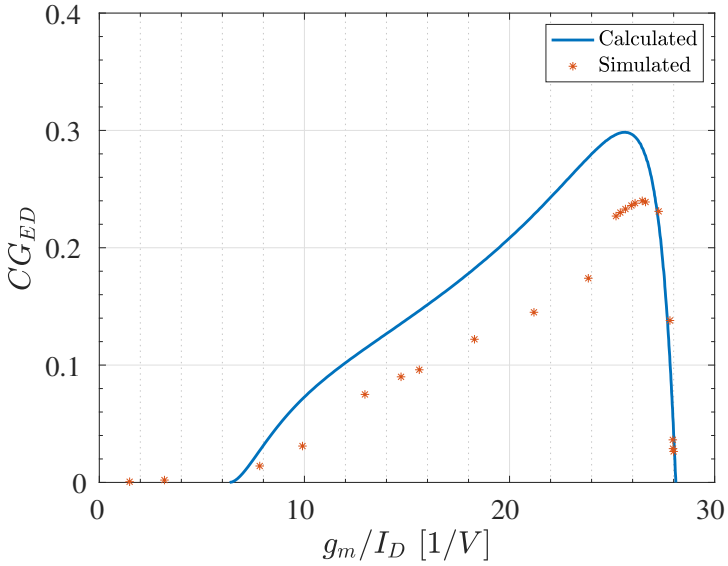


Figure 3.8: Conversion gain of the envelope detector relative to g_m/I_D in a common gate transistor configuration; the blue line represents theoretical values calculated in (3.3), while the red line shows PSS simulations result.

A low-pass filter follows the envelope detector to eliminate high-frequency components, consisting of a capacitance $C_{LP,ED}$ and a diode-connected transistor $M_{LP,ED}$. To minimize the silicon footprint, the selected capacitance is the smallest varicap-based available in the BiCMOS 55 nm technology. As shown in Fig. 3.7(b), the LIF-based NWR design incorporates an additional diode-connected transistor M_c in cascode with $M_{LP,ED}$. The reason behind that is to maintain the stability of the demodulated voltage, since the supply voltages implemented for both circuits, ML-based NWR and LIF-based NWR, are not the same. The component dimensions are detailed in Appendix D.

B .Transconductance eSynapse

The transconductance eSynapse serves as a bridge connecting the RF components to the neural elements within the NWR stage of the RF NeuroAS system. It converts the demodulated voltage (V_{ED}) at the output of the envelope detector into a synaptic excitatory current (I_{trans}), with a higher output impedance. This current stimulates the subsequent eNeuron differently, based on various power levels of the RF input signals. As shown in Fig. 3.7, it is composed of two current mirrors, one NMOS and one PMOS (M_1 to M_4). However, the sizing of its components varies depending on whether the NWR is ML-based or LIF-based, with specific dimensions detailed in Appendix D.

C . eNeuron

As the NWR stage includes two types of eNeurons, ML and LIF, their effects on system performance vary, especially regarding silicon area, power consumption, dynamic range, and sensitivity to signal detection. Here is a brief description of each eNeuron circuit. Details on the eNeurons sizing are presented in Appendix D.

ML eNeuron, redesigned from [156], operates at a high-firing rate to enhance the dynamic range of the system. A detailed description of the model and equations related to this biomimetic eNeuron are provided in Sec. 2.3.2. Figure 3.7(a) illustrates its design at the transistor level. Briefly, when an input synaptic current is applied, the membrane capacitance C_m is charged through MP_{Na} and discharged through MN_K. This leads to a large yet brief change in membrane potential V_m, which is referred to as action potentials (spikes). This process, facilitated by MP_{Na} and MN_K transistors, emulates the natural ion exchange of Na and K across neuronal cell membranes during brain activity. Additionally, a negative feedback loop is established by two cascaded inverters, MP_2/MN_2 and MP_3/M_3, with MN_K, while a positive feedback loop is created by inverter MP_1/MN_1 with MP_{Na}.

LIF eNeuron, redesigned from [155], features high energy efficiency with a considerable firing rate. A detailed description of the model and equations related to this simplified eNeuron are provided in Sec. 2.3.2. Figure 3.7(b) illustrates its design at the transistor level. The design uses parasitic capacitances as the eNeuron's membrane capacitance, which can achieve low power consumption with low silicon area. Briefly, once an input synaptic current is delivered, it is integrated by the feedback capacitance C_f (charging mechanism), causing a gradual increase in the membrane voltage V_m of the eNeuron. Once V_m hits a certain threshold level, the inverters (MN_1/MP_1, and MN_2/MP_2) are activated, and output volatge V_{out} rises then to V_{DD}. Concurrently, C_f discharges through MN_3 (discharging mechanism), reducing V_m and causing the inverters to switch again, which brings V_{out} back down to V_{SS}.

3.4 . Analog Spiking Neural Network

Following the neuromorphic pre-processing stage, as depicted in Fig. 3.1, the final stage of the RF NeuroAS system introduces the analog spiking neural network (A-SNN). Its initial objective is to accurately predict the position of the source in the plane, specifying its coordinates (θ_s and d_s) with 10-degree angular resolution. Additionally, the second objective is to develop a neural network that integrates the functional properties and design constraints of the analog spiking eNeurons from a physical format into a software model. This critical step is essential for the complete hardware design of the A-SNN, promising enhancements in both speed and energy efficiency.

3.4.1 . Neural Network Structure

Figure 3.9 depicts the neural network's structure, which includes an input layer (i), three hidden layers (h_1, h_2, h_3), and an output layer (o) of ML eNeurons. This eNeuron type is selected for its biomimetic behavior and its broad dynamic range that enhances learning accuracy [23]. The input layer of the network consists of four eNeurons from the neuromorphic pre-processing

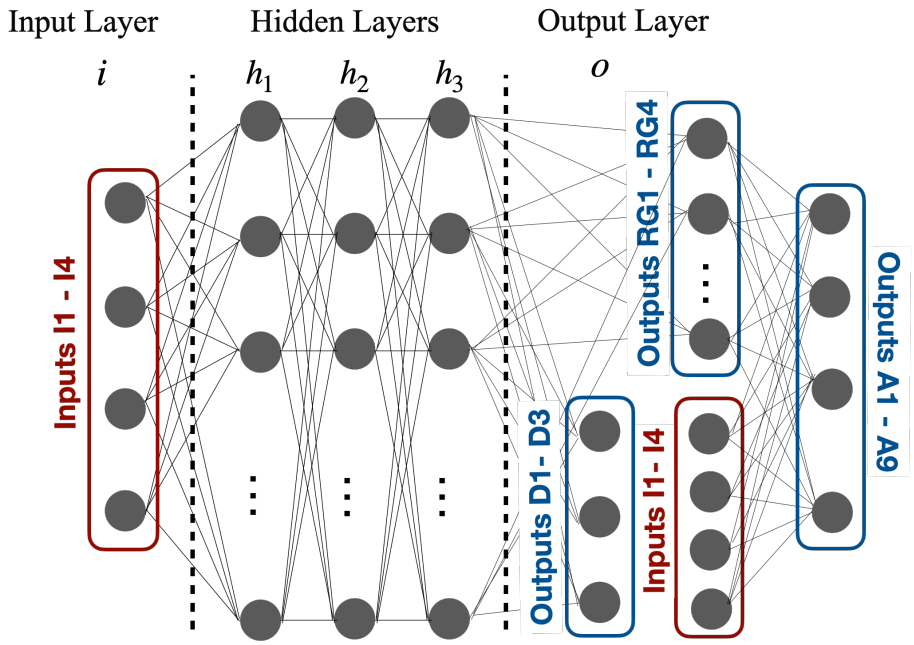


Figure 3.9: Illustration of the analog spiking neural network (A-SNN) for 10-degree angular resolution, composed of an input layer, three hidden layers, and an output layer.

stage, each providing pre-processed data from one of the four receivers. All three hidden layers feature a fully connected design, meaning each neuron in one layer connects to every neuron in the next. The depth and the number of nodes are minimized, due to the requirements for low power usage and hardware compatibility. Consequently, each hidden layer consists of merely 12 neurons.

The output layer (o) of this neural network structure accurately identifies the source's location in terms of angle and distance. For distance prediction, output neurons D1-D3 categorize the three specific distance classes from the origin, as previously established in Sec. 3.1. For angle prediction, a classification approach is also used, typically requiring one output neuron per category. Given the RF configuration scenario, where the source moves on a 360-degree trajectory with 10-degree increments, this requires 36 neurons to accurately represent each possible angular position.

To streamline the network and decrease the number of neurons in each layer, an alternative classification approach is introduced, as explained in Sec. 3.2. This method divides the entire 360-degree space into four regions, labeled as r_{s1}-r_{s4}. Through output neurons RG1-RG4, the network determines the correct region for the source's location, as shown in Fig. 3.9. Following this, the network uses 9 output neurons A1-A9 to classify the angle within the chosen region into one of nine categories, a_{s1}-a_{s9}, maintaining a 10-degree resolution. For precise angle prediction, the network uses a concatenation of input features and output regions.

3.4.2 . Learning Technique

Training and testing the neural network requires several elements: the choice of the platform, the neuron model, the activation function, the datasets, and the hyperparameters. The A-SNN, within the RF NeuroAS system, is trained and tested through a specific deep learning software technique, using the TensorFlow machine learning framework [68]. It uses the same backpropagation learning technique as traditional ANN training within TensorFlow [7]. Besides, it adopts an innovative analog-based deep learning strategy to incorporate the eNeuron model and its analog properties into the training process and weight adjustments. This technique is thoroughly described in Sec. 4.1, where the feasibility analysis and the synthesis framework essential for the design of the A-SNN are explored. Deep learning was chosen over bio-plausible learning for the A-SNN within the RF NeuroAS system due to its superior performance, as demonstrated in Chap. 4 and Chap. 5.

Although Sec. 4.1 provides an extensive description of the developed technique, here is a brief overview of the methodology applied to the A-SNN stage of the RF NeuroAS system. In the A-SNN, the ML eNeuron follows a spiking-rate-based model common in sensory processing systems, as it modulates its spiking rate based on the strength of the input it receives. As the input strength increases, so does the firing rate of the eNeuron. The spiking-rate-based model was preferred over the spiking-time-based model because the latter is more affected by the random noise of transistors, as demonstrated later in Sec. 4.2.

To train and test the A-SNN in TensorFlow, it is essential to develop a custom activation function that takes into consideration the eNeuron model. This activation function is extracted from the post-layout transfer function of the ML eNeuron relative to its dynamic range. It is denoted as h and defined for an eNeuron eN_i as follows

$$f_{spike_i} = h(I_{syn_i}) = h(\sum_{k=1}^{n} g(f_{spike_k})) = h(\sum_{k=1}^{n} w_{syn_{ki}} \cdot f_{spike_k} + b_1). \tag{3.6}$$

Here, the output spike rate f_{spike_i} of eN_i is obtained from its input synaptic current I_{syn_i} through the function h. The input current I_{syn_i} is the sum of incoming synaptic currents, where each one is given by the transfer function $g(f_{spike_k})$ of an eSynapse, that links an eNeuron eN_i to a previous eNeuron eN_k ($\forall\, k \in [1, n]$) having spike rate f_{spike_k}. In this case, this transfer function of eSynapse is simplified to a linear expression, relating synaptic weights $w_{syn_{ki}}$ and spike rates f_{spike_k} from preceding connected eNeurons.

To consider the post-layout activation function in the TensorFlow training procedure, it must be modeled by an appropriate equation that conforms to the properties of an activation function (the non-linearity, the differentiability, and the smooth gradient) [55]. Results for three proposed fits for the post-layout activation function are shown in Sec. 5.1.4. The following study will demonstrate why the polynomial fit is the best option for the activation function model.

Regardless of whether analog characteristics are taken into account, the training and testing phases use the "SimLocRF" and "MeasLocRF" datasets, as outlined in Sec. 3.2. These datasets are crucial for source localization within the specific scenario considered for the RF NeuroAS system.

3.4.3 . Hyperparameters

The adaptive moment estimation (Adam) optimizer is a good candidate for the neural network training phase, with the sigmoid activation function. If one considers analog spike characteristics and post-layout activation function during the training phase, then the Adam optimizer can lead to a training drop due to the unbounded increase of gradients during weight updates. Thus, the stochastic gradient descent (SGD) optimizer is chosen for the A-SNN training.

Given that the source localization relies on a classification problem, the categorical cross-entropy is identified as the suitable loss function. The learning rate is set at 0.01, with the training phase requiring 100 epochs and a batch size of 32, determined after several training sessions to find the optimal choices. The hyperparameters for the A-SNN configuration are listed in Tab. 3.2.

Table 3.2: Hyperparameters used for training the A-SNN with a 10-degree resolution

Hyperparameters	**Configuration**
Optimizer	SGD
Loss Function	Categorical cross-entropy
Learning rate	0.01
Epoch number	100
Batch size	32

3.4.4 . Performance Evaluation

To evaluate the system performance in the RF source localization problem, accuracy (ACC) serves as a widely adopted metric, especially suitable for classification tasks in neural networks. Accuracy is defined as

$$ACC = N_c/N_f \times 100, \tag{3.7}$$

where N_c represents the count of accurate predictions; N_f denotes the total count of test samples.

Given that the system achieves a 10-degree resolution, the normalized angle error (NAE) is used as an additional performance metric. NAE measures the mean deviation between actual and predicted angles, adjusted for the system's resolution. It is calculated as follows

$$NAE = \frac{1}{N_f \times RE} \cdot \sum_{i=1}^{N_f} |\theta_{s_i} - \hat{\theta}_{s_i}|, \tag{3.8}$$

where the error is adjusted based on the system's resolution (RE), set as 10 degrees in this scenario. The actual angle θ_s is calculated as $r_s \times n_r + a_s$, where r_s represents the actual region; n_r is the count of regional range; a_s indicates the exact angle within that region. For the setup within the RF NeuroAS system, the space is divided into four regions ($r_s = 0, 1, 2, 3$), with each region covering 90 degrees ($n_r = 90$). Further details on these configurations are provided in Sec. 3.2. Concurrently, $\hat{\theta}_s$ is the angle estimated by the A-SNN, formulated as $\hat{r_s} \times n_r + \hat{a_s}$, where $\hat{r_s}$ is the predicted region; $\hat{a_s}$ is the predicted angle within that $\hat{r_s}$.

3.5 . Simplified RF NeuroAS System: A Fully Analog Circuit Design

A streamlined version of the RF NeuroAS system is designed entirely on an analog circuit to demonstrate the practicality of hardware-based RF NeuroAS implementations. Unlike conventional source localization methods that rely on complex signal processing and artificial intelligence algorithms, requiring extensive computational resources, this simplified RF NeuroAS system presents an innovative alternative. It introduces an energy-efficient solution with reduced computational complexity, based on the neuromorphic approach. This setup enables real-time detection of RF sources with significantly reduced power consumption.

Figure 3.10 depicts the architecture of the simplified RF NeuroAS system, designed in the BiCMOS SiGe 55 nm technology. Similar to the RF NeuroAS system depicted in Fig. 3.1, the simplified version comprises an RF environment setup, neuromorphic pre-processing, and neuromorphic computing stages. The simplification of the RF NeuroAS system involves three key modifications:

1. To reduce the complexity of the design, the configuration of the simplified RF NeuroAS system features a mobile source positioned solely on the positive part of the 2D plane and uses two receivers instead of four.

2. The neuromorphic pre-processing stage adopts a LIF-based NWR configuration (illustrated previously in Fig. 3.7(b)), chosen for the LIF eNeuron's advantages in power efficiency and smaller footprint compared to the ML eNeuron.

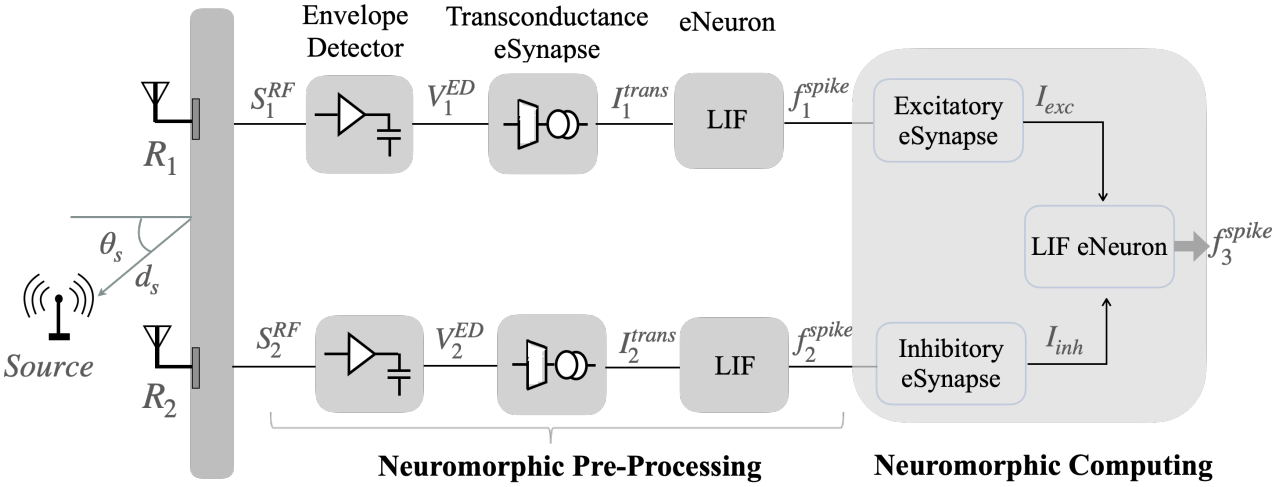


Figure 3.10: Illustration of the simplified RF NeuroAS system proposed in a fully analog circuit design. It focuses on RF source localization through a spike rate-based mechanism using data from two receivers. It is mainly composed of neuromorphic pre-processing and neuromorphic computing stages.

3. Replacing the multi-layered deep neural network in the neuromorphic computing stage, this simplified version features a two-layer spiking neural network, incorporating three LIF eNeurons and two bifunctional eSynapses (one excitatory and one inhibitory).

3.5.1 . Simplified RF NeuroAS Functionality

The objective of the simplified RF NeuroAS system is to identify the position of the mobile source, determined by an angle θ_s and a distance d_s from the origin of the plane. As depicted in Fig. 3.10, the system uses a spike rate-based approach to deduce the source position. Here's an outline of how the system operates.

Initially, two distinct 2.4 GHz signals, S_1^{RF} and S_2^{RF}, are transmitted from the source and captured by receivers R_1 and R_2. These signals are then processed by the neuromorphic pre-processing stage, which converts the RF signals into spiking frequencies. In this stage, the envelope detector demodulates the RF signal, and subsequently, the transconductance eSynapse converts this signal into a current that activates the LIF eNeuron. The LIF eNeuron produces spikes in the time domain, with a spiking frequency that varies according to the input power of the received RF signal. As a result, each neuromorphic pre-processing stage outputs a spiking frequency, f_1^{spike} and f_2^{spike}, which correlate with the respective input powers, P_1^{RF} and P_2^{RF}, of the RF signals.

Following this, f_1^{spike} is converted into an excitation current I_{exc} by a bifunctional eSynapse acting as an excitatory element. Thus, it increases the spiking frequency f_3^{spike} of the LIF eNeuron in the output layer. Conversely, f_2^{spike} is converted into an inhibition current I_{inh} by a bifunctional eSynapse acting as an inhibitory element. Thus, it decreases the spiking frequency f_3^{spike} of the subsequent LIF eNeuron. This LIF eNeuron then processes the sum of both currents and produces an f_3^{spike} that mirrors the difference between the two received input power levels ($\Delta P_{RF} = P_2^{RF}$ - P_1^{RF}).

This relationship, along with the established correlation between ΔP_{RF} and the angle of the source θ_s detailed later in Sec. 3.5.2, facilitates the connection between f_3^{spike} and θ_s. Ultimately, the simplified RF NeuroAS system determines the position of the source, denoted by θ_s, using the spike rate f_3^{spike} at a resolution of 1 kHz. This resolution is selected according to the spike observation window, which is established at 1 ms.

3.5.2 . Simplified RF NeuroAS Setup

The RF environment setup for the simplified RF NeuroAS system resembles the complete system (shown in Fig. 3.2), but it uses a mobile source restricted to the positive half of the plane and only two receivers. The distance d_s between the source and the origin varies among three specific distances: 0.1, 0.5, and 1 meter. The source angle θ_s, formed by the source and the horizontal axis of the plane, ranges from 0 to π. The receivers are positioned as two fixed points on the horizontal axis of the plane, equidistant ($d_r = 0.07$ meters) from the origin. Receiver R_1 is located on the positive side of the horizontal axis at a distance d_1 from the source, whereas receiver R_2 lies on the negative side at a distance d_2 from the source.

Just like in the complete RF NeuroAS system, the antennas are assumed to operate in a free space environment, with the power at each receiver determined using (3.1) and (3.2). In the context of the simplified RF NeuroAS system's configuration, and following these equations, the

relationship between the source angle θ_s (in degrees) and the difference in input powers ΔP_{RF} (in dB) is established as follows

$$\Delta P_{RF} = P_2^{RF} - P_1^{RF} = 10 \log_{10} \left(\frac{d_s^2 + d_r^2 - 2 d_s d_r \cos \theta_s}{d_s^2 + d_r^2 + 2 d_s d_r \cos \theta_s} \right). \tag{3.9}$$

If the source lies on the vertical axis ($\theta_s = \pi/2$), the distance to both receivers is identical ($d_1 = d_2$), resulting in equal power levels received by each ($P_1^{RF} = P_2^{RF}$). When the source is on the positive side of the horizontal axis ($\theta_s = 0$), R_1 is nearer to the source compared to R_2, leading to R_1 receiving a higher input power than R_2 (ΔP_{RF} is negative). The opposite occurs when the source is on the negative side of the horizontal axis ($\theta_s = \pi$), leading to R_2 receiving a higher input power than R_1 (ΔP_{RF} is positive).

With the system's resolution set at a spiking frequency of 1 kHz, the simplified RF NeuroAS can detect a minimum power difference ΔP_{RF} of 1 dB. Consequently, this requirement imposes a specific limitation on the system's configuration, necessitating that d_1 must not exceed 1.12 x d_2, following (3.9).

3.5.3 . Bifunctional eSynapse

Excitatory and inhibitory synapses are among the most common types in the brain. In spike rate-based neural networks, a synapse typically conveys the spiking membrane voltage from a pre-neuron and generates a current to a post-neuron. When the synapse acts as a current source (excitatory), it injects a positive excitation current into the post-neuron, leading to an increased spiking frequency. Conversely, the synapse serves as a current sink (inhibitory) by supplying a negative inhibition current to the post-neuron, causing a reduction in the post-neuron's spiking frequency.

The bifunctional eSynapse is illustrated in Fig. 3.11, featuring both excitation and inhibition.

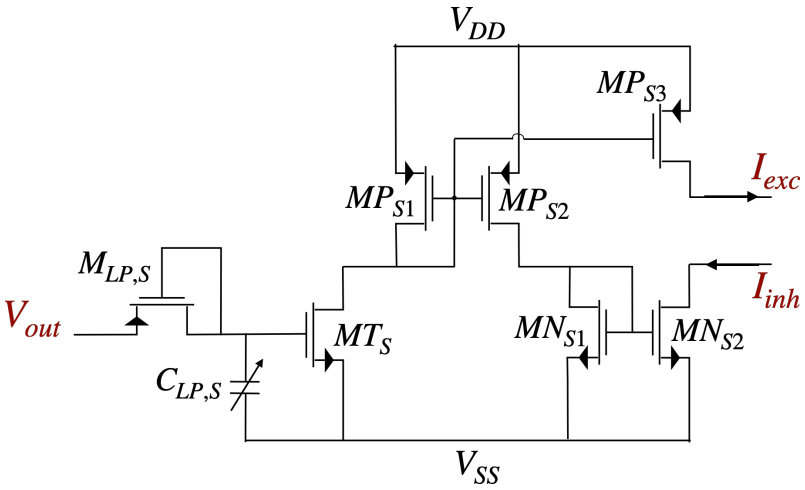


Figure 3.11: Circuit level of the bifunctional eSynapse in the simplified RF NeuroAS system, featuring dual functions: excitatory and inhibitory (V_{DD} = 200 mV and V_{SS} = 0 V).

This eSynapse consists of an RC filter, a transconductance component, and two sets of current mirrors. The RC filter is composed of a diode-connected transistor $M_{LP,S}$ and a capacitor $C_{LP,S}$ and captures spike rate information from the output voltage (V_{out}) of the pre-neuron. This information is then converted into a current by the transconductance transistor $M_{T,S}$. Excitatory action is generated by a PMOS current mirror, comprising MP_{S1} and MP_{S2}, which outputs the synaptic current I_{exc} via MP_{S3}. On the other hand, inhibition occurs through an NMOS current mirror with MN_{S1} and MN_{S2}, which produces the inhibited current I_{inh}.

In the simplified RF NeuroAS system, as depicted in Fig. 3.10, each eSynapse neglects either its excitatory or inhibitory capability, which might result in increased power consumption without a benefit in this specific application. Nevertheless, such eSynapse with dual functionality may be useful for a larger neural network, where a single eNeuron connects to several others through eSynapses that fulfill both excitatory and inhibitory functions.

3.6 . Conclusion

Chapter 3 introduced an innovative analog spike-based neuromorphic system (RF NeuroAS) designed for precise and energy-efficient RF source localization. It outlined the RF environment setup of the system, the process of data extraction involving the generation of both simulated and measured datasets, the pre-processing, and the design of the neural network. This neural network elaborated the training and testing phases with these datasets on the TensorFlow platform, and incorporated the analog functions of spiking eNeurons. A comprehensive discussion on the analog spiking neural network, its feasibility, and its learning methodology will be detailed in the following Chap. 4. Chapter 3 concluded with the presentation of a fully analog circuit design for a streamlined version of the RF NeuroAS system, using BiCMOS 55 nm technology. The results that validate the proposal from Chap. 3 are presented in Chap. 5.

Chapter 4

Feasibility Study of Analog Spiking Neural Networks: From Spiking eNeurons to Learning Techniques

In the previous Chap. 3, an RF analog spike-based neuromorphic system (RF NeuroAS) was proposed and designed for accurate and energy-efficient RF source localization in low-power IoT devices. A crucial component of this system is the analog spiking neural network (A-SNN), which consists of interconnected layers of CMOS analog spiking neurons (eNeurons) and synapses (eSynapses). The adoption of A-SNN, including eNeurons and eSynapses, aims to enhance the energy efficiency of the RF NeuroAS system by leveraging spike coding behavior and analog circuit design. Recent research focus has led to highly optimized eNeurons and eSynapses circuits [145, 153, 155], with lower interest presented in global network properties. However, implementing the A-SNN requires a comprehensive examination of the hardware design feasibility of such a network. Like any neural network, the A-SNN necessitates an appropriate learning technique that considers the models and functions of its main components (eNeurons and eSynapses), and adjusts its algorithm accordingly for effective training and testing.

Since the 90s, with the emergence of Industry 4.0 and the Internet of Things (IoT), **deep learning** has been a promising learning technique in cutting-edge research, dominating software-based neural networks and artificial intelligence. More details on this learning method can be found in Sec. 2.2. As outlined in Sec. 2.2.2, deep neural networks (DNNs) are architectures with multiple hidden layers that use deep learning techniques to address complex tasks and demonstrate advanced processing capabilities. These networks are trained and tested using software frameworks like TensorFlow, known for their flexibility and extensive library support [68]. However, their reliance on cloud computing and the von Neumann architecture makes them energy-intensive and introduces challenges in hardware integration, complicating their use in low-power devices.

Within recent advancements in AI, neuromorphic computing has appeared as an exciting alternative to von Neumann architecture, providing edge computing solutions optimized for ultra-low-power applications. Spiking neural networks (SNNs) stand as the most prevalent models in neuromorphic computing by enhancing energy efficiency through spike coding, as detailed in Sec. 2.3. Numerous hardware implementations for SNNs, particularly on digital processors, are reported in the literature and typically trained and tested using Brian 2 simulator [182]. A widely used learning method for SNNs is the biologically inspired **time-based learning** technique. Spike-timing-dependent plasticity (STDP) stands out as a prominent example within this approach. Further information on this learning technique is provided in Sec. 2.3.3.

To the best of knowledge, no detailed studies have been conducted on the feasibility of analog spiking neural networks or on suitable learning techniques for eNeurons and eSynapses models. This chapter is dedicated to explore the feasibility of A-SNN, from its fundamental components to

potential learning tools. It assesses both **deep learning** and **time-based learning** techniques on A-SNN to identify an effective learning approach for neural networks designed on analog integrated circuits. Initially, Sec. 4.1 focuses on the application of **deep learning** technique to A-SNN to bridge the gap between hardware and software AI. Subsequently, Sec. 4.2 addresses the application of **time-based learning** to A-SNN, particularly focusing on the STDP rule, to bridge the gap between hardware AI and biological neural systems.

4.1 . Deep Learning Approach for A-SNN

Deep learning stands out in software AI research, primarily leading to the development of deep neural networks (DNNs). DNNs are known for their robust performance and ability to tackle complex challenges, as they rely on non-linear functions and multiple hidden layers [46, 214, 215]. Their straightforward layered architecture, connecting neurons only to those in subsequent layers, simplifies their structure compared to alternative network models. On the other hand, neural networks based on analog spiking eNeurons, referred here as A-SNNs, gain attention for their remarkable energy efficiency. A-SNN, like DNN, relies on non-linear eNeuron models that align with standards set by software AI research. This raises the pivotal question of whether deep learning techniques, which have proven effective for DNNs, are applicable and effective on A-SNNs. Addressing this question is essential to implement an A-SNN that fully benefits from both the strengths of deep learning and the energy efficiency offered by analog spike behavior.

This section aims to investigate the feasibility and the synthesis of A-SNN with deep learning, by conducting a comparative analysis between DNN and A-SNN under the application of deep learning technique. Initially, it explores the fundamental components of both DNN and A-SNN, highlighting various eNeuron and eSynapse models examined in this study (Sec. 4.1.1). It then conducts a feasibility analysis of applying deep learning to A-SNN, focusing on their unique transfer functions and capabilities compared to DNNs (Sec. 4.1.2). Following this, the section presents a synthesis framework for integrating deep learning into A-SNN (Sec. 4.1.3). This framework is showcased through solving the MNIST problem (Sec. 4.1.4), as a demonstration of A-SNN's performance when applying deep learning technique.

4.1.1 . Neuron Models Compared: DNN and A-SNN

Figure 4.1 illustrates the neuron models used in DNN and A-SNN for the deep learning purpose, depicted in Fig. 4.1(a) and 4.1(b) respectively. In the context of a DNN, the neuron model, often called an artificial neuron or node, serves as a basic computational unit that simplifies the principal functions of a biological neuron. As shown in Fig. 4.1(a), an artificial neuron, labeled as a_i, performs mathematical functions on inputs from the neurons in the previous layer, labeled a_1, a_k to a_n. These inputs are adjusted by a set of synaptic weight parameters w_{1i}, w_{ki} through w_{ni}, abstracting the role of biological synapses. The neuron then calculates a weighted input sum, denoted as z_i, which is then processed through a non-linear activation function, f, to generate an output, x_i. The straightforward design of the artificial neuron contributes to its dominance in software-based neural networks. Detailed equations for this neuron model are discussed later, in Sec. 4.1.2.

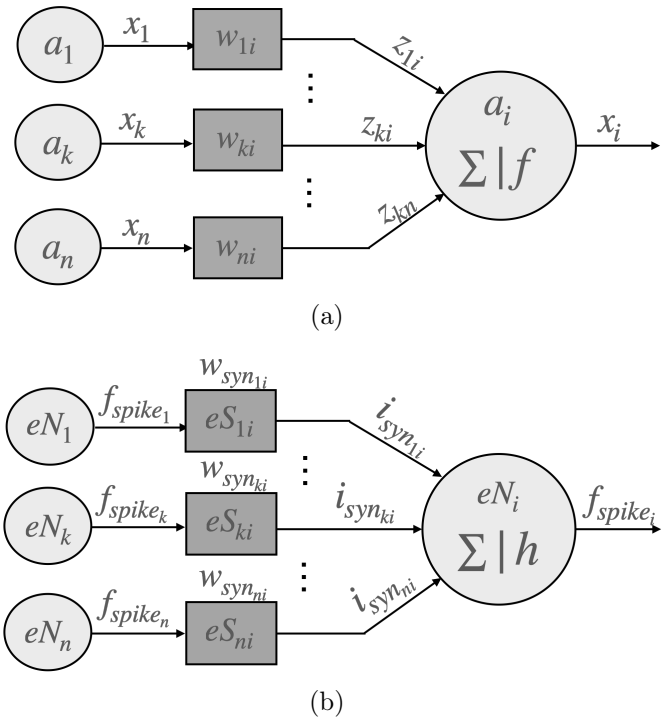


Figure 4.1: Illustration of (a) the artificial neuron model used in a deep neural network (DNN) and (b) the spike-rate-based eNeuron model within an analog spiking neural network (A-SNN), with deep learning technique being applied to each.

In the context of A-SNN, the neuron model diverges from the artificial neuron of DNNs, functioning not as a basic node but as an analog spiking neuron (eNeuron) that closely emulates biological neuron behaviors. Unlike the continuous output of artificial neurons, eNeurons generate discrete spikes upon reaching a specific membrane potential threshold.

For the purpose of studying deep learning applicability in A-SNN, eNeurons process information through the rate of spike occurrences, not the timing of individual spikes. This means that the information processing relies on the rate at which spikes occur in response to input currents. The reason behind that is to derive from the transfer function of the spike-rate-based eNeuron model — which connects input current to output spike rate — the non-linear function crucial for deep learning. As illustrated in Fig. 4.1(b), an eNeuron eN_i receives the synaptic currents sum from preceding layer eNeurons (eN_1, eN_k to eN_n) via eSynapses (eS_{1i}, eS_{ki} to eS_{ni}). These eSynapses adjust their weights, labeled as $w_{syn_{1i}}$, $w_{syn_{ki}}$ to $w_{syn_{ni}}$, to translate spike frequencies f_{spike_1}, f_{spike_k} to f_{spike_n} into synaptic currents $i_{syn_{1i}}$, $i_{syn_{ki}}$ to $i_{syn_{ni}}$. Consequently, the eNeuron eN_i, activated by its synaptic input, generates spikes at f_{spike_i} through the activation function h. More details on this eNeuron model's equations are provided in Sec. 4.1.2.

The eNeuron relies on spike-rate-based model, proposed in Fig. 4.1(b). The study of deep learning applicability within A-SNN incorporates three distinct types of eNeurons. The types explored are the biomimetic Morris Lecar (b-ML) eNeuron redesigned from [156], known for its

biological accuracy, alongside its simplified version (s-ML) redesigned from [153], and the Axon Hillock-based Leaky Integrate and Fire (AH-LIF) eNeuron redesigned from [155], which offers a less biologically detailed approach. Details on their circuit designs and the spiking mechanism equations are outlined in Sec. 2.3.2, with a comparative analysis of energy efficiency, spiking frequency range, and size presented in Tab. 2.2. Additionally, these types differ in their transfer functions, which define the relationship between input currents and the resulting output spike rates. In this context of applying deep learning within A-SNN, where spike rate is the principal form of information, these diverse transfer functions lead to different learning performance of A-SNN.

The eSynapse selected for the deep learning analysis of A-SNN is the excitatory eSynapse detailed in Sec. 2.3.2. This choice aligns with the spike-rate-based eNeuron model used for this study, prioritizing spike rate over spike timing for weight adjustments. Essentially, it comprises an RC filter and a transconductance to extract the rate of the input set of spikes from an eNeuron and convert it into an excitatory current. This current is then adjusted by the weight of this eSynapse, determined by the gain in its current mirror design. This relation, mapping input spike rate to output synaptic current, defines the transfer function of the excitatory eSynapse. The transfer function equations for both the eNeuron and eSynapse in spike-rate-based models are fundamental for the deep learning feasibility study on A-SNN, detailed in the next Sec. 4.1.2.

4.1.2 . Deep Learning Feasibility Analysis Compared: DNN and A-SNN

A .Analysis on DNN

An example architecture of the DNN is depicted in Fig. 4.2, illustrating L layers of interconnected artificial neurons modeled as shown in Fig. 4.1(a). Within this structure, each neuron is connected only to the neurons in the subsequent layer. In the construction of a DNN, each neuron a_i within a given layer is characterized by an input z_i and an output x_i. This output is determined by the formula $x_i = f_i(z_i)$, where f_i represents the activation function assigned to the neuron a_i. Thus, the generic expression for DNNs can be obtained as follows

$$x_i = f_i(z_i) = f_i(\sum_{k=1}^{n} z_{ki}) = f_i(\sum_{k=1}^{n} w_{ki} \cdot x_k + b_1), \tag{4.1}$$

where z_i denotes the sum of products of the inputs x_k from previous layer neurons a_k ($\forall; k \in [1, n]$) and their corresponding weights w_{ki}, and b_1 represents a constant bias term.

Extending this relationship to a layer-wide perspective, let's consider a p^{th} layer ($\forall\ p \in [1, L]$), comprising n neurons. The input vector Z_p is considered as the collective inputs to this layer, which are the inputs for its n neurons. Similarly, the output vector X_p is considered as the collective outputs from these neurons. Based on these definitions, the input Z_p for layer p is determined by applying a linear combination to the output X_{p-1} of the previous layer, expressed as $Z_p = W_p \cdot X_{p-1}$. Here, W_p denotes the weight matrix that links the p^{th} layer with the $(p-1)^{th}$ layer. Consequently, the output X_p of layer p is derived from its input Z_p via the activation function $F_p(Z_p)$. The formulations for Z_p and X_p are detailed below

$$\begin{aligned} Zp &= (z_{p1}, z_{p2}, \ldots, z_{pn})^T = W_p \cdot X_{p-1}, \\ Xp &= (x_{p1}, x_{p2}, \ldots, x_{pn})^T = (f_{p1}(z_{p1}), f_{p2}(z_{p2}), \ldots, f_{pn}(z_{pn}))^T = F_p(Z_p). \end{aligned} \tag{4.2}$$

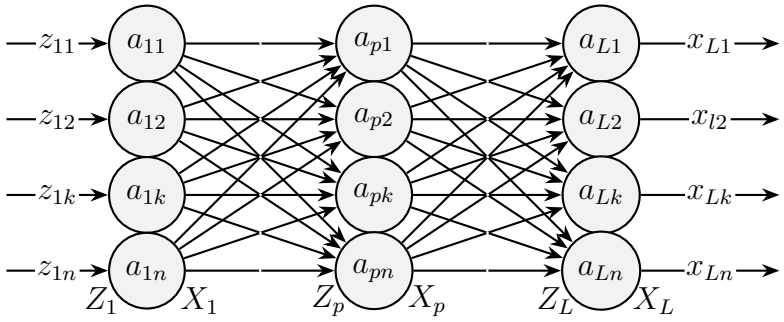


Figure 4.2: Illustration of a deep neural network (DNN), showing layered connectivity using the neuron model defined in Fig. 4.1(a).

Combining the aformentioned equations for the input and output vectors, Z_p and X_p at the p^{th} layer, reveals a fundamental relationship as $X_p = F_p(W_p \cdot X_{p-1})$. The activation function, F_p, commonly represents a non-linear activation function, typically chosen among sigmoid, tangent hyperbolic ($tanh$), or rectified linear unit ($ReLU$) functions [55]. The non-linearity properties of F_p are pivotal to the deep learning capabilities of the network, comprising multiple hidden layers. It is this non-linear nature of the activation function that enables the network with the capacity to learn and represent complex, non-linear patterns.

When the activation function F_p of the p^{th} layer is linear, it indicates that the activation function of each neuron is linear, effectively scaling its input. This operation across a p^{th} layer of n neurons can be represented by a diagonal matrix $F_p = \text{diag}(f_{p1}, f_{p2}, \ldots, f_{pn})$, where f_{pk} denotes the linear activation function of the k^{th} neuron in the p^{th} layer. Thus, the output of the p^{th} layer (X_p) can be formulated as the result of the matrix product $X_p = F_p \cdot W_p \cdot X_{p-1}$. Given linear activations, the output of the final layer (X_L) can be represented as a single linear equation from the initial input I_1, expressed as

$$X_L = \prod_{p=1}^{L} F_p W_p \cdot I_1. \tag{4.3}$$

This property underscores the fact that a network composed entirely of linear activations, regardless of its depth (i.e. the number of layers L), can ultimately be simplified to a single linear function from input to output. This effect arises because the composition of multiple linear functions is itself a linear function. Consequently, the network could be equivalently represented by just an input and an output layer, effectively negating the potential benefits of the deep learning technique.

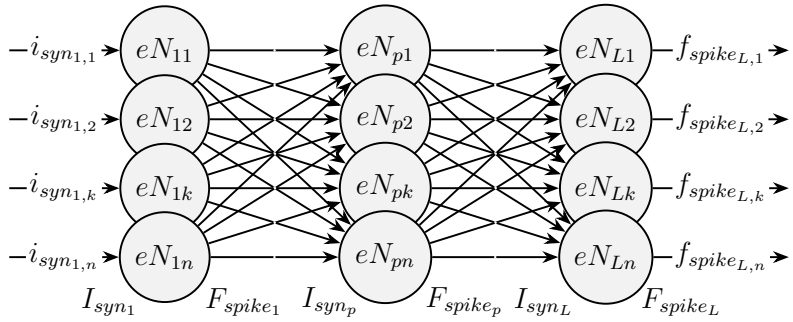


Figure 4.3: Illustration of an analog spiking neural network (A-SNN), showing layered connectivity using the eNeuron model defined in Fig. 4.1(b).

B .Analysis on A-SNN

Figure 4.3 illustrates an architecture of the A-SNN with L interconnected layers of eNeurons modeled as shown in Fig. 4.1(b). Each eNeuron eN_i processes an incoming synaptic current i_{syn_i} and emits spikes at a rate f_{spike_i}, reflecting the intensity of i_{syn_i}. This relationship between the input current and the output rate is captured by $f_{spike_i} = h_i(i_{syn_i})$, with h_i representing the transfer function of eN_i. In this network structure, eN_i is linked to n preceding layer eNeurons eN_k $(\forall; k \in [1, n])$ through eSynapses eS_{ki}. These eSynapses convert the spiking rate f_{spike_k} from eN_k into a synaptic current $i_{syn_{ki}}$ directed towards eN_i. The conversion of the spiking rate into synaptic current by eS_{ki} is described by $i_{syn_{ki}} = g_{ki}(f_{spike_k})$, where g_{ki} denotes the transfer function of eS_{ki}. Accordingly, the total input current i_{syn_i} for eNeuron eN_i is the sum of synaptic currents from connected eNeurons, expressed as $i_{syn_i} = \sum_{k=1}^{n} i_{syn_{ki}} = \sum_{k=1}^{n} g_{ki}(f_{spike_k})$. Thus, the generic expression for A-SNNs, where eNeurons and eSynapses are based on the spike-rate model, is obtained as follows

$$f_{spike_i} = h_i(i_{syn_i}) = h_i(\sum_{k=1}^{n} i_{syn_{ki}}) = h_i(\sum_{k=1}^{n} g_{ki}(f_{spike_k})). \tag{4.4}$$

Extending this relationship to a layer-wide perspective, let's consider a p^{th} layer $(\forall\ p \in [1, L])$, comprising n eNeurons. The input vector I_{syn_p} represents the synaptic inputs to this layer, feeding into its n eNeurons, and the output vector F_{spike_p} represents the collective spiking responses from these neurons. For this layer, and based on the aforementioned definitions, the input vector can be expressed as $I_{syn_p} = G_p(F_{spike_{(p-1)}})$, with G_p is the vector of cumulative transfer functions of eSynapses at the p^{th} layer, and $F_{spike_{(p-1)}}$ is the output vector of the $(p-1)^{th}$ layer. Consequently, the output F_{spike_p} of p^{th} layer is derived from its input as $F_{spike_p} = H_p(I_{syn_p})$, where H_p is the vector representing the transfer functions of eNeurons within this layer.

The formulations for I_{syn_p} and F_{spike_p} are detailed below

$$\begin{aligned} I_{syn_p} &= (i_{syn_{p,1}}, i_{syn_{p,2}}, \ldots, i_{syn_{p,n}})^T, \\ &= (\sum_{k=1}^{n} g_{p,k1}(f_{spike_{(p-1),k}}), \sum_{k=1}^{n} g_{p,k2}(f_{spike_{(p-1),k}}), \ldots, \sum_{k=1}^{n} g_{p,kn}(f_{spike_{(p-1),k}}))^T, \\ &= G_p(F_{spike_{(p-1)}}). \\ F_{spike_p} &= (f_{spike_{p,1}}, f_{spike_{p,2}}, \ldots, f_{spike_{p,n}})^T, \\ &= (h_{p1}(i_{syn_{p,1}}), (h_{p2}(i_{syn_{p,2}}), \ldots, h_{pn}(i_{syn_{p,n}}))^T, \\ &= H_p(I_{syn_p}). \end{aligned} \tag{4.5}$$

Combining (4.2) with the input and output vectors, I_{syn_p} and F_{spike_p}, at the p^{th} layer uncovers a key formula $F_{spike_p} = H_p(G_p(F_{spike_{(p-1)}}))$. When either or both transfer functions of eNeurons and eSynapses exhibit non-linearity, the corresponding vector functions H_p and G_p also become non-linear. Under such conditions, mirroring the analysis conducted for DNNs in Sec. 4.1.2.A, deep learning becomes feasible for A-SNNs equipped with numerous hidden layers. This non-linearity opens up opportunities to benefit from the deep learning technique, enabling the network A-SNN to address complex problems.

Conversely, if the transfer functions of both eNeurons and eSynapses are linear, then the functions H_p and G_p for any layer p are linear as well. In this scenario, the H_p for a layer of n eNeurons can be depicted by a diagonal matrix as $H_p = \text{diag}(h_{p1}, h_{p2}, \ldots, h_{pn})$, with h_{pk} representing the linear transfer function of the k^{th} eNeuron ($\forall\ k \in [1, n]$) in the p^{th} layer. Additionally, G_p can be described as a linear combination applied to the output $F_{spike_{(p-1)}}$ from the preceding layer, formulated as $G_p = W_{syn_p} \cdot F_{spike_{(p-1)}}$, where W_{syn_p} denotes the synaptic weight matrix between layers $(p-1)$ and p. Consequently, the output spiking frequency F_{spike_p} of a p^{th} hidden-layer can be obtained from the matrix product $F_{spike_p} = H_p \cdot W_{syn_p} \cdot F_{spike_{(p-1)}}$.

Under linear functions, the output F_{spike_L} of the final layer L simplifies to a single linear mapping from the initial input I_{syn_1}, as follows

$$F_{spike_L} = \prod_{p=1}^{L} H_p W_{syn_p} \cdot I_{syn_1}. \tag{4.6}$$

This characteristic observed in A-SNNs with linear transfer functions, as detailed in (4.6), mirrors that of DNNs with linear activation functions, noted in (4.3). It highlights that an A-SNN, when composed of linear transfer functions across its eNeurons and eSynapses, effectively functions as a network with only two layers. Since deep learning fundamentally requires non-linear functions within networks that include hidden layers, this structure negates the applicability of the deep learning technique. Results of this feasibility study, considering different transfer functions for eNeurons and eSynapses, are elaborated in Sec. 5.2.1.A.

4.1.3 . A-SNN Synthesis with Deep Learning

Deep learning application in A-SNN becomes viable when either or both eNeurons and eSynapses demonstrate non-linear behavior. This scenario necessitates a structured framework for A-SNN synthesis with deep learning, as shown in Alg. 4.1, with the corresponding code available in [28]. This framework involves training and testing A-SNN in TensorFlow, taking into account physical constraints and neuromorphic-circuits characteristics. The synthesis of A-SNN through deep learning includes several key steps, as follows:

The Activation Function plays a critical role in training networks via deep learning on TensorFlow, linking the neuron's input sum to its output. It performs typically non-linear relationship for deep learning purposes, as highlighted in Sec. 4.1.2. Thus, one should identify the activation function appropriate for the A-SNN learning based on its main characteristics. In traditional DNNs, the activation function is represented by f in the generic expression (4.1). It includes sigmoid, tanh, or ReLU, all recognized for their non-linearity. By comparing the formulations of DNNs and A-SNNs from (4.1) and (4.4), one can deduce the suitable activation function for A-SNN. This deduction is based on analyzing the eNeuron's h and the eSynapse's g transfer functions, derived from post-layout simulation results. These functions display non-linear characteristics over their entire operational range, which leads to two potential methods for determining A-SNN's activation function in the context of deep learning: the simplified approach and the standard approach. In both approaches, the activation function requires normalization to enhance training effectiveness within the TensorFlow framework [68].

The Simplified Approach involves the non-linear transfer function of the eNeuron and the linear transfer function of the eSynapse within a restricted dynamic range. Under this scenario, the generic expression of A-SNN, as defined in (4.4), simplifies to

$$f_{spike_i} = h_i(\sum_{k=1}^{n} g_{ki}(f_{spike_k})) = h_i(\sum_{k=1}^{n} w_{syn_{ki}} f_{spike_k} + b_2). \tag{4.7}$$

Here, the transfer function g_{ki} of eSynapse is linear, relying on synaptic weights $w_{syn_{ki}}$ and spiking rates f_{spike_k} from preceding eNeurons, with b_2 as a constant bias term. Synaptic weights $w_{syn_{ki}}$ are given from the current mirror gain ratio $\frac{(W/L)_1}{(W/L)_2}$, obtained from MP_1 and MP_2 transistors in the eSynapse circuit illustrated in Fig. 2.10 (see Sec. 2.3.2). This formula aligns with the expression (4.1) of DNN, allowing the transfer function h_i of eNeuron to directly provide the activation function for A-SNN learning. This simplified approach for the activation function was used for the A-SNN stage of the RF NeuroAS system.

The Standard Approach assumes that both eNeurons and eSynapses possess non-linear transfer functions. Under such conditions, simplifying the A-SNN's generic formula (4.4), to identify an appropriate activation function for A-SNN synthesis with deep learning becomes infeasible. To address this challenge, an alternative strategy redefines the generic expression of A-SNN. It focuses on the output of an eSynapse eS_{ij} linking an eNeuron eN_i from the p^{th} layer to an eNeuron eN_j in the $(p+1)^{th}$ layer. This focus diverges from examining the output of eN_i linked with prior layer eNeurons eN_k from $(p-1)^{th}$ layer. Thus, the revised generic expression (4.4) for A-SNN is presented as

$$i_{syn_{ij}} = g_{ij}(f_{spike_i}) = g_{ij}(h_i(i_{syn_i})), \tag{4.8}$$

where $i_{syn_{ij}}$ denotes the synaptic current from the eSynapse eS_{ij}; i_{syn_i} is the aggregated synaptic currents fed into eN_i, computed as $\sum_{k=1}^{n} i_{syn_{ki}}$. When considering an ideal current mirror in the circuit, the function g_{ij} is modeled as the product of a non-linear function $g_{n_{ij}}$ and a constant current gain, leading to

$$i_{syn_{ij}} = g_{n_{ij}}(h_i(i_{syn_i})) \cdot \frac{(W/L)_1}{(W/L)_2}. \tag{4.9}$$

Accordingly, the activation function for A-SNN learning emerges from the relationship $g_{n_{ij}}(h_i)$. The activation function for A-SNN's deep learning is obtained from the transfer functions of eNeurons and eSynapses to incorporate the analog features. However, this activation function is not directly available in TensorFlow. It requires a compatible fitting from post-layout simulation results to effectively train and test the A-SNN within TensorFlow.

The Activation Function Fit depends on the approach being used. When the simplified activation function is considered, the most appropriate fit can be a high-order polynomial fit (greater than 12). Details results of this fit, used in the A-SNN stage of the RF NeuroAS system, are shown in Sec. 5.1.4. When the standard activation function is used, two fits are considered. For the training stage, a sigmoid fit is suggested as the activation function with a generalized logistic function [216]. The sigmoid fit provides a differentiable and closely aligned fit that supports gradient descent learning algorithms and enhances the network convergence. For the testing stage, either a sigmoid or polynomial fit may be applied to closely match the activation function of A-SNN. The high-order polynomial fit can more accurately represent the activation function across its entire dynamic range, especially around zero. However, the requirement for a high-order polynomial fit makes it less suitable for training due to the lack of derivability around zero, leading to discontinuity. This derivability is mandatory for standard learning optimizers like SGD and Adam. Detailed results of both fits, considering various eNeuron types, are discussed in Sec. 5.2.1.A.

Concerning the Weight Extraction, after completing the offline training of A-SNN with the selected activation function model (i.e. the fit model), the trained weights of the network are fixed for subsequent testing. This testing can take place either through TensorFlow, using the established weights, or directly within the A-SNN's hardware setup. For on-chip testing, the trained weights are particularly applied in the dimensions of transistors within the current mirror of the eSynapse, as detailed in Sec. 4.1.1. However, translating these numerical weights to circuit parameters can introduce accuracy losses due to inherent quantization in analog design. Besides, the transistor sizing in current mirrors can potentially lead to mismatches due to process variability, resulting in imprecisions in the synaptic weights of the eSynapse circuit. To accurately consider on-chip testing conditions within the software framework (via Tensorflow), it is crucial to model these variations and incorporate them. Therefore, during software testing of A-SNN in TensorFlow, trained synaptic weights are represented as statistical variables adhering to a normal distribution. Here $\bar{w}_{syn}$ represents the mean synaptic weight refined during the training phase, with a standard deviation calculated as $\sigma_{w_{syn}}$=0.01 $\times$ $\bar{w}_{syn}$. This statistical representation of synaptic weights in software testing aims to account for the variances that hardware weight implementation might introduce, ensuring a precise evaluation of A-SNN with real-world performance capabilities.

Algorithm 4.1 A-SNN Synthesis Framework with Deep Learning

1. Network Structure for the MNIST problem
`Structure = [` ▷ Structure could be adjusted for any other problem
  `locally_connected7x7(filters=1, kernel=(4,4), strides=(4,4))`
  `locally_connected3x3x3(filters=3, kernel=(2,2), strides=(2,2))`
  `dense(outputs=10)]`
2. Load post-layout (PLS) transfer function of eNeuron and eSynapse
`transfer_functions = load_PLSresults(model_eNeuron, model_eSynapse)`
3. Activation Function Identification
`activation_function = (`
  `transfer_functions`,
  `approach: simplified or standard`,
  `dynamic_range_restriction: True/False`,
  `normalization`
  `fitting: sigmoid_fit or polynomial_fit`)
4. Network Model for Training and Testing
`model_training = (Structure, activation_function, MNIST_train_data)`
`model_testing = (Structure, activation_function, MNIST_test_data)`
5. Network Model Training on MNIST dataset
for $epoch = 1$ to 100 **do**
  `accuracy_training, tensor_weight= learning(model_training, epoch)`
end for
5. Weight Extraction
if `accuracy_training` ≥ 0.9 **then**
  `trained_weight = tensor_weight`
else if $epoch == 100$ **then**
  `trained_weight = tensor_weight`
end if
6. Process Variability Consideration
$\bar{w}$ = `trained_weight`
$\sigma_w = 0.01^*$ `trained_weight`
`statistic_weight = normal_distribution(`$\bar{w}$`,` σ_w`)`
7. Network Model Testing on MNIST dataset
`accuracy_testing = testing(model_testing , statistic_weight)`

4.1.4 . A-SNN in solving MNIST Problem: Handwritten Digit Recognition

To assess the capabilities of A-SNN with deep learning, the MNIST problem is addressed, which is a benchmark for recognizing and classifying handwritten digits. This challenge stands for the MNIST dataset, which is a compilation from the Modified National Institute of Standards and Technology [49]. The MNIST dataset contains a large set of 28x28 pixel grayscale images of handwritten digits, from 0 through 9, used for training and testing the neural network. This

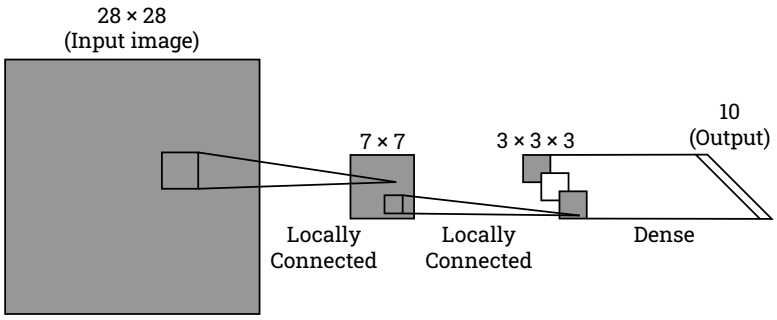


Figure 4.4: Representation of the A-SNN architecture for handwritten digit recognition, using MNIST dataset.

dataset is pivotal for training and evaluating neural networks ability to accurately classify each image into its corresponding digit class. For this purpose, A-SNN is trained with 60,000 images from the MNIST dataset and tested against a separate set of 10,000 images. This process is facilitated through TensorFlow, utilizing the adaptive moment estimation (Adam) optimizer to enhance the performance and the scalability of the network [53].

The structure of A-SNN dedicated for the MNIST problem, illustrated in Fig. 4.4, incorporates 86 eNeurons and 1238 eSynapses across two hidden layers. This network processes a 28x28 pixel grayscale image representing a handwritten digit (ranging from 0 to 9), leading to an output layer of 10 eNeurons, each corresponding to one of the ten digit categories. The first hidden layer consists of 49 (7x7) eNeurons and is locally connected to the input layer, implying that the eNeurons in this layer target only specific regions of the input image. It identifies local characteristics of the image through a convolution operation with a 4x4 kernel, using a single filter and a stride of 2x2. This approach progressively applies the kernel across different sections of the image, effectively reducing the original 28x28 image to a 7x7 layer, thereby streamlining the data for deeper analysis while preserving essential details.

The second hidden layer comprises 27 (3x3x3) eNeurons and is locally connected to the first hidden layer through a 2x2 kernel, using three filters and a stride of 2x2. This configuration further refines the extraction of relevant information, optimizing network complexity for efficient processing. The second hidden layer is then connected densely to the output layer, ensuring every eNeuron in the second hidden layer links to all eNeurons in the output layer. This connectivity facilitates a robust synthesis of processed information, leading to accurate digit classification.

The framework for the synthesis of A-SNN, applying deep learning to solve the MNIST problem, is detailed in Alg. 4.1. This synthesis of A-SNN, for training and testing, follows the steps outlined in Sec. 4.1.3. Therefore, it adopts the standard approach for activation function identification, and uses the sigmoid fitting for training and polynomial fitting for testing, followed by the application of the weight extraction technique. Concerning the key elements of A-SNN, it incorporates the three distinct eNeuron types along with the excitatory eSynapse, as discussed in Sec. 4.1.1. This diversity allows for an evaluation of the A-SNN's efficacy in addressing the MNIST problem across eNeurons ranging from less to more biologically plausible.

4.2 . Time Learning (STDP) Approach for A-SNN

One of the primary obstacles in implementing deep learning for A-SNN (as developed in previous Sec. 4.1) is its lack of biological plausibility. This issue poses significant challenges for its integration with neuromorphic processors, as elaborated in Sec. 2.3.3. Among the prominent categories of bio-inspired learning that could be both efficient and accurate on neuromorphic hardware is STDP. This section introduces the application of STDP, which is a method that adjusts synaptic weights based on the timing of spikes between pre and post eNeurons within the A-SNN.

To explore the potential of STDP with A-SNN, the section begins with the development of an eNeuron model suitable for STDP learning using the Brian 2 Simulator (Sec. 4.2.1). Subsequently, it conducts an in-depth noise analysis of the eNeuron to establish an accurate and complete model, crucial for understanding the impact of noise on spike timing (Sec. 4.2.2). This analysis addresses the feasibility of STDP for A-SNN, given that precise spike timing is critical for this time-based learning method. Following this, the section outlines a synthesis framework developed for A-SNN based on this time-based learning technique (Sec. 4.2.3). Lastly, this framework is showcased through solving the XOR and MNIST problems (Sec. 4.2.4), as a demonstration of A-SNN's performance when applying the bio-plausible STDP learning.

4.2.1 . Neuron Model in A-SNN

As the purpose is to study the applicability of STDP learning within the A-SNN, the eNeuron model is designed to encode and process information based on the timing of individual discrete spikes. To this end, the spike-time-based eNeuron model is adopted, focusing on its discrete spike behavior and precise timing rather than the continuous spike rate transfer functions used for deep learning application in A-SNN. Figure 4.5 illustrates the configuration of an eNeuron within the A-SNN, labeled eN_i, linked to three other eNeurons — eN_1, eN_2, and eN_3 — via

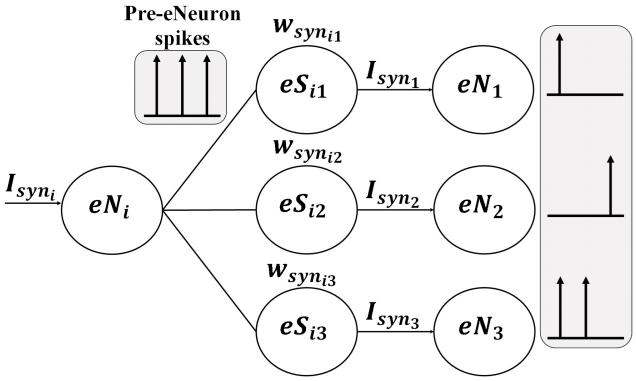


Figure 4.5: Structure of an A-SNN composed of an eNeuron connected to three other eNeurons through eSynapses. Information is processed through discrete spikes.

synapses eS_{i1}, eS_{i2}, and eS_{i3}. This setup processes information through discrete spikes over time, with synaptic weights adjusted accordingly. Following this, a detailed description of the eNeuron and eSynapse models used in this scenario is provided.

A .eNeuron

To incorporate the eNeuron into an A-SNN for training via STDP learning on the Brian2 simulator platform, a comprehensive and precise model of this eNeuron is necessary. Various models are discussed in the literature and detailed in Sec. 2.3.2, ranging from less to more biologically inspired, as depicted in Fig. 2.8. These models feature a spectrum of mathematical complexity, from simple to more intricate equations. However, these eNeuron models are abstract and do not consider physical constraints and process variability. To develop an accurate model of the eNeuron, it is essential to consider the post-layout simulation behavior of the designed circuit. Consequently, four different eNeurons have been redesigned from state-of-the-art models; these include two based on the Morris-Lecar (ML) model: the biomimetic ML (b-ML) model from [156], and the parasitic capacitance-based ML (p-ML) from [158], along with two based on the Leaky Integrate-and-Fire (LIF) model: the typical (t-LIF) from [157], and the axon hillock-based (AH-LIF) from [155]. These circuits were redesigned using BiCMOS 55 nm technology, as previously shown in Fig. 2.9, with layouts and corresponding sizes detailed in Appendix D.

To accurately model the eNeuron behavior derived from analog circuit design, three principal properties should be extracted from the post-layout simulations and incorporated into the model:

1- Dynamics of potential membrane behavior over time. Understanding the spiking behavior is crucial, as it reflects the changes in membrane potential over time, as shown in Fig. 2.6(b). The spiking behavior, also known as action potential, includes several stages such as depolarization, repolarization, hyperpolarization, and resting potentials. Important parameters to be included in the model encompass the resting potential, where the eNeuron remains at rest in the absence of stimuli; the threshold voltage that triggers a spike in response to a stimulus; the timing of spike generation; the leakage time which describes how quickly the membrane potential decays towards the resting state due to leakage.

2- Firing rate response over input excitation. The eNeuron circuit is designed to generate a series of spikes at varying rates for various input strengths depending on the circuit parameters. It is vital to account for the transfer function of the eNeuron, which describes how the firing rate response is influenced by different synaptic current values.

3- Random noise from the transistors in analog circuit. To get a fully accurate model of the eNeuron circuit, the random noise from the transistors within the analog circuit must be included. This aspect is particularly important for STDP learning applications, where noise can cause variations in spike timing, a critical factor for adjusting synaptic weights through STDP. Details regarding the noise property will be provided in Sec. 4.2.2, and subsequently integrated into the eNeuron model.

In this thesis, b-ML eNeuron is modeled following the three properties, instead of relying on ML model (shown in (2.6), (2.7), (2.8)) that do not consider the physical constraints of analog circuits. This model stands out for its flexibility, allowing for easy adaptation to other eNeuron types while maintaining low computational demands and high precision in fitting eNeuron

properties. The dynamics of the spike is presented as

$$\frac{dV_m}{dt} = \frac{(V_{rest} - V_m)}{\tau_{leak}} + I_{syn} \cdot \frac{R_m}{\tau_m}. \tag{4.10}$$

It describes the evolution of the membrane potential V_m of the b-ML eNeuron, influenced by two main components: the leakage current and the synaptic input current. The first term $((V_{rest} - V_m)/\tau_{leak})$ represents the leaky behavior of the b-ML eNeuron, driving V_m back towards the resting potential V_{rest} over time, in the absence of input current. This term models the natural tendency of the eNeuron's membrane to resist changes in potential and return to a baseline state due to the leak channels, ensuring stability and preventing runaway excitation. The time constant τ_{leak} indicates how quickly the eNeuron's membrane potential returns to rest. Post-layout simulations show that the b-ML eNeuron circuit exhibits different τ_{leak} values for varying magnitudes of input currents. A constant value of τ_{leak}, equal to the mean over the entire range of current, is selected.

The second term $(I_{syn} \cdot R_m/\tau_m)$ accounts for the input-driven current through the eNeuron's membrane, enabling the eNeuron to process and respond to environmental information. This term reflects the direct influence of external current I_{syn} on driving the membrane potential away from V_{rest}, potentially towards the threshold for firing an action potential. The membrane time constant τ_m (typically $R_m \cdot C_m$) represents how quickly the eNeuron's membrane potential responds to input currents, encompassing both the charging and discharging dynamics. Here, the eNeuron model in (4.10) closely resembles the simple equation of the LIF model in (2.5).

To perfectly model the spike shape of the b-ML eNeuron, a precise control over the dynamics of V_m during its rise and fall phases is necessary. This can be achieved through a variable membrane resistance R_m, which changes in a non-linear piecewise manner based on the membrane potential as described by

$$R_m = \sum_{i=1}^{n} (\text{if } (V_m \leq V_i \text{ and } V_m > V_{i-1}) \text{ then } R_i \text{ else } 0)\,. \tag{4.11}$$

In this approach, R_m is set to different values R_i depending on V_m relative to a series of voltage thresholds V_{i-1} and V_i. This segmentation divides the range of membrane potential V_m into different segments, each associated with a distinct resistance value R_i.

This extension into the model provided by (4.11) enhances the fidelity of the b-ML eNeuron model over the simple LIF model by reflecting the activation and inactivation of ion channels occurring as the ML eNeuron depolarizes or hyperpolarizes. The proposed model (4.10), (4.11) is a physical-informed version of the typical ML model (2.6), (2.7), (2.8), while maintaining similar behavior and avoiding the need for numerous constants. A finer segmentation of R_m allows more precise adjustments to small changes in V_m, better reflecting the nonlinear properties of the ML eNeuron. However, while finer segmentation can increase the precision and realism of the model, it also increases its complexity.

Equations (4.10) and (4.11) address the first property of the b-ML eNeuron, identified as the spike shape over time. To consider the second property of the eNeuron model — its transfer function given by the firing rate response over input current — the synaptic current I_{syn} in (4.10)

is replaced by

$$I'_{syn} = \delta(I_{syn}) \cdot f_{\text{eNeuron}}(I_{syn}), \tag{4.12}$$

where $f_{\text{eNeuron}}(I_{syn})$ is the interpolation function derived from the post-layout simulations results of the nonlinear b-ML eNeuron transfer function. The coefficient $\delta(I_{syn})$ adjusts the interpolated function affected by the leakage term. Additionally, the current I'_{syn} is limited to 15 nA to prevent transistors from operating out of the deep-subthreshold region. The results of the b-ML eNeuron model and its accuracy are discussed in Sec. 5.2.2.A.

B .eSynapse

In the architecture of A-SNN, eSynapses play a critical role in connecting eNeurons, serving as the fundamental conduits for learning, particularly through STDP. To effectively support STDP learning, synaptic adjustments are made based on the precise timing of spikes. Thus, eSynapses must finely tune synaptic weights in response to temporal spike patterns. For this purpose, the conductance-based model is considered for eSynapses, providing a biologically inspired representation of synaptic functionality [198].

Unlike current-based eSynapses, which simply add a fixed current to the eNeuron, conductance-based eSynapses capture the dynamic time-varying nature of synaptic efficacy driven by variations in synaptic conductance. In a conductance-based eSynapse, which can be excitatory ($l = e$) or inhibitory ($l = i$), the synaptic current I_{syn} directly affects the membrane potential of the postsynaptic eNeuron and is calculated as follows

$$I_{syn,l} = \gamma_l \cdot g_l \cdot (E_{syn,l} - V_m), \quad l \in \{e, i\} \tag{4.13}$$

where γ_l is a scaling factor differentiating the impact of excitatory and inhibitory synapses on the total postsynaptic current; g_l is the synaptic conductance; $E_{syn,l}$ is the equilibrium potential of the eSynapse; and V_m is the membrane potential of the postsynaptic eNeuron.

The excitatory eSnapse has its equilibrium potential $E_{syn,e}$ set above the resting membrane potential to promote depolarization of the postsynaptic eNeuron. This depolarization leads to a positive increase in synaptic current $I_{syn,e}$ that increases the likelihood of the eNeuron firing. Conversely, for an inhibitory eSynapse, $E_{syn,i}$ is typically close to or below the resting membrane potential, causing hyperpolarization of the postsynaptic eNeuron. This hyperpolarization leads to a negative increase in synaptic current $I_{syn,i}$ that reduces the likelihood of the eNeuron firing.

These variations in synaptic current (either a positive increase in $I_{syn,e}$ or a negative increase in $I_{syn,i}$) are driven by changes in synaptic conductance over time. For both cases, the conductance (g_e or g_i) increases by an amount corresponding to the synaptic weight and then exponentially decays over time toward zero, as described by

$$\frac{dg_e}{dt} = -\frac{g_e}{\tau_e}, \tag{4.14}$$

$$\frac{dg_i}{dt} = -\frac{g_i}{\tau_i}, \tag{4.15}$$

where τ_e and τ_i are the decay constants for excitatory and inhibitory conductances, respectively, determining how quickly each conductance returns to baseline. This conductance-based eSynapse model has proven feasible in analog circuits, effectively supporting the dynamic adjustment of synaptic weights necessary for STDP learning [152].

4.2.2 . Noise Analytical Tools

The eNeuron model, given by (4.10), (4.11), and (4.12) from the previous section, accurately captures the spike shape and firing rate properties of the eNeuron circuit. To enhance the model's precision with physical-informed parameters from eNeuron analog circuitry, it is crucial to incorporate its random noise. In this context, two types of noise are typically encountered: external noise from input signals and intrinsic noise from transistors. The eNeuron model demonstrates robustness against external noise by averaging the input synaptic currents. Therefore, this study focuses solely on the random noise caused by transistors, which is critical for achieving a more accurate representation of the eNeuron's functionality in analog circuits.

Figure 4.6 illustrates the noise model for the standard membrane node circuit, common to all considered eNeuron types (b-ML, p-ML, t-LIF, and AH-LIF). The membrane node of the eNeuron primarily consists of two transistors, MP_{Na} and MN_K, alongside a membrane capacitance C_m. In the p-ML eNeuron circuit, which exclusively uses transistors, C_m is assumed to represent the total parasitic capacitances at the membrane level of the eNeuron. Transistors MP_{Na} and MN_K simulate the dynamics of ion flows in and out of the eNeuron membrane, acting like electronic charge pumps. These are regulated by positive and negative feedback loops, forming a network of inverters that correspond to the activities of Potassium (K) and Sodium (Na) ions. For the purposes of noise analysis, the circuit is simplified by omitting these feedback loops and instead using V_{b1} and V_{b2} to represent the feedback voltages.

When the eNeuron receives a synaptic current I_{syn} from the eSynapse transistor MP_{syn}, the membrane capacitance C_m is charged through MP_{syn} and MP_{Na} (pull-up) and discharged through MN_K (pull-down). This process results in a rapid and significant fluctuation in the membrane potential V_m, prompting the eNeuron to generate a spike. To achieve this nonlinear functionality in eNeurons, whether using ML or LIF models, while also reducing power con-

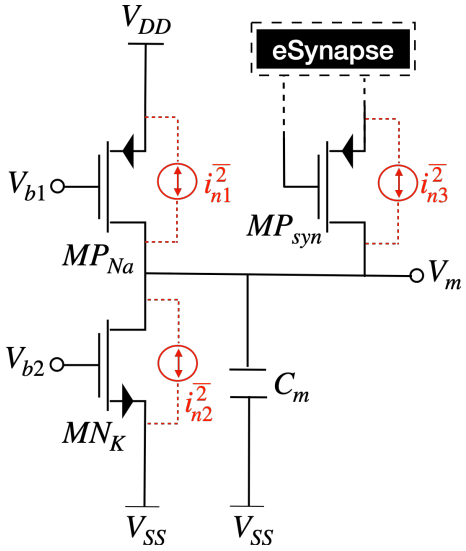


Figure 4.6: Noise model of an eNeuron membrane node connected to one eSynapse

sumption, the transistors are biased in the deep-subthreshold region. In this biasing setup, with low synaptic currents in the pA range, carrier diffusion becomes the dominant mechanism in the channel current, leading to significant variation of the current that flows across V_m. Under subthreshold region, the shot noise causes variations in the arrival time of individual electrons and holes at V_m. Shot noise is predominant over other types of noise, such as thermal noise, in this operation region [218]. Consequently, this study focuses on shot noise within the random noise model of the eNeuron and examines its impact on spike timing.

To effectively incorporate the noise of the eNeuron circuit into the eNeuron model, a thorough investigation of the noise characteristics is essential. This involves several key steps: **(A)** defining the expression of noise in spike timing, **(B)** understanding the type of noise distribution involved, **(C)** modeling its standard deviation from circuit parameters, and **(D)** assessing how the noise could impact the applicability of STDP.

A .Noise-Driven Temporal Deviation

As previously mentioned, shot noise impacts the timing of spike occurrence, potentially causing deviations from the expected timing. It is essential to accurately identify these noise-driven temporal deviations, particularly for time-dependent learning mechanisms like STDP, since they can influence the proper adjustment of synaptic weights. Figure 4.7 contrasts the spike timing of an ideal eNeuron, where noise is ignored, with that of a real eNeuron, where noise is taken into account. It highlights t_{rise}, the moment when the membrane potential just exceeds the threshold voltage, and T, the interval between consecutive spikes. In the ideal eNeuron, spikes occur consistently, such that a k^{th} spike is produced at rise time $t_{rise,ik} = t_{rise,i1} + k \cdot T_i$, where $t_{rise,i1}$ is the ideal rise time of the first spike, and T_i represents the consistently recurring ideal period. However, in a real eNeuron, the rise time and periods are inconsistent. The rise time for each spike varies, leading to spikes occurring earlier (e.g., $t_{rise,n1} < t_{rise,i1}$) or later (e.g., $t_{rise,n2} > t_{rise,i2}$) than in the ideal model. These variations lead to irregular periods

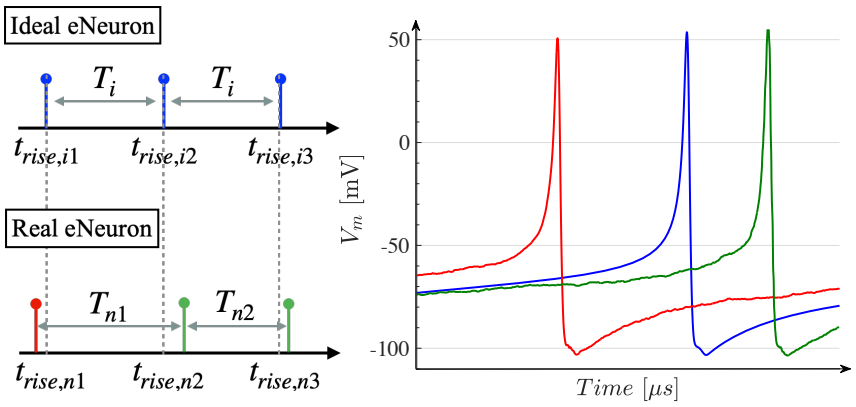


Figure 4.7: Comparison of spike timing and voltage response in ideal and noisy eNeurons. Left panel: spike intervals and rise times. Right panel: membrane potentials over time with a noise-free transient simulation in blue and trans-noise simulations in red and green.

(T_{n1}, T_{n2}, etc., not equal to the ideal period T_i). The graph in Fig. 4.7, through post-layout simulation results, illustrates how spike timing can shift due to noise: either to the left ($t_{rise,n}$ - $t_{rise_i} < 0$, depicted in red) or to the right ($t_{rise,n}$ - $t_{rise,i} > 0$, shown in green) relative to the ideal regular spike timing (plotted in blue).

There are two ways to account for this noise-driven temporal deviation: noise-driven period deviation and noise-driven rise deviation. Noise-driven period deviation ($\Delta T_n = T_n$ - T_i) refers to variations in the intervals between consecutive spikes. Noise-driven rise deviation ($\Delta t_{rise,n} = t_{rise,n}$ - $t_{rise,i}$) pertains to deviations in the time it takes for a spike to reach its threshold. Given the goal of applying the STDP learning — where synaptic weights are adjusted based on the exact timing of spikes between pre and post eNeurons — it becomes imperative to focus on the noise-driven rise deviation. This focus better accounts for deviations in spike occurrence timing rather than inter-spike intervals. Consequently, throughout this work, the identified noise parameter will be the noise-driven rise deviation, emphasizing its role in modeling the eNeuron for STDP learning.

B .Distribution of Noise-driven Rise Deviation

Noise in electronic circuits is a random phenomenon, and accurately modeling it requires a thorough understanding of its distribution. Figure 4.8 illustrates the post-layout distribution of noise-driven rise deviation in two scenarios: a capacitance-free eNeuron circuit (e.g., p-ML eNeuron, shown in Fig. 4.8(a)) and an eNeuron with inherent membrane capacitance (e.g., b-ML eNeuron, shown in Fig. 4.8(b)). Both eNeurons are stimulated with the same 25 pA synaptic current. The multiple plots in the figure represent the distributions of noise-driven rise deviation at various spike indices over time. For each spike occurrence (from $k = 1$ to n), the noise-driven rise deviation in both eNeurons follows a Gaussian distribution, though with different means and standard deviations. To model these means and standard deviations for any eNeuron at any spike occurrence, one should analyze the set of noise-driven rise deviation distributions across all spike occurrences.

Overall spike occurrences, as depicted in Fig. 4.8, the noise-driven rise deviation for both types of eNeurons exhibits a Gaussian random walk distribution [219]. This indicates that while each spike's timing noise follows a Gaussian distribution, the cumulative effect scales with the spike index k, aligning with a stochastic process typically used to model such random components. The mean of the noise-driven rise deviation distribution varies between the capacitance-free and capacitive eNeurons. The capacitance-free eNeuron demonstrates a random walk with a mean of zero, implying that for any spike index, the average noise-driven rise deviation is centered around zero (see Fig. 4.8(a)). However, for the capacitive eNeuron, the mean is not zero across spike occurrences (see Fig. 4.8(b)). According to the definition of a random walk [219], in cases where the mean is not zero, this mean varies linearly. It can be estimated as

$$\mu(\Delta T_{rise,n}) = k \times \mu_1(\Delta T_{rise,n}), \tag{4.16}$$

where k is the spike index of occurrence and $\mu_1(\Delta T_{rise,n})$ is the mean of the first spike occurrence (when $k = 1$). This means that the capacitive eNeuron exhibits an accumulated average noise-driven rise deviation over each spike occurrence, a typical result for any eNeuron with a membrane capacitance. This is likely due to the charging and discharging of the membrane capacitance over

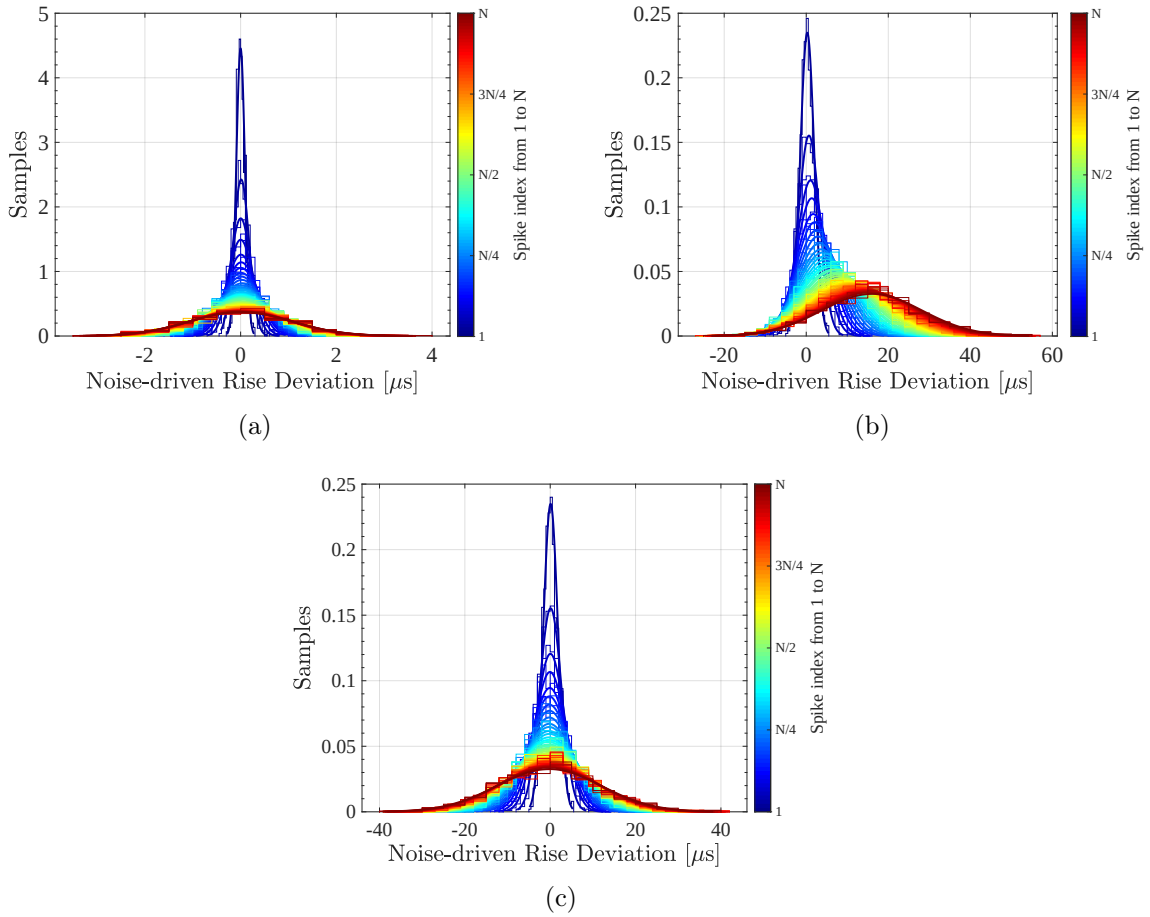


Figure 4.8: Distribution of noise-driven rise deviation in eNeurons: (a) p-ML eNeuron, (b) b-ML eNeuron before zero-mean adjustment, and (c) b-ML eNeuron after zero-mean adjustment, illustrating noise characteristics across multiple samples.

time, which linearly integrates the noise average into the spike generation process. This linear offset can be easily corrected to obtain the adjusted distribution of noise shown in Fig. 4.8(c) for the capacitive eNeuron.

Considering the random walk for both capacitance-free and capacitive eNeurons (Fig. 4.8(a) and Fig. 4.8(c), respectively), this distribution is characterized by a zero mean and a standard deviation that exhibits nonlinear cumulative variability over given spike occurrences [219]. This deviation can be estimated as

$$\sigma(\Delta T_{rise,n}) = \sqrt{k} \times \sigma_1(\Delta T_{rise,n}), \tag{4.17}$$

where k is the spike index, and $\sigma_1(\Delta T_{rise,n})$ is the standard deviation at the first spike. This nonlinear deviation likely results from the nonlinear accumulation of noise in the feedback inverter loops of the eNeuron circuit over spike occurrences.

To fully model the noise-driven rise deviation as a Gaussian random walk distribution (0, $\sqrt{k} \cdot \sigma_1(\Delta T_{rise,n})$), obtaining $\sigma_1(\Delta T_{rise,n})$ is crucial. It is the standard deviation of the random noise at the first spike, setting the stage for modeling the accumulation of this noise with each spike occurrence.

C .Noise Model

As discussed in the previous section, accurately modeling noise in an eNeuron requires determining the standard deviation of noise at the first spike occurrence, denoted as $\sigma_1(\Delta T_{rise,n})$. Abidi has proposed tools to analyze random noise contributions in ring oscillators [220]. He demonstrated that such noise can cause variations in switching timing. Referring to this work, and given that spiking eNeuron circuits operate on similar oscillation principles, insights from Abidi's noise jitter analysis were used for this purpose. The value of $\sigma_1(\Delta T_{rise,n})$ varies based on the circuit design (refer to Fig. 4.8(a) and Fig. 4.8(c), which show different orders of noise-driven rise deviation values) and the applied synaptic input current.

As illustrated in Fig. 4.6, the noise-driven rise deviation originates from three current noise sources, leading to two primary noise events. One event occurs in the pull-up branch where the synaptic current from the PMOS transistor MP_{syn} and the drain current from MP_{Na} increase V_m as charges accumulate in C_m. The other event occurs in the pull-down branch, driven by the drain current of the NMOS transistor MN_K, which reduces V_m as charges leave C_m. These two events are assumed to be uncorrelated, resulting in combined propagation delays expressed by

$$\sigma_1^2(\Delta T_{rise,n}) = \sigma_{t_{dN}}^2 + \sigma_{t_{dP}}^2, \tag{4.18}$$

where $\sigma_{t_{dN}}^2$ and $\sigma_{t_{dP}}^2$ are the noise sources from the two propagation delays of the pull-up and pull-down branches, respectively.

Before the discharge phase starts, PMOS pull-up transistors introduce initial noise into the capacitance C_m. The resulting mean square voltage noise and the associated noise-driven rise deviation are calculated as follows

$$\overline{V_n^2} = \int S_{V_n}(f)df = S_{I_n}. \int Z_{out}^2(f)df, \tag{4.19}$$

$$\sigma_{t_{dN1}}^2 = \frac{\overline{V_n^2}}{(I_D/C_m)^2}, \tag{4.20}$$

where $S_{V_n}(f)$ and S_{I_n} are the spectral densities of the integrated voltage noise and current noise, respectively. In this case, S_{I_n} refers to the shot noise, given as $S_{I_n} = 2\ q\ I_D$ [218]. The output impedance $Z_{out}(f)$ of the eNeuron circuit consists of the parallel combination of C_m and the total output resistances of the transistors.

In addition to $\sigma_{t_{dN}}^2$, during the discharge phase, the capacitance C_m integrates noise into current over the duration t_{dN} in the transistor MN_K. Thus, the appropriate variation caused by this noise source is

$$\sigma_{t_{dN2}}^2 = \frac{q.t_{dN}}{I_D}, \tag{4.21}$$

where q is the electronic charge and I_D is the drain current of the transistor MN_k. In this context, all currents flowing into the eNeuron membrane are considered in the same order of magnitude, i.e. I_D and I_{syn} are supposed to be very close.

Using (4.18) and the total noise $\sigma^2_{t_{dN1}} + \sigma^2_{t_{dN2}}$ integrated during the pull-down process, the variance of noise-driven rise deviation in the eNeuron circuit can be expressed as

$$\sigma_1^2(\Delta T_{rise,n}) = \frac{q}{I_{syn}} \left(\frac{1}{f_{spike}} + \frac{r_{ds}.C_m}{\pi} \right), \tag{4.22}$$

where r_{ds} is the output resistance of the transistors, and f_{spike} is the spiking frequency defined over a specific time window. Given that the dynamic behavior of the eNeuron resembles that of an oscillator with a periodic waveform, f_{spike} is calculated as $(t_{dN} + 2 \cdot t_{dP})^{-1}$.

The phase noise is directly linked to this noise expression as [220]

$$\mathcal{L}(\Delta f) = \sigma_1^2(\Delta T_{rise,n}) \cdot f_{spike}^3, \tag{4.23}$$

where Δf is the frequency offset relative to the spiking frequency of the eNeuron.

Consequently, the noise-driven rise deviation model for any considered eNeuron circuit is represented by a Gaussian random walk distribution, with a mean of 0 and a standard deviation of $\sqrt{k} \cdot \sigma_1(\Delta T_{rise,n})$, where $\sigma_1(\Delta T_{rise,n})$ is calculated from (4.22). This noise model is dependent on circuit parameters C_m, f_{spike}, and r_{ds}, as well as the external input current I_{syn}. All detailed results for the noise model can be found in Sec. 5.2.2.A.

D .How Does Noise Affect the Applicability of STDP?

Given the noise model of the eNeuron circuit, it is crucial to evaluate whether STDP can be effectively applied in A-SNNs that use these eNeurons. Two variables are critical for this analysis:

1. ΔT_s represents the time difference between spikes of the pre and post-synaptic eNeurons. This interval is crucial for updating synaptic weights in STDP, as detailed in (2.11).

2. $\Delta T_{rise,n}$ has been previously defined as the deviation in spike rise timing due to shot noise from transistors.

As illustrated in Fig. 4.9, noise can cause deviations in the spike occurrence times of the post-eNeuron, shifting from T_1 to T_2 or T_4, with similar shifts possible for the pre-eNeuron (not shown for simplicity). If $\Delta T_{rise,n}$ is significantly smaller than ΔT_s, it does not affect the STDP weight adjustment. However, if $\Delta T_{rise,n}$ is on the same order as ΔT_s, it could significantly impact STDP. For example, if a pre-synaptic spike shortly precedes a post-synaptic spike (ΔT_s is positive and small), the synaptic weight should substantially increase according to STDP rules (Fig. 2.11). However, if $\Delta T_{rise,n}$ of the pre-synaptic eNeuron exceeds ΔT_s, the pre-synaptic spike might occur after the post-synaptic spike. Consequently, ΔT_s could shift to a slightly negative value, reducing the synaptic weight. This incorrect adjustment could adversely affect the accuracy of STDP learning. Post-layout simulations are conducted to determine the magnitude of $\Delta T_{rise,n}$ and evaluate the feasibility of implementing STDP in A-SNNs. Results of this feasibility analysis are presented in Sec. 5.2.2.B.

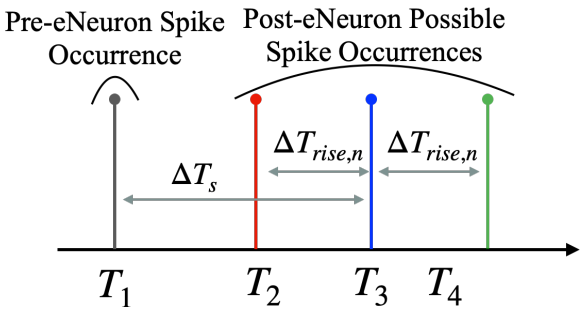


Figure 4.9: Impact of noise on STDP: Timing diagram showing pre-eNeuron spike at T_1 and potential post-eNeuron spikes at T_2 and T_4 due to noise ($\Delta T_{rise,n}$). ΔT_s represents the critical interval for STDP weight adjustments, illustrating how noise can shift spike timing and affect learning outcomes.

4.2.3 . A-SNN Synthesis with STDP Learning

To effectively implement STDP learning within an A-SNN, a structured framework is essential, which is outlined in Alg. 4.2, with the corresponding code available in [29]. It systematically organizes both the training and testing phases of the A-SNN within the Brian 2 simulator, taking into consideration the specific properties of eNeurons and the selection of learning parameters. The synthesis of A-SNN through STDP learning involves several key steps, outlined as follows.

Incorporation of eNeuron Model — Initially, the primary focus in the development of the A-SNN should be the eNeuron model. A thorough consideration of the eNeuron's properties within the model is crucial for an in-depth understanding of how these characteristics influence the learning process. In pursuit of simulating the A-SNN using a biologically inspired learning mechanism, the STDP, the b-ML eNeuron circuit has been selected due to its highly bio-inspired behavior compared to other eNeurons under study. The complete model of the b-ML circuit, previously detailed in Sec. 4.2.1 and Sec. 4.2.2, is implemented in Brian 2 simulator. This implementation accounts for several critical features: the spike shape as described in (4.10) and (4.11), the firing rate response to synaptic input from (4.12), and the noise distribution from (4.17) and (4.22). The flexibility of this model allows it to be adapted for use with other, less biologically inspired eNeuron circuits.

In addition to the eNeuron model, the conductance-based eSynapse model is also implemented, following (4.13), (4.14), and (4.15). Once the eNeuron and eSynapse models are integrated into the Brian 2 simulator, the architecture of the A-SNN can be designed with varying numbers of layers, eNeurons, and excitatory/inhibitory eSynapses, configured to address the specific complexity and requirements of the given problem. These correspond to the steps 1, 2, and 3 of Alg. 4.2.

Concerning the scenarios of noise in step 2 of Alg. 4.2, each accounting for noise in the eNeuron model differently, with details on the noise model provided in Sec. 4.2.2. **Scenario 1** incorporates the eNeuron model with a constant standard deviation (SD) for its random noise. This scenario does not the Gaussian walk distribution of the noise over spike occurrences, which

leads to a non-linear accumulation of its standard deviation over time. Here, the standard deviation considered is the average calculated across all spike occurrences. **Scenario 2** involves simulations with the complete eNeuron model, where the full noise model is considered. This includes the Gaussian walk distribution of the noise and its variable standard deviation over spike occurrences. In **Scenario 3**, simulations are conducted using the eNeuron model without considering its noise from the analog circuit. **Scenarios 3.1** and **3.2** reintroduce the noise model from Scenarios 1 and 2 only in the testing phase, respectively, enabling the assessment of the impact of random noise on post-trained A-SNN.

Input Encoding — The first layer of eNeurons receives external information and feeds it into the A-SNN. In this setup, input data, such as sensory stimuli, are encoded into the spiking activity of neurons through rate coding. This means that the strength of the input is translated into the firing rate of input neurons, with each eNeuron spiking at a rate that corresponds to the encoded information. For example, a more intense stimulus would lead to a higher firing rate in certain input eNeurons, while a less intense stimulus would result in a lower firing rate. Rate coding is preferred over the well-known temporal coding despite the latter's speed and energy efficiency. The rate coding offers greater resilience to noise by averaging out fluctuations, as demonstrated in results in Sec. 5.2.2.B. These specific spike rates are calculated based on (4.12) of the eNeuron model, which maps the firing rate response to the input excitation. The input encoding corresponds to step 4 of Alg. 4.2.

Unsupervised Learning with STDP — To authentically follow a bio-inspired approach for training the A-SNN using STDP, unsupervised learning is used (see details in Sec. 2.3.3). Unlike supervised learning, which relies on data tagged with correct answers, unsupervised learning focuses on identifying patterns without any pre-existing labels. The unsupervised learning strategy adopted is influenced by Diehl and Cook's 2015 framework [198], which uses STDP for training SNNs. However, their implementation focuses on a simple LIF model and does not account for the circuit properties or the effects of random noise from transistors, which are crucial for a more comprehensive neuromorphic system. The learning process is structured into three distinct phases: training, labeling, and testing, corresponding to step 5 of Alg. 4.2.

During the training process, the network uses STDP to adjust synaptic weights based on the timing of spikes between eNeurons. This phase operates without supervision, target labels, or error correction guidance directly from the input data. The network self-organizes according to the input patterns it receives, without supervision or reinforcement. Following training, a labeling process assigns meaningful labels to the eNeurons in the output layer. This step is crucial for evaluating the network's performance in tasks requiring categorization or classification. During labeling, the activity of each eNeuron in the output layer is monitored as the network is exposed to the same inputs used during training. Each eNeuron is then labeled post hoc, based on the input type that most frequently causes it to fire. Finally, the network's performance is evaluated during the testing phase. In this phase, new inputs are presented to the network, and the response of the output neurons is observed. The response of the neurons is checked against the labels assigned during the labeling phase. These three functions are repeatedly executed for a specified number of iterations, as detailed in step 6 of Alg. 4.2.

Algorithm 4.2 A-SNN Synthesis Framework for Unsupervised STDP Learning

```
1: 1. PLS activation function of eNeuron
2: activation_function = load_PLSresults(ML_eNeuron)
3: 2. Definition of eNeuron and synapses
4: MLeNeuron = [activation_function,
5:   Rm:  segmentation(nb=3),
6:   dynamic_range_restriction:  15nA,
7:   noise:  Scenario 1 or 2 or 3 ];
8: synapses = [conductance_based,
9:   type:  exci or inhi];
10: 3. Network Structure for the MNIST problem
11: SNN = [ #Network can be adjusted for any problem
12:   Input_layer(nb=784,type=MLeNeurons)
13:   Output_layer(nb=1225,type=MLeNeurons)
14:   fully_connect(Input_layer,Output_layer, synapses = exci & Inhi)
15:   Lateral_connect(Output_layer,Output_layer, synapses = Inhi)];
16: 4. Encoding input data   #from pixel to current value
17: train,test_data = load_MNIST()*3nA/255
18: 5. Network Model for Learning
19: training = (SNN,train_data,time=120μs, learning=STDP,weight_normalization)
20: labeling = (SNN,train_data)
21: testing = (SNN,test_data)
22: 6. Network Model Training on MNIST dataset
23: for epoch = 0 to max_epoch do
24:    accuracy = learning(SNN, epoch)
25:    if accuracy ≥ accuracy_min then
26:       break;
27:    end if
28: end for
29: final_accuracy,weights = testing()
30: if noise == 3 then
31:    accuracy_noise = testing(noise=1,2)
32: end if
```

In the framework, the dataset includes both features and labels, even though the training algorithm is unsupervised. This is essential for the testing phase, which requires correctly labeled data to compare against the network's output and determine the accuracy. Additionally, the framework incorporates a boolean setting to enable or disable the random noise of transistors within the eNeuron model. This feature allows for performance comparisons under various conditions and helps quantify the impact of eNeuron circuit noise on STDP training.

4.2.4 . A-SNN in solving XOR and MNIST Problems Using Unsupervised STDP

To evaluate the effectiveness of the A-SNN using unsupervised STDP, as outlined in the framework detailed in previous Sec. 4.2.3, two classification challenges are undertaken. The first challenge is the XOR problem, short for exclusive OR, which is a logical operation that outputs true only when the inputs differ from one another, either one input is true and the other is false, or vice versa. It is traditionally used as a fundamental test of a computational model's ability to handle non-linearity. The second challenge is the MNIST problem, described in Sec. 4.1.4, a standard benchmark in the field of machine learning for recognizing and classifying handwritten digits. It involves the MNIST dataset that comprises 70,000 images of digits ranging from 0 to 9, with each image having a resolution of 28x28 pixels.

Figure 4.10 illustrates the A-SNN architecture, which consists of an input layer and an output layer, designed to address either the XOR or MNIST problem. The eNeurons in the input layer are fully connected to those in the output layer through both excitatory and inhibitory eSynapses. All eNeurons are based on the b-ML eNeuron circuit model, and all eSynapses follow the conductance-based model. Lateral inhibition is implemented in the final layer through inhibitory eSynapses, connecting each eNeuron to all other eNeurons in the same layer. This "winner-takes-all" mechanism allows activated neurons to suppress the activity of their neighbors. It enhances contrast and sharpness in the network's output, emphasizing the most relevant signals and improving feature discrimination and noise reduction. The number of eNeurons in both layers varies with the complexity of the problem: for the XOR problem, 2 input neurons and 13 output neurons are used, while for the MNIST problem, 784 input neurons and 1225 output neurons are

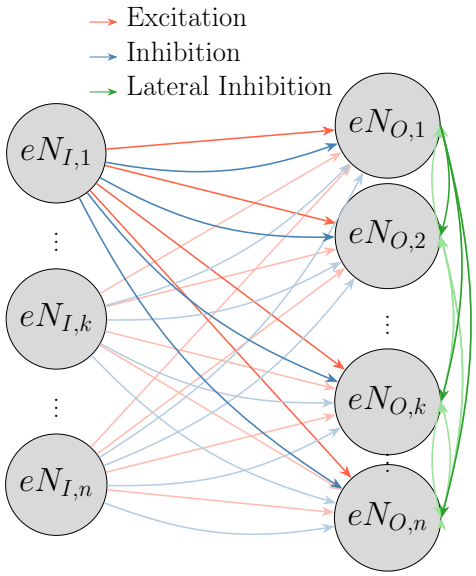


Figure 4.10: A-SNN architecture for XOR or MNIST problems, illustrating the input and output layers of eNeurons connected by excitatory (blue), inhibitory (red), and lateral inhibitory (green) eSynapses.

employed.

In the framework, input encoding uses a rate-code approach, with the following specifics for feeding inputs into the network: For the XOR problem, two possible logical values, 0 and 1, are presented to the network. A logical value of 0 corresponds to a low power input current of 0.3 nA, which triggers a spike rate of 215 kHz in the input eNeuron layer. A logical value of 1 triggers a higher input current of 2 nA, resulting in a spike rate of 540 kHz. These specific spike rates are calculated based on (4.12) from the eNeuron model, which maps the firing rate response to the input excitation. For the MNIST problem, a grayscale image is presented to the network, where each pixel's value ranges from 0 (black) to 255 (white), with intermediate values depicting varying shades of gray. The intensity of each pixel is translated into a current that varies from 0 to 3 nA, driving eNeuron spike rates between 0 and 610 kHz.

The accuracy performance results of the A-SNN using the unsupervised STDP framework for both the XOR and MNIST problems are detailed in Sec. 5.2.2.C. This analysis compares the different scenarios of noise consideration, detailed in Sec. 4.2.3.

4.3 . Conclusion

This chapter examined the learning processes in A-SNNs, which consist of eNeurons and eSynapses, requiring a thorough analysis that incorporates the models and circuit design constraints of these components. It covered two distinct learning techniques suitable for A-SNNs: one inspired by software-based deep neural networks and another based on temporally-driven STDP learning from biological neural mechanisms. Each method included a feasibility study and a synthesis framework, validated by case studies such as the MNIST and XOR problems. For deep learning in A-SNNs, the applicability depended on the nonlinearity and dynamic range of the spike-rate-based eNeuron's transfer function. Training was performed on the TensorFlow platform, using activation functions derived from the post-layout transfer functions of eNeurons and eSynapses. In contrast, STDP learning in A-SNNs was influenced by the random noise from transistors in the spike-time-based eNeuron circuits, which could cause variability in spike timings. The training for this approach was conducted using the Brian 2 simulator, highlighting the unique challenges each learning paradigm presented. The results validating the feasibility and synthesis of each technique for the A-SNN proposed in this chapter are detailed in Chap. 5.

Chapter 5

Results and Discussion

To tackle the challenges of RF localization within the Internet of Things (IoT), Chap. 3 introduced the analog spike-based neuromorphic system (RF NeuroAS). This system pinpoints the location of a mobile source on a two-dimensional plane, offering high precision and energy efficiency. The RF NeuroAS system comprises four key stages: the RF configuration of transmitter and receiver placements, data extraction for signal capture and dataset creation, neuromorphic pre-processing, and an analog-based spiking neural network (A-SNN). The structure of the A-SNN is designed to boost the efficiency of the RF NeuroAS, by featuring a network of interconnected layers of CMOS analog spiking neurons (eNeurons) and synapses (eSynapses). Therefore, a thorough investigation into the feasibility of learning techniques within the A-SNN is required, taking into account the design and functionality of its primary components to ensure effective learning.

Following this, Chap. 4 conducted an in-depth examination of two distinct learning techniques for the A-SNN, ranging from deep learning inspired by software-based deep neural networks (Sec. 4.1) to temporal learning derived from biological neural mechanisms (Sec. 4.2). This chapter aims to present performance insights derived from this thesis, highlighting the effectiveness of the RF NeuroAS in achieving energy-efficient source localization (discussed in Sec. 5.1) and the practical outcomes of employing learning techniques within the A-SNN (explored in Sec. 5.2).

5.1 . The Efficacy of RF Neuromorphic Localization

This section details the results of the RF NeuroAS system, which was presented in Chap. 3, delving into the outcomes associated with each stage of the system. Section 5.1.1 focuses on the RF configuration, where it delves into the antenna specifications for transmitters and receivers within the setup. Section 5.1.2 then examines the results of data extraction, where the behavior of features across various source positions and noise levels is analyzed, providing insights into the robustness and sensitivity of the data acquisition process.

Following this, Sec. 5.1.3 examines the neuromorphic pre-processing stage, highlighting its performance in terms of converting received power to spike rate and detecting bit patterns, which are critical for effective signal processing. Section 5.1.4 then assesses the performance of the analog spiking neural network stage, focusing on its accuracy in source detection. This evaluation underscores the efficacy of the neuromorphic approach in RF localization. Finally, Sec. 5.1.5 discusses the post-layout simulation results for a simplified version of the RF NeuroAS system, previously introduced in Sec. 3.5. This version is designed as a fully analog circuit using BiCMOS 55 nm technology, demonstrating the RF NeuroAS system's implementation efficiency.

5.1.1 . RF Configuration Evaluation

The RF configuration setup, described in Sec. 3.1 and shown in Fig. 3.2, outlines the placement of the mobile transmitter and four stationary receivers used in the RF NeuroAS system. These elements are strategically positioned on a 2D plane to optimize source identification by the RF NeuroAS system, using coordinates specified by angle θ_s and distance d_s from the origin with a resolution of 10 degrees. This arrangement is experimentally implemented in an anechoic chamber, and measurements are conducted using dipole bow-tie antennas for both transmission and reception, as detailed in Sec. 3.2.2. Understanding the characteristics of these antennas is crucial for interpreting the experimental results obtained from the anechoic chamber and comparing them with simulated results. The performance of the antenna is depicted through its measured return loss and its radiation pattern, illustrated in Fig. 5.1 and Fig. 5.2, respectively.

Return loss is defined as the measure of power that is reflected back from the antenna due to an impedance mismatch, represented as the ratio of reflected power to incident power and expressed in decibels (dB). A return loss value below -10 dB is typically considered favorable, indicating that less than 10% of the power is reflected, thereby enhancing the efficiency of power transmission and reception. The return loss of the antenna used in the experimental setup, denoted as S11, is plotted across the frequency range from 2 to 2.5 GHz as shown in Fig. 5.1. This plot demonstrates effective antenna matching, with the return loss consistently below -10 dB between 2.15 and 2.4 GHz. Furthermore, the minimum return loss occurs around 2.25 GHz, where it falls below -25 dB, indicating a very low level of reflected power. These results verify that the antenna operates effectively across this frequency range, perfectly aligning with the operational objectives of the RF NeuroAS system, which is designed to function at an operational frequency of 2.4 GHz.

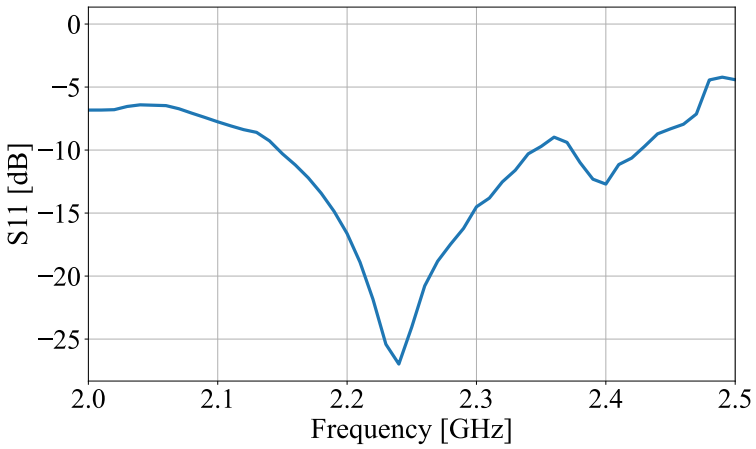
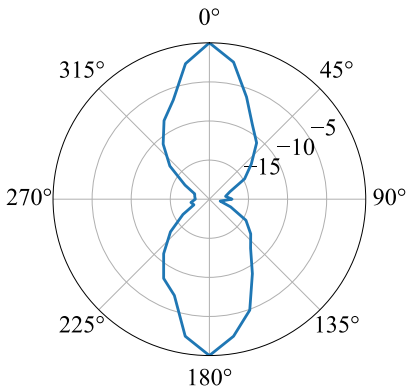


Figure 5.1: Measured return loss (S11) in dB of the bow-tie dipole antenna, spanning the frequency range of 2 GHz to 2.5 GHz.

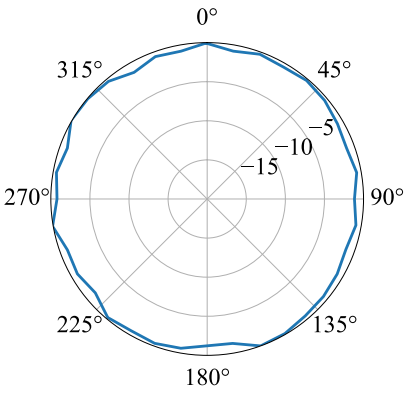
The radiation pattern of an antenna describes how it distributes energy into space, crucially indicating its directional characteristics and signal strength. Figure 5.2(a) illustrates the E-plane radiation pattern of the antenna. This pattern quantifies radiation intensity in dBi, as the

antenna completes a full 360-degree vertical rotation. It features two pronounced main lobes, which demonstrate strong signal strength in specific directions and highlight the antenna's ability to direct energy precisely. Moreover, the clear distinction and orientation of these lobes illustrate the bidirectional radiation pattern characteristic of dipole antennas, crucial for high-precision applications such as source localization.

Figure 5.2(b) displays the H-plane radiation pattern of the antenna. It captures the distribution of radiation intensity in dBi during a full 360-degree horizontal sweep. This pattern reveals a nearly uniform radiation across all directions, indicative of the near-omnidirectional radiation typical of dipole antennas. Such uniformity is critical as it provides consistent coverage essential when the source rotates in a 2D plane and transmits signals to the four receivers. Therefore, this uniform radiation pattern ensures reliable signal reception from every direction, a point further discussed in Sec. 5.1.2.



(a)



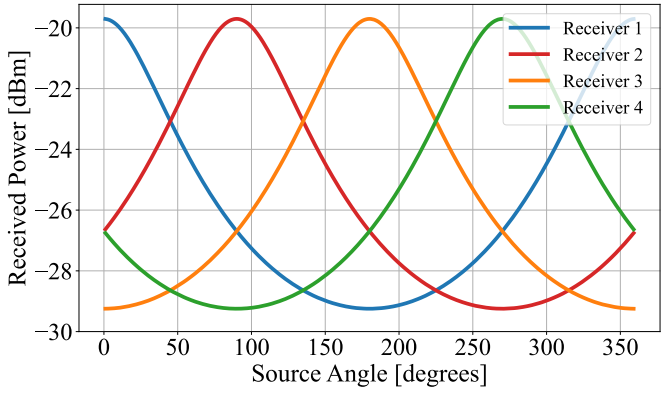
(b)

Figure 5.2: Measured radiation patterns of the dipole bow-tie antenna at 2.4 GHz: (a) E-plane (electric field) radiation pattern, illustrating bidirectional characteristics, (b) H-plane (magnetic field) radiation pattern, demonstrating a near-omnidirectional distribution.

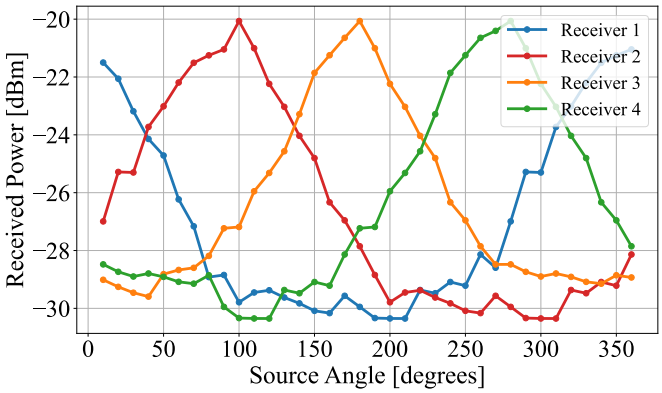
5.1.2 . Feature Extraction: Simulated vs. Experimental Datasets

After configuring the RF setup for the source and receivers, data is collected at varying source locations and under different noise conditions, as outlined in Sec. 3.2. This data, organized into features and labels as depicted in Fig. 3.3, is used to create a simulated dataset in Matlab (Sec. 3.2.1), and an experimental dataset obtained from measurements in the anechoic chamber (Sec. 3.2.2). Figure 5.3 illustrates the power patterns received by the four receivers, spanning the complete range of source angles from 0 to 360 degrees. Figure 5.3(a) presents these patterns from the SimLocRF dataset, while Fig. 5.3(b) presents the patterns from the MesLocRF dataset. In both scenarios, the source is placed 0.5 meters from the origin and transmits a signal of 10 dBm in an environment with a high signal-to-noise ratio (SNR) of 20 dB.

The received power from the SimLocRF dataset, shown in Fig. 5.3(a), shows a smooth curve ranging between -20 to -29 dBm across the full 360-degree source rotation. This variation



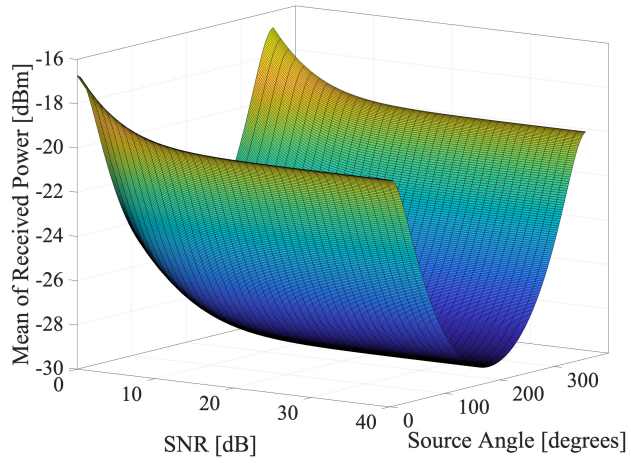
(a)



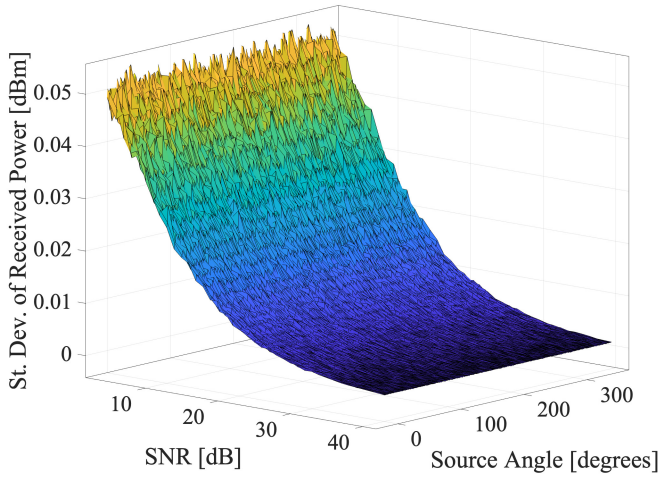
(b)

Figure 5.3: Illustration of the received power (in dBm) at the four receivers as a function of the source angle, ranging from 0 to 360 degrees with 10-degree increments. Panel (a) shows data from SimLocRF, while panel (b) presents data from the MeasLocRF.

remains within the sensitivity and dynamic range set by the subsequent neuromorphic preprocessing stage of the RF NeuroAS system, as detailed later in Sec. 5.1.3. The periodic change in signal strength reflects the controlled conditions of MATLAB simulations, corresponding with the source's changing positions relative to the four receivers. On the other hand, the received power from the MeasLocRF dataset depicted in Fig. 5.3(b) fluctuates between -20 and -31 dBm as the source rotates. These irregularities arise from imperfections in the anechoic chamber, such



(a)



(b)

Figure 5.4: Variation in (a) mean and (b) standard deviation of received power at Receiver 1, plotted against SNR (in dB) and source angle (in degrees). Data obtained from the SimLocRF dataset.

as the material properties and the effects of practical equipment elements like RF cables, supports, and anechoic foams. Although the antennas are designed to be omnidirectional, variations in their radiation patterns are evident as the source rotates, as demonstrated in Fig. 5.2(b). Despite these variances, the fundamental shape of the power distribution in the experimental dataset aligns closely with the simulated data.

Figure 5.4 depicts the variation in received power, from the SimLocRF dataset, at Receiver 1 across source angles from 0 to 360 degrees and SNR levels from 0 to 40 dB. Figure 5.4(a) illustrates the average received power impacted by noise levels, while Fig. 5.4(b) shows its standard deviation. The average power is the lowest when the source is furthest from Receiver 1 and increases as the source moves closer. Additionally, an increase in SNR corresponds with a reduction in the average received power, indicating sensitivity to noise levels. The standard deviation of received power remains relatively stable across various source angles, suggesting consistent signal variability. However, as the noise in the environment increases, indicated by a lower SNR, the variability in received power also intensifies.

5.1.3 . Neuromorphic Pre-Processing Performance

The neuromorphic pre-processing, also referred to as neuromorphic-enhanced wake-up radio (NWR), constitutes the third stage of the RF NeuroAS system (see Fig. 3.1 and Fig. 3.7 for details). NWR operates as a spike-rate encoding system that converts received RF signals into spike trains at a specific rate f_{spike}. This allows for the correlation of spike rates (f_{spike}) with received power (P_{RF}) and the identification of bit patterns from OOK-modulated RF signals, associating them with spike rates. These capabilities of NWR are demonstrated in Sec. 5.1.3.A and Sec. 5.1.3.B, respectively. The circuit design, which includes an envelope detector, a transconductance eSynapse, and an ML or LIF eNeuron, is thoroughly described in Sec. 3.3. Circuit Layouts for both ML-based and LIF-based NWR, which were developed using BiCMOS SiGe 55 nm technology from ST Microelectronics, are included in Appendix D. The ML-based NWR occupies a silicon area of 9.8 $\times$ 25.09 μm^2 and consumes 1.2 nW of power at a supply voltage of 200 mV, while the LIF-based NWR uses a 9.03 $\times$ 10.52 μm^2 of area and consumes only 0.25 nW of power at the same voltage.

A .Spike Rate and Received Power Correlations

To evaluate the NWR's ability to correlate spike rates with received power, it is essential to determine the minimum detectable signal, defined by the sensitivity of the envelope detector integrated within the NWR. The power of the minimum detectable signal (P_{mds}) can be expressed in dBm as [221]

$$P_{mds} = NF_{tot} + 10logB - 174 + SNR_{min}, \tag{5.1}$$

where NF_{tot} represents the total noise figure of the circuit, B is the bandwidth of the envelope detector, and SNR_{min} minimum signal-to-noise ratio necessary for reliable detection by the OOK modulation. For this analysis, $SNR_{min} = -12$ dB and $B = 10$ MHz are considered. Figure 5.5 shows the P_{mds} for both the ML-based NWR (blue line) and the LIF-based NWR (red line), as derived from (5.1). The sensitivity of the envelope detector is determined by where the received power (P_{RF}, black line) intersects the P_{mds}. In this setup, the ML-based NWR shows a

sensitivity of -29 dBm, while the LIF-based NWR exhibits a sensitivity of -25 dBm. Thus, these sensitivity levels establish the dynamic range limits of the NWR, which will be explored further in Sec. 5.1.3.C

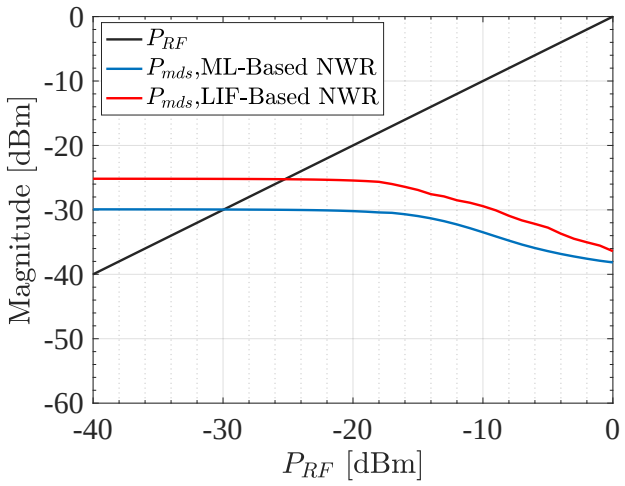


Figure 5.5: Sensitivity of the envelope detector within the ML-based and LIF-based NWRs: the black line is the P_{RF} and its cross with the blue and red lines are P_{mds} for both NWRs, respectively.

Figure 5.6 demonstrates the relationship between the received power and the output of each stage in the NWR, comparing both ML-based (blue line) and LIF-based (red line) configurations. Specifically, Figure 5.6(a) shows that the voltage at the envelope detector output, V_{ED}, increases exponentially with received power P_{RF}. Figure 5.6(b) depicts a decreasing trend between the synaptic current I_{trans} at the transconductance eSynapse output and P_{RF}. This decreasing relationship is achieved through the use of two complementary current mirrors in the eSynapse design, with the reason for this approach detailed in Sec. 5.1.3.B.

Figure 5.6(c) maps the correlation between f_{spike} and P_{RF}. For the ML-based NWR, f_{spike} decreases from 249 to 19 kHz as P_{RF} increases from -29 dBm to 0 dBm, remaining constant at 249 kHz below -29 dBm. Conversely, for the LIF-based NWR, f_{spike} drops from 65 kHz to 0.12 kHz as P_{RF} increases from -25 dBm to 0 dBm, and stabilizes at 65 kHz below -25 dBm. The limited dynamic range is evidenced by the obtained P_{mds} values (-29 dBm for ML-based and -25 dBm for LIF-based NWR), as depicted in Fig. 5.5. Additionally, spiking rates vary according to the type of eNeuron used; ML eNeurons demonstrate higher firing rates compared to LIF eNeurons, as shown in Tab. 2.2.

B .Bit Patterns Identification

To verify the ability of the NWR to identify bit patterns in RF signals, an example of an OOK-modulated RF signal is generated at the NWR's input. Figure 5.7 showcases this signal, focusing on the first three bits, "010", and also displays the outputs from each stage of the ML and LIF-based NWRs, from the envelope detector through the transconductance eSynapse to the final ML and LIF eNeuron stages. These results are derived from an average of eight transient

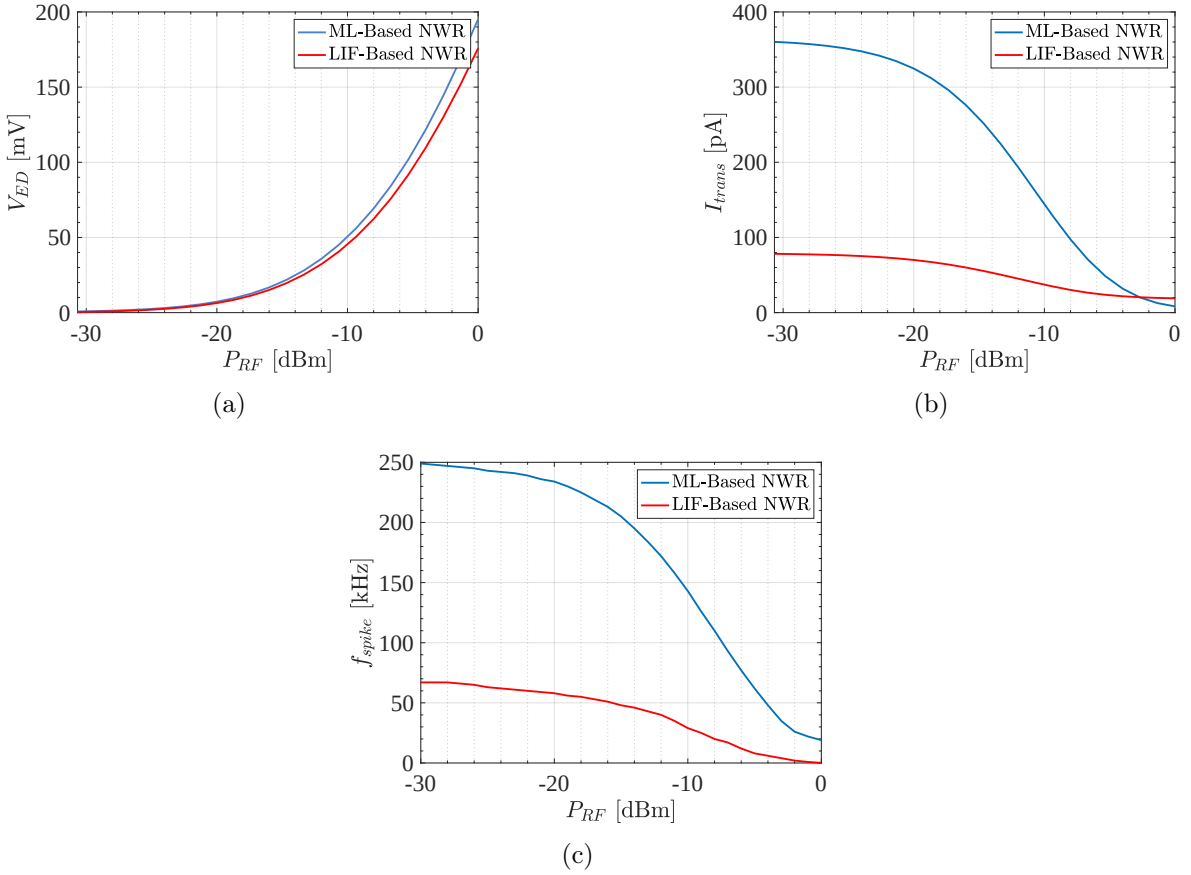


Figure 5.6: Post-layout simulation results showing the relationship between P_{RF} and the output of each stage in the NWR: (a) V_{ED} in mV, (b) I_{trans} in pA, and (c) f_{spike} in kHz. Results for both ML-based and LIF-based NWRs are in blue and red lines, respectively.

noise post-layout simulations. As depicted in Fig. 5.7, the input OOK RF signal operates at a frequency of 2.4 GHz, a data rate of 1 kbps, and a power level of P_{RF} = -10 dBm, displaying the sequence "010" over a 3 ms window. In this signal, the bit '0' corresponds to zero voltage, while the bit '1' is an RF sinusoidal signal with an amplitude of V_{RF} = -90 mV.

The RF signal is demodulated by the envelope detector, producing two distinct voltage levels corresponding to different bits: a consistently low, near-zero voltage for bit '0' (V_{ED} = 6 μV for both types of NWR, regardless of the received power P_{RF}), and a variable voltage for bit '1' (V_{ED} = -44 mV for ML-based NWR and V_{ED} = -50 mV for LIF-based NWR at P_{RF} = -10 dBm, fluctuating with P_{RF} as shown earlier in Fig. 5.6(a)). The increased V_{ED} for the LIF-based NWR compared to the ML-based NWR for bit '1' is due to an additional transistor M_c in the LIF-based design, which enhances the voltage gain. Given that the demodulated voltage for bit '0' remains constant across all received power levels, it is converted into a steady synaptic current by the transconductance eSynapse (I_{trans} = 382 pA for ML-based NWR and I_{trans} =

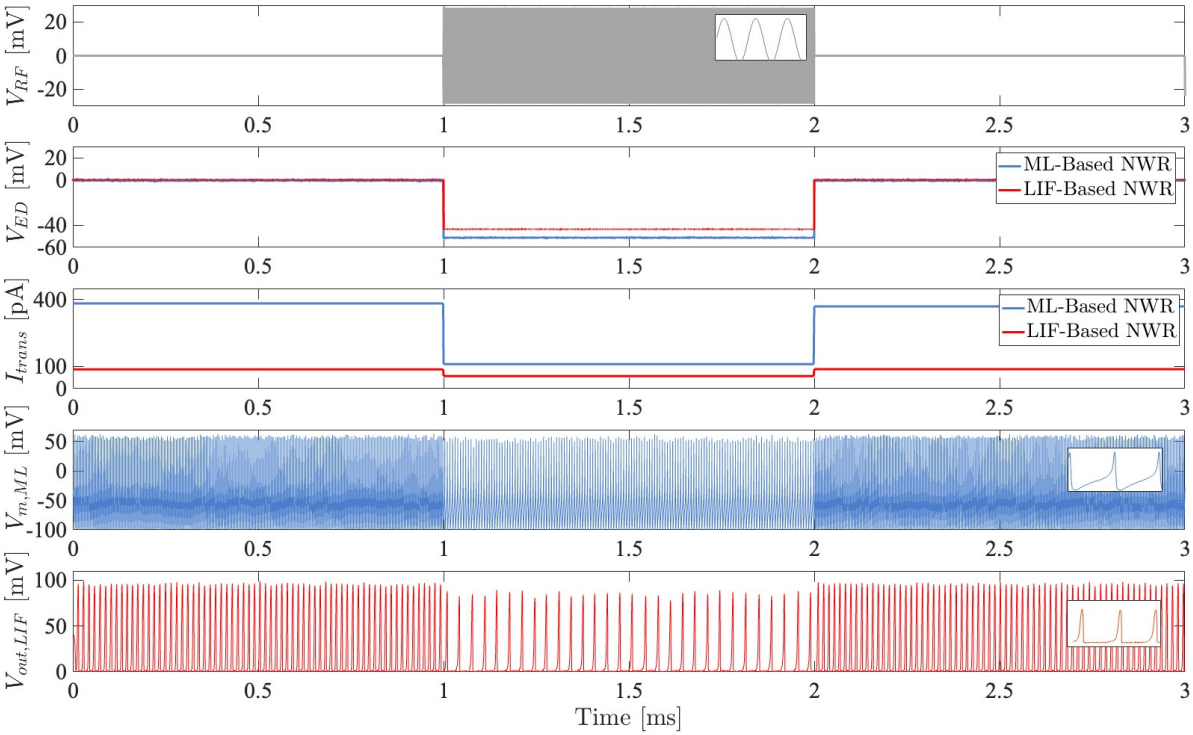


Figure 5.7: Transient noise post-layout simulation results showing the response of each stage of ML-based and LIF-based NWRs for an OOK-modulated RF signal at its first three bits "010", with a P_{RF} = -10 dBm. An inset is added for V_{RF}, $V_{m,ML}$, and $V_{out,LIF}$ to clarify the sinusoidal and spiking behaviors.

86 pA for LIF-based NWR). Conversely, for bit '1', the transconductance eSynapse excites the post-eNeuron differently, as it generates an average current value that varies with P_{RF}, as shown in Fig. 5.6(b). In the example shown in Fig. 5.7, at P_{RF} = -10 dBm, I_{trans} is 109 pA for ML-based NWR and 56 pA for LIF-based NWR.

Finally, the spiking behavior of the membrane voltage at the output of both ML and LIF eNeurons is illustrated in Fig. 5.7. The spiking frequency, f_{spike}, is determined by counting the number of spikes over a given time period (in this context, the duration of a bit is 1 ms). For bit '0' and at any P_{RF}, both ML and LIF eNeurons exhibit a constant f_{spike} due to the steady synaptic current; f_{spike} is 250 kHz for ML-based NWR and 65 kHz for LIF-based NWR. However, for bit '1', f_{spike} increases as P_{RF} decreases, a relationship that is illustrated in Fig. 5.6(c) and results from the decreasing relationship between I_{trans} and P_{RF}, as explained in Sec. 5.1.3.A. This design choice allows f_{spike} for bit '1' to reach a high, saturated level at the minimum detectable signal (P_{mds}), which aligns with the f_{spike} observed for bit '0'. This configuration is intended to optimize the energy efficiency (measured in fJ/spike) of the eNeuron when handling bit '0' scenarios, where the input voltage is zero.

C .Performance Metrics

The performance of the neuromorphic pre-processing varies depending on whether it is an ML-based or LIF-based configuration. To determine the most suitable type for the RF NeuroAS system, several performance metrics need to be considered.

Minimum Detectable Signal (P_{mds}) — State-of-the-art envelope detectors aim for narrowband matching to obtain low-noise performance, achieving a P_{mds} of approximately -50 dBm for high data rate designs [225]. Additionally, while lower P_{mds} values are achievable, they typically require higher μW-range power consumption. The ML-based NWR attains a P_{mds} of -29 dBm at a total power of 1.2 nW, whereas the LIF-based NWR reaches -25 dBm at just 0.24 nW, both at a 1 kbps data rate. To enhance the P_{mds}, implementing an off-chip narrowband matching network rather than metallic resistance matching could be advantageous, as detailed in Sec. 3.3.1.A.

Additionally, the performance of the NWR is assessed by displaying spiking trains within a 1 ms observation window per bit (data rate = 1 kbps). For higher data rates, the observation window could be reduced to 0.1 ms (data rate = 10 kbps) or even to 0.01 ms (data rate = 100 kbps), without any drawback on power consumption P_{rms}. This is not the case of the state-of-the-art where a low P_{rms} and a high P_{mds} are achieved but only for low data rates [223], [224]. Table 5.1 summarizes a comparison of the performance of envelope detectors across different technologies, operating frequencies, power consumption, and energy efficiency (in pJ/bit).

Dynamic Range — The ML-based NWR has an f_{spike} for bit '1' that decreases as input power ranges from -29 dBm to 0 dBm, then remains constant at lower power levels and matches that of bit '0' (f_{spike} = 250 kHz). Similarly, the LIF-based NWR shows an f_{spike} that decreases for bit '1' between -25 dBm and 0 dBm, and then stabilizes at the same rate as for bit '0' (f_{spike} = 65 kHz). With a system resolution of 1 kHz, the ML-based NWR is validated up to a P_{RF} of -29 dBm, while the LIF-based NWR is validated up to a P_{RF} of -25 dBm, corresponding to the P_{mds} for each configuration. Consequently, the LIF-based NWR exhibits a narrower dynamic range (25 dB) compared to the dynamic range of the ML-based NWR (29 dB).

Temperature Variation — To assess the robustness of the NWR against temperature variations, Fig. 5.8(a) illustrates the changes in f_{spike} of the ML-based NWR across a temperature range of 0 to 60 degrees for bit '1' signals at P_{RF} = -29 dBm (= P_{mds}), 0 dBm, and for bit '0'. Similarly, Fig. 5.8(b) displays the temperature-related variations in spiking frequency for the LIF-based NWR for bit '1' signals at P_{RF} = -25 dBm (= P_{mds}), 0 dBm, and for bit '0'. These figures demonstrate that, within this temperature range, the NWR system maintains consistent P_{mds} in both configurations, by preserving the differential between f_{spike} for bit '0' and f_{spike} for bit '1' when $P_{RF} = P_{mds}$.

Monte Carlo Simulations — The distributions of the spiking frequency (f_{spike}) for both ML-based and LIF-based NWRs are shown in Fig. 5.9(a) and Fig. 5.9(b) respectively. These graphs represent the outcomes of 500 iterations of transient post-layout simulations for a bit '1' RF input signal with the lowest power level ($P_{RF} = P_{mds}$ = -29 dBm for the ML-based NWR and -25 dBm for the LIF-based NWR). At these power levels, the ML-based NWR exhibits a f_{spike} of 249 kHz, whereas the LIF-based NWR shows a f_{spike} of 64 kHz, as detailed in Sec. 5.1.3.B. The frequency distributions in both configurations seem to adhere to a Poisson distribution, with the

highest probabilities aligning with the expected range of spiking frequencies.

Regarding performance, the ML-based NWR achieves a better P_{mds}, offering a higher dynamic range but at a greater power consumption compared to the LIF-based NWR. Consequently, the ML-based NWR was selected for the RF NeuroAS system due to its biological plausibility and its enhanced dynamic range, which improves the accuracy of the subsequent neural network stage.

Table 5.1: Envelope Detector Performance Comparison

Ref	[222]	[223]	[224]	[225]	[226]	This Work
Techn. (nm)	180	65	180	180	130	55
f_R (GHz)	2.4	0.434	0.113	2.4	0.9	2.4
P_{rms} (nW)	120	0.42	4.5	2400	5	1.2^a, 0.25^b
P_{mds} (dBm)	-48.5	-79.2	-69	-50	-26	-29^a, -25^b
E_{eff} (pJ/bit)	48	4.2	15	22.5	5	1.2^a, 0.25^b

^{a}ML-based NWR, ^{b}LIF-based NWR

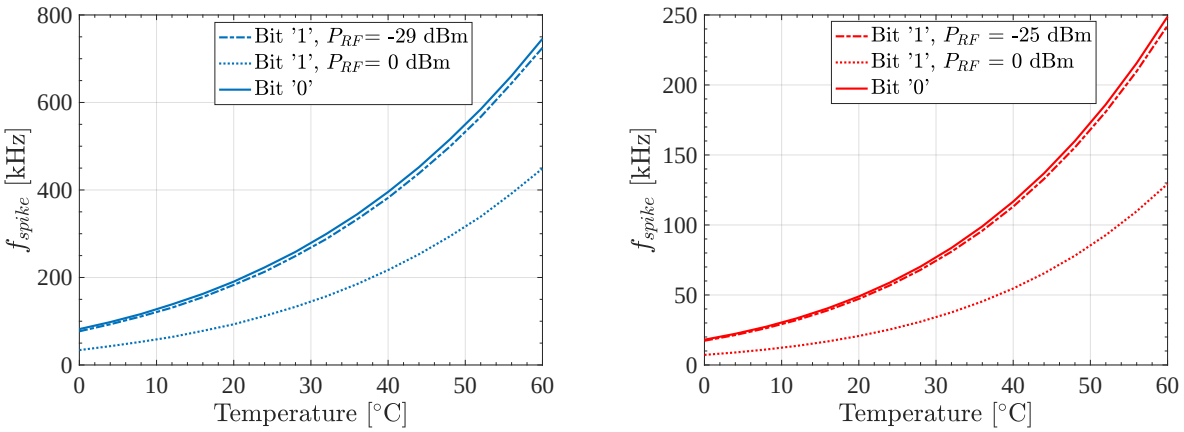


Figure 5.8: The variation of the spiking rate at the output of (a) ML-based NWR and (b) LIF-based NWR, in function of the temperature. Three cases are considered: a bit '1' signal with $P_{RF} = P_{mds}$, $P_{RF} = 0$ dBm, and a bit '0' signal.

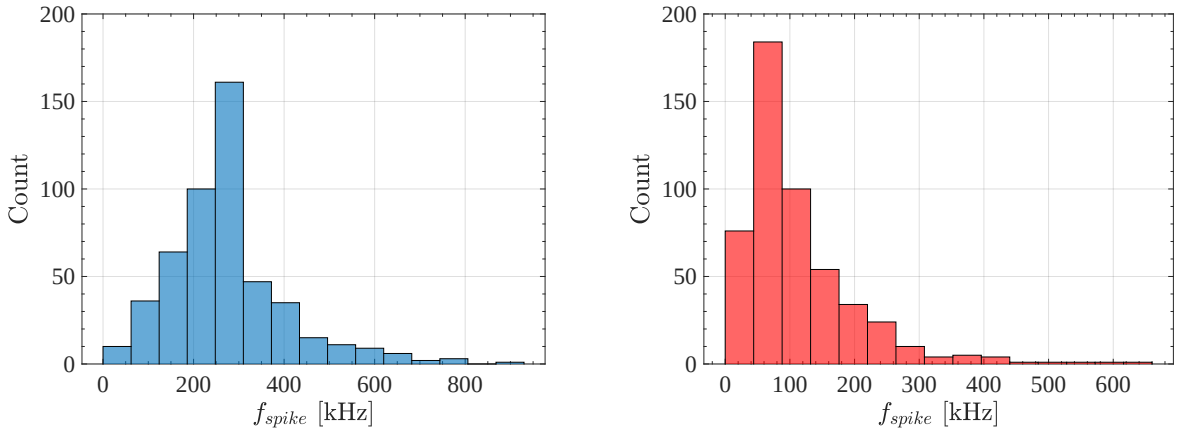


Figure 5.9: The distribution of the spiking frequency at the output of (a) ML-based NWR and (b) LIF-based NWR, for a bit '1' signal with $P_{RF} = P_{mds}$. Results are obtained for 500 iterations of transient PLS.

5.1.4 . Analog Spiking Neural Network Performance

The analog-based spiking neural network (A-SNN) represents the final component of the RF NeuroAS system, shown in Fig. 3.9 and detailed in Sec. 3.4. Its core function is to identify the source position accurately, characterized by its parameters r_s, θ_s, and d_s, which are elaborated upon in Sec. 3.2. For deep learning capabilities, the A-SNN uses a fitted activation function obtained from the post-layout transfer function of ML eNeurons, as detailed in Sec. 3.4.2. Therefore, the A-SNN is trained and tested on the SimLocRF and MeasLocRF datasets, using a specialized methodology designed for deep learning that accounts for analog characteristics, as described in Sec. 4.1.3. To assess the functionality of the A-SNN, Sec. 5.1.4.A first validates the A-SNN setup by examining the suitability of the fitted activation function and evaluating the training process across different noise levels. Following this, Sec. 5.1.4.B examines the precision of the A-SNN in source localization at 10-degree resolution.

A .Neural Network Setup

Figure 5.10 depicts three potential fits for the post-layout activation function: a piecewise polynomial fit of 2^{nd} order (represented by an orange dashed line), a polynomial fit of 12^{th} order (shown as a green dashed line), and a sigmoid fit (illustrated with a red dashed line). The accuracy of these models is assessed using the R-squared (R^2) metric, which quantifies the fit quality. The R^2 values achieved are 0.998 for the piecewise polynomial, 0.997 for the polynomial, and 0.981 for the sigmoid fit.

The piecewise polynomial fit, with its highest R^2 score, or an interpolation function, could be preferable. However, they require adjustments for smooth transitions, especially at the boundary point of 0, where the derivative tends to infinity, resulting in a discontinuity of the function. The sigmoid fit, though simpler and more suitable for classification tasks, achieves an acceptable fit.

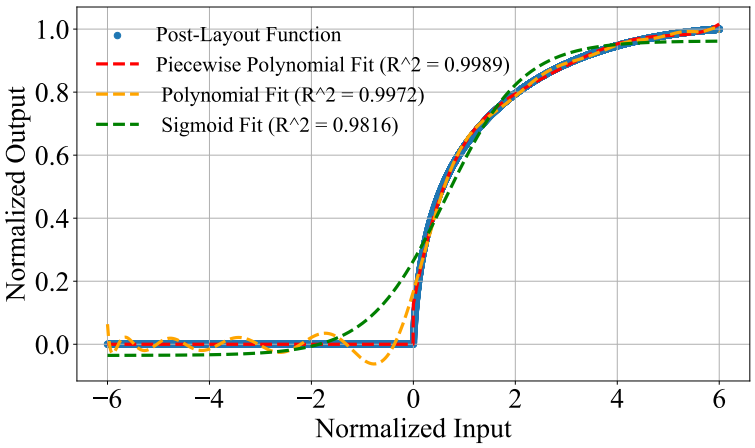


Figure 5.10: Post-layout activation function, with three fitted models: piecewise polynomial, polynomial, and sigmoid functions.

Despite the complexity of a 12^{th} order equation, the polynomial fit offers a high R^2 score. This study opts for the polynomial fit based on multiple training sessions of the A-SNN, where it consistently resulted in the highest accuracy.

Figure 5.11 shows the progression of accuracy for the A-SNN using two different activation functions: the fitted post-layout activation function (blue line) and the standard sigmoid activation function (orange line). The graph shows the accuracy trends over 100 epochs during both the training (solid lines) and validation (dashed lines) phases. Both activation functions allow the A-SNN to surpass 96% accuracy after 100 epochs, but the improvement in accuracy with the fitted function is noticeably slower. This slower rate of progression is likely due to the increased complexity in optimizing weights when simulating analog behaviors with the fitted function. The model plots for the fitted activation function, derived from the transfer functions of various eNeurons and described in Sec. 4.1.3, are thoroughly presented later in Sec. 5.2.1.

The A-SNN is initially trained using subsets from either the SimLocRF or MeasLocRF datasets and later evaluated on unseen subsets. Both training and testing are conducted across various noise levels (SNR = 0, 10, or 20 dB). Proper training is essential for the network to perform effectively on unseen data, necessitating an analysis of various noise levels to determine the optimal one for robust neural network training. Table 5.2 illustrates the changes in angular accuracy from (3.7) of the A-SNN across different training and testing scenarios, categorized by SNR levels. The network is trained with datasets characterized by three distinct SNR setups: (a) a combined SNR incorporating 20, 10, and 0 dB, (b) an exclusive SNR of 20 dB, and (c) an exclusive SNR of 0 dB. Following training, the network is tested against unseen data at SNR levels of 20, 10, and 0 dB.

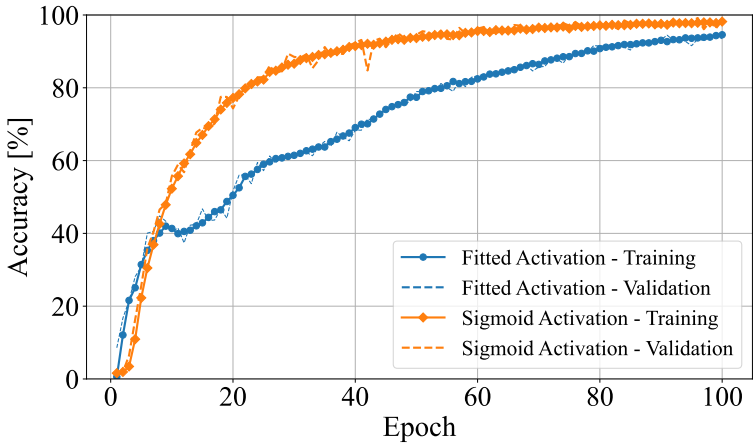


Figure 5.11: Accuracy progression for the A-SNN trained using the 12^{th} order polynomial fitted activation function (blue line) and a sigmoid activation function (orange line). Solid lines represent training accuracies, while dashed lines indicate validation accuracies over 100 epochs.

Results from Tab. 5.2 show that training with either the combined SNR dataset or the 20 dB SNR dataset leads to high accuracy in tests at 20 and 10 dB, but significantly lower accuracy at 0 dB (67.2% and 49.2%, respectively). However, training the network with a dataset at 0 dB SNR consistently yields high accuracy (over 90%) across all testing scenarios. This demonstrates that training with noisier data considerably improves the network's ability to handle various testing conditions. This relates to the eNeuron's capacity to mitigate external noise, which is defined at various SNR levels in the dataset. As a result, the A-SNN was purposefully trained using the dataset at an SNR of 0 dB to enhance its overall robustness.

Table 5.2: Angle Accuracy (%) for Different Training and Testing SNR Levels (dB)

Training SNR (dB)	Testing SNR (dB)		
	20 dB	10 dB	0 dB
20, 10, 0	97.6%	94.2%	67.2%
20	98.4%	92.7%	49.6%
0	97.1%	95.6%	90.5%

B .Performance Metrics

The network's performance is assessed using accuracy and normalized angle error from (3.7) and (3.8), respectively. The A-SNN determines the source position by its distance from the origin (d_s), region on the plane (r_s), and angle (θ_s) with a 10-degree resolution. Three scenarios are used to evaluate network performance: (a) both training and testing are conducted using the SimLocRF dataset from MATLAB, (b) both phases are performed with the MeasLocRF dataset from anechoic chamber measurements, and (c) training is done on SimLocRF data, and testing on MeasLocRF data. When using either SimLocRF or MeasLocRF, 70% of the data is

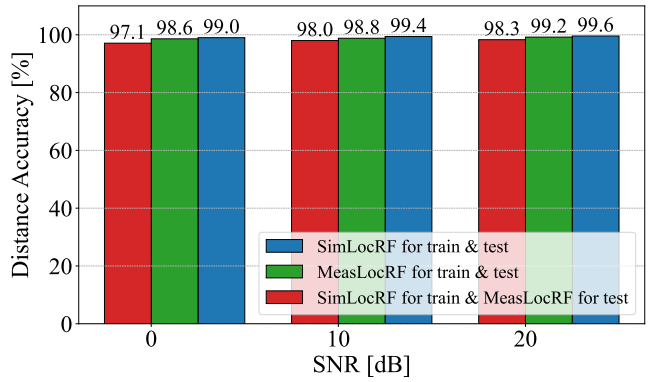
allocated for the training and validation phase, while the remaining 30% is used for testing. The performance is reported as the average of evaluations over 20 tests, using the statistical trained weights, as described in Alg. 4.1

Distance and region accuracies across three different SNR levels (0, 10, and 20 dB) are depicted for each scenario in Fig. 5.12(a) and Fig. 5.12(b), respectively. Distance accuracy remains high (above 97%) across all SNR levels due to the network's ability to distinguish among three distance options ($d_s = 0.1$ m, 0.3 m, 0.5 m). Similarly, region accuracy consistently exceeds 96%, given four possible outcomes, across all scenarios and SNR levels.

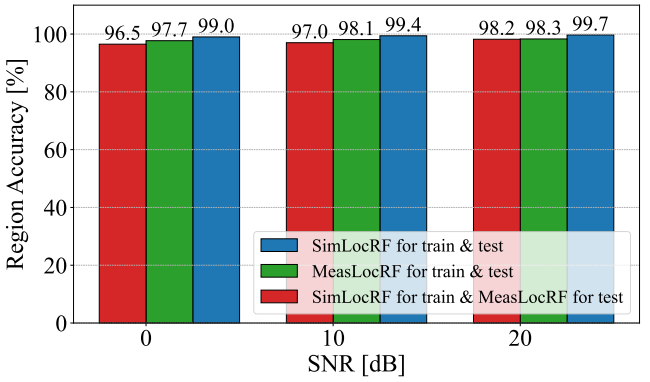
Regarding the angular detection of the source, as shown in Fig. 5.12(c), its accuracy tends to decrease with lower SNR levels, indicative of noisier conditions. The network reaches its peak angular accuracy of 96.7% when it is trained and tested using the SimLocRF dataset, benefiting from the structured and symmetrical nature of MATLAB-generated data, as detailed in Sec. 3.2.1. The angle accuracy drops when training and testing occur with the MeasLocRF dataset, which includes more variability due to the experimental setup. However, it still remains relatively acceptable with the lowest recorded accuracy at 86.5% for 0 dB SNR. In cases where the network is trained on SimLocRF data and then tested on new, unseen MeasLocRF data, accuracy declines further. However, it remains within acceptable bounds, with the lowest point at 80.4% for 0 dB SNR. Despite these challenges, the results confirm the neural network's capacity to effectively localize the source even when exposed to new and varied datasets.

Figure 5.13 presents the normalized angle error (NAE) results for the A-SNN across three different scenarios at various SNR levels. At a 20 dB SNR, the lowest NAE values recorded (0.05, 0.07, and 0.12 for the first, second, and third scenarios respectively) demonstrate high precision in angle estimation. Furthermore, even at the challenging 0 dB SNR level, the highest NAE values are 0.11, 0.15, and 0.22 for the respective scenarios, maintaining the 10-degree resolution of the network.

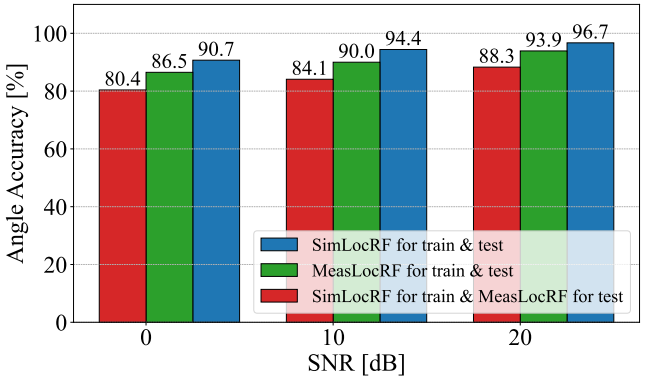
Given that each neuromorphic pre-processing unit consumes 1.2 nW, the ML eNeuron consumes 0.9 nW, and the eSynapse consumes 0.4 nW [24], the RF NeuroAS system's power consumption can be estimated. Following the methodology outlined in [23], the RF NeuroAS system, comprised of 4 preprocessing units, 56 ML eNeurons, and 444 eSynapses, consumes only 233 nW of power. These results demonstrate the system's robustness in performance and its ability to maintain a simple, low-power architecture, well-suited for efficient RF localization.



(a)



(b)



(c)

Figure 5.12: Performance of the A-SNN at 10-degree resolution in terms of (a) distance accuracy (%), (b) region accuracy, and (c) angle accuracy (%) of the source position, for three different scenarios of training and testing, and for three SNR levels.

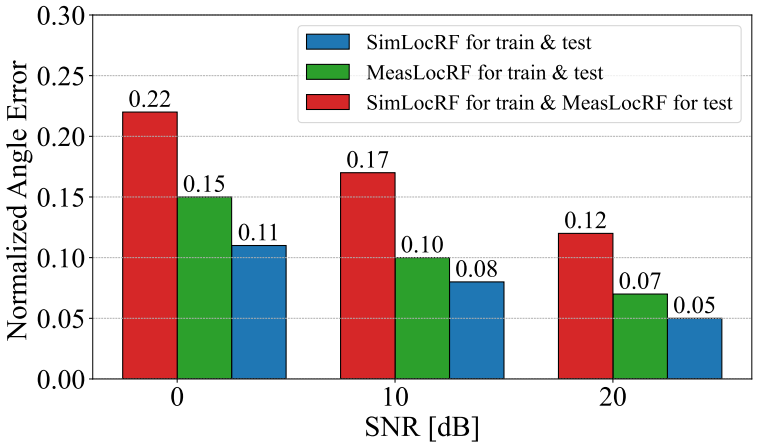


Figure 5.13: Performance of the A-SNN at 10-degree resolution in terms of the normalized angle error (in degrees) of the source position, for three different scenarios of training and testing, and for three SNR levels.

5.1.5 . Simplified RF NeuroAS System: Post-Layout Results

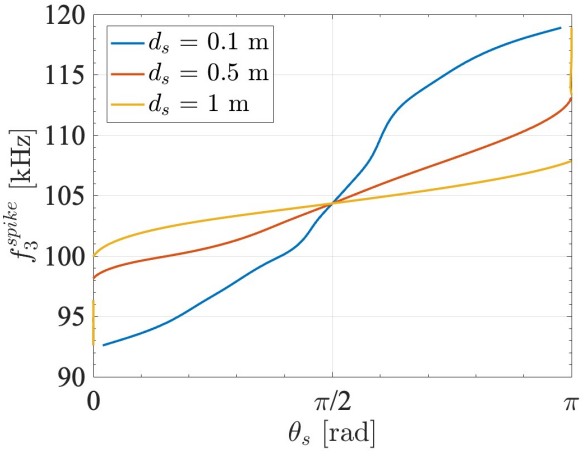
The simplified version of the RF NeuroAS system is fully implemented in an analog circuit using BiCMOS 55 nm technology (see details in Sec. 3.5). Its layout, shown in Appendix D, occupies 18.3 x 20.3 μm^2, consumes only 1.1 nW from a 200 mV power supply, and achieves a dynamic range of 30 dB. This system is capable of determining the position of a source located in the positive half of the plane, with possible angles θ_s ranging from 0 to π and distances d_s of 0.1, 0.5, or 1 meter. As previously presented in Fig. 3.10, the simplified RF NeuroAS system includes two stages of neuromorphic pre-processing (NWRs), each linked to a receiver, and a neuromorphic computing unit that calculates the output spike rate corresponding to the source's position.

Figure 5.14 illustrates the post-layout variations in output spiking rate (Fig. 5.14(a)) and energy efficiency (Fig. 5.14(b)) according to different source positions, which are determined by their angle θ_s and distance d_s. As depicted in Fig. 5.14(a), if the source is close to the receivers (e.g., $d_s = 0.1$ m), then f_{spike} fluctuates between 92 and 119 kHz. Conversely, if the source is farther from the receivers (e.g., $d_s = 1$ m), then f_{spike} shows a narrower variation, ranging from 100 to 108 kHz. Given that the system's resolution is set at 1 kHz for an observation window of 1 ms, the range of spike rates impacts the angular resolution the system can detect. Here, the angular resolution is 5, 12, and 20 degrees for distances of $d_s = 0.1$, 0.5, and 1 m, respectively. This illustrates that the angular resolution diminishes as the spike rate range narrows, indicative of the source being further away. Extending the observation window of the spiking behavior can improve the system's resolution, thereby enhancing angular resolution.

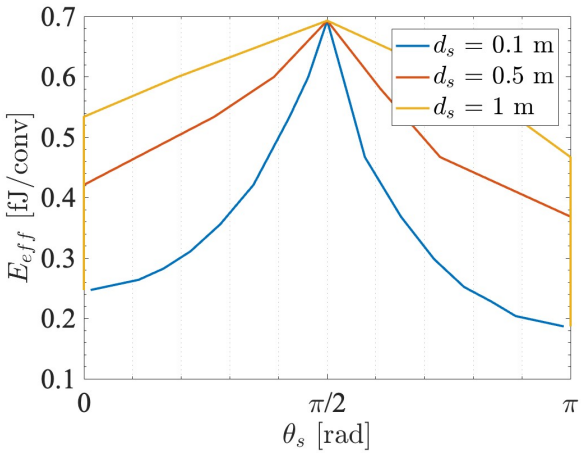
Figure 5.14(b) illustrates the energy efficiency of the simplified RF NeuroAS system, calculated using Walden's figure-of-merit [227], as defined by the equation in fJ/conv

$$E_{eff} = \frac{P_{rms}}{f_3^{spike} \cdot 2^N}, \tag{5.2}$$

where P_{rms} is the total power consumption of the system and N is the full-scale dynamic range. The energy efficiency ranges from a minimum of only 0.7 fJ/conv when the source is far from the receivers to a maximum of 0.2 fJ/conv when the source is close to either receiver R_1 or R_2.



(a)



(b)

Figure 5.14: Post-layout variations in (a) output spike rate and (b) energy efficiency of the simplified RF NeuroAS, according to source positions determined by θ_s and d_s.

5.2 . Advancements in Learning Techniques for Analog Spiking Neural Networks

The analog spiking neural network (A-SNN) primarily comprises eNeurons and eSynapses, leveraging spike coding in an analog circuit design for high energy efficiency. A comprehensive feasibility study and synthesis of the A-SNN are presented in Chap. 4 to guarantee its proper implementation and effective usage. Within this study, two learning techniques are evaluated for the A-SNN: deep learning (Sec. 4.1) and time learning (Sec. 4.2). The results from applying these techniques to the A-SNN are presented in Sec. 5.2.1 and Sec. 5.2.2, respectively.

5.2.1 . Deep Learning Impact on A-SNN

This section details the results of applying deep learning technique to the A-SNN. Section 5.2.1.A discusses the outcomes of the feasibility analysis conducted in Sec. 4.1.2, which tested deep learning on the theoretical model of the A-SNN. These results determine the feasibility of deep learning based on the transfer functions of various eNeuron types used within the A-SNN. When deep learning proves feasible for the A-SNN, a synthesis framework is outlined in Sec. 4.1.3, which requires appropriate fits for its activation function. Section 5.2.1.B then presents the results of fitting activation functions specifically for the A-SNN using deep learning, taking into account the various eNeuron types. Finally, Sec. 5.2.1.C highlights the A-SNN's performance in solving the MNIST problem, described in Sec. 4.1.4, by incorporating the deep learning synthesis framework.

A .Feasibility Results

The feasibility study detailed in Sec. 4.1.2 determines that deep learning can be applied to the A-SNN if either or both the eNeurons and eSynapses exhibit non-linear transfer functions. Accordingly, the transfer functions linking the output spike rate to the input synaptic current for eNeurons, and the output synaptic current to the input spike rate for eSynapses, are illustrated in Fig. 5.15. These results are obtained from post-layout simulations conducted using the BSIM4 model. The tested eNeurons include the b-ML (Fig. 5.15(a)), the s-ML (Fig. 5.15(b)), and the AH-LIF (Fig. 5.15(c)) eNeurons, along with the excitatory eSynapse (Fig. 5.15(d)), all of which are described in the spike rate-based model in Sec. 4.1.1. The linearity of these transfer functions is assessed using the least mean squares r^2 method from MATLAB tools. If a precise linear fit is established for any of the eNeurons or eSynapse under test, they are deemed unsuitable for use in deep learning applications. Alongside the transfer functions, the energy efficiency of the eNeurons, calculated as $E_{eff} = P_{rms}/\ f_{spike}$ (in fJ/spike), is highlighted in blue.

eNeurons Results — The transfer functions and energy efficiency of the eNeurons are obtained from a synaptic current I_{syn} sweep ranging from 0 to 1 nA. As depicted in Fig. 5.15, each eNeuron type exhibits a unique dynamic range of spike rate (f_{spike}) response, characterized by saturation (as observed in b-ML and s-ML eNeurons) or a drop (as in the AH-LIF eNeuron) at certain I_{syn} levels. For $I_{syn} > 200$ pA, b-ML and s-ML eNeurons (shown in Fig. 5.15(a) and Fig. 5.15(b)) demonstrate high energy efficiency and satisfactory linear fits for their transfer functions ($r^2 = 0.99$), highlighted by a dashed red line. However, at $I_{syn} < 200$ pA, these eNeurons exhibit lower energy efficiency and non-linear transfer function characteristics. These

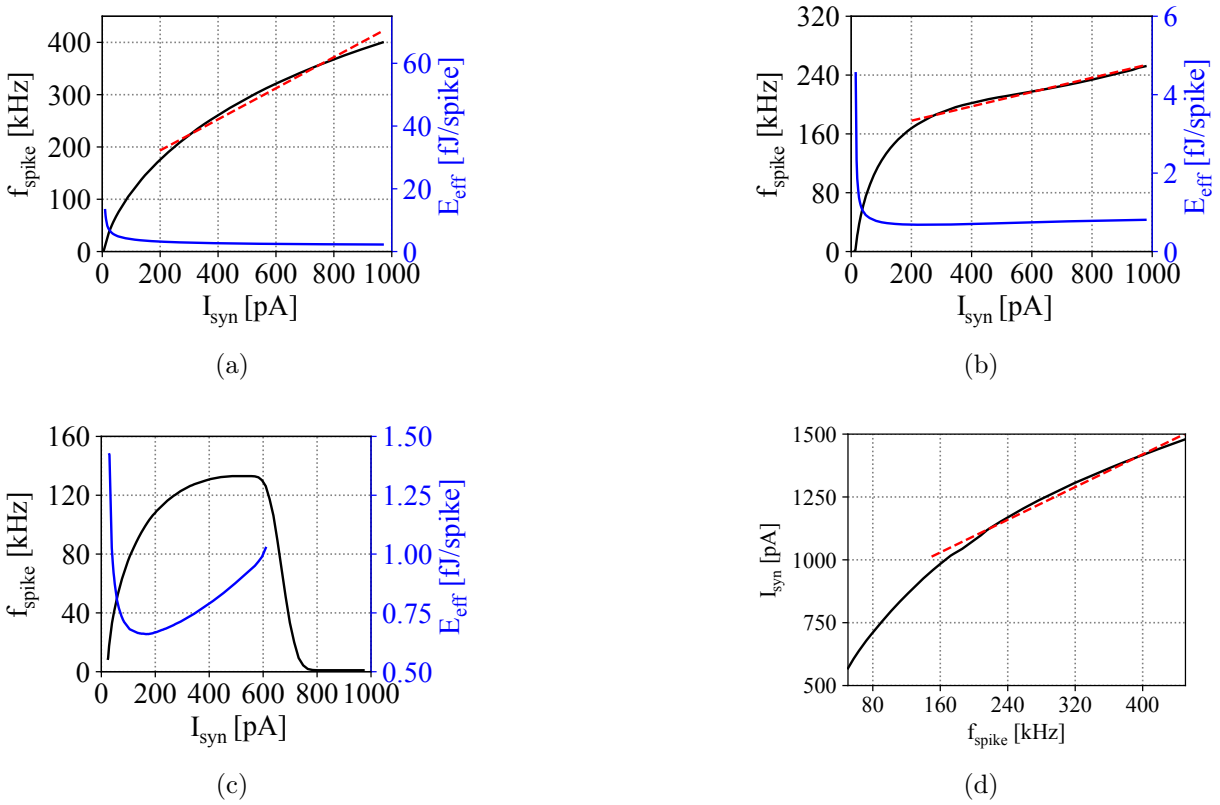


Figure 5.15: Post-layout transfer functions of (a) b-ML, (b) s-ML, (c) AH-LIF eNeurons, and (d) excitatory eSynapse (in black line). The linear fits of the transfer functions are presented in dashed red lines, and the energy efficiency (in fJ/spike) in the blue line.

observations illustrate a trade-off between energy efficiency and the potential for deep learning when using b-ML and s-ML eNeuron types. Design choices affect the operational scope and performance: using the full rate range of eNeurons results in lower energy efficiency, but enables the application of deep learning techniques. Conversely, restricting to higher f_{spike} ranges yields high energy efficiency and linear activation function, which limits the system to basic neural network functionalities unless non-linear transfer functions from eSynapses are considered.

The AH-LIF eNeuron, depicted in Fig. 5.15(c), displays a narrower dynamic range compared to b-ML and s-ML eNeurons and lacks a satisfactory linear fit for its transfer function. It maintains high energy efficiency across its range, peaking at only 1.5 fJ/spike. Therefore, using AH-LIF eNeurons offers the benefits of non-linearity, suitable for deep learning, and high energy efficiency. However, this comes at the expense of a reduced dynamic range, which may limit the performance of A-SNNs that use AH-LIF eNeurons, as further discussed in Sec. 5.2.1.C.

eSynapse Results — The transfer function of the eSynapse is determined through a spike rate sweep of the pre-synaptic eNeuron, ranging from 0 to 450 kHz, which represents the widest dynamic range achieved by the b-ML eNeuron. As depicted in Fig. 5.15(d), the output current of the eSynapse increases non-linearly with the input frequency. Within the range where eNeurons

exhibit a $f_{spike} > 170$ kHz and demonstrate high E_{eff}, a linear fit for the eSynapse's transfer function can be achieved with an $r^2 > 0.99$. This linear fit limits the potential for deep learning. This indicates that deep learning and energy efficiency are often mutually exclusive in an A-SNN that incorporates these specific eNeurons and eSynapses.

B .Activation Function Fit Results for Synthesis

For synthesizing A-SNNs using deep learning, their activation functions are derived from the post-layout transfer functions of eNeurons and eSynapses, as explained in Sec. 4.1.3. These activation functions require fitted models for use in TensorFlow for training and testing purposes. Two suitable fitted models are identified: sigmoid and polynomial fits. Figure 5.16 presents the fitting results for the standard activation function of the A-SNN, featuring the b-ML eNeuron (Fig. 5.16(a)), the s-ML eNeuron (Fig. 5.16(b)), and the AH-LIF eNeuron (Fig. 5.16(c)). The figure displays the normalized results, represented in arbitrary units (a.u.): the post-layout activation function in a solid black line, the sigmoid fit in a dashed blue line, and the polynomial fit in a dashed red line. A green vertical bar in the figure indicates a restriction to high energy efficiency by limiting the dynamic range of the activation function when I_{syn} exceeds 200 pA.

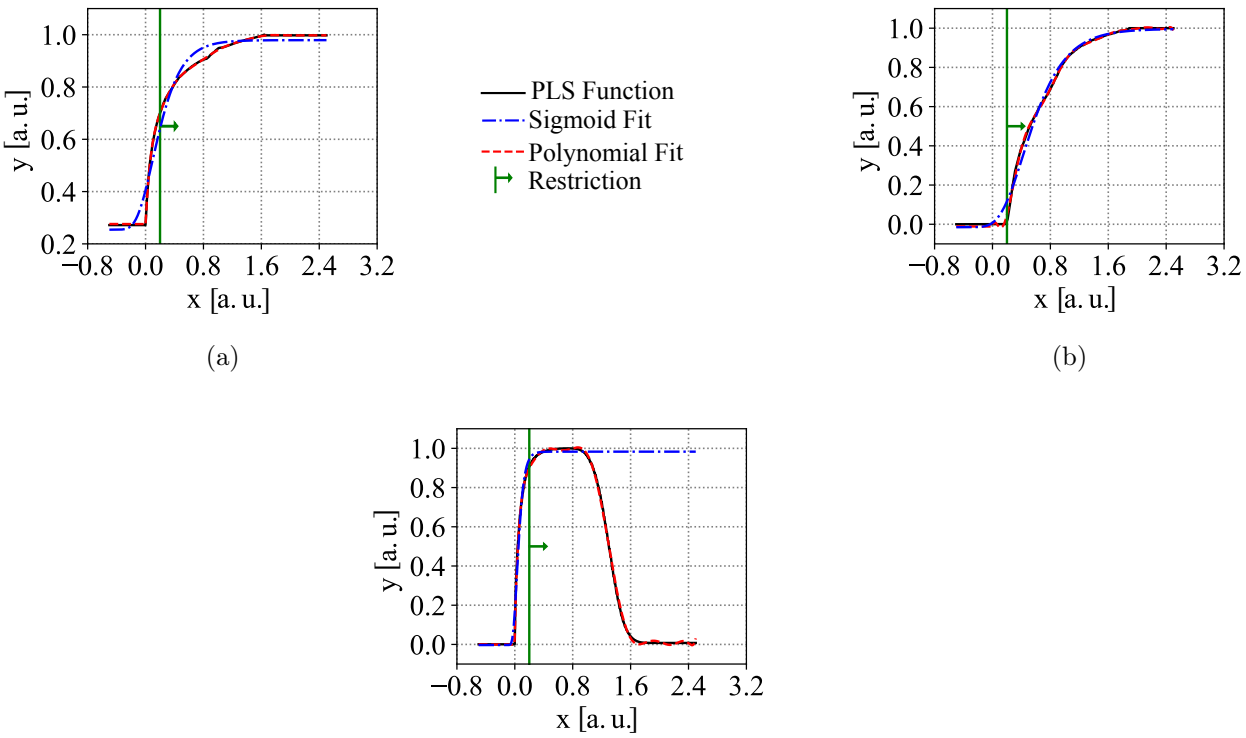


Figure 5.16: Post-layout fitting results for the standard activation function of the A-SNN, using (a) b-ML, (b) s-ML, and (c) AH-LIF eNeurons. A green vertical bar represents the dynamic range restriction.

This approach, discussed in Sec. 5.2.1.A, involves applying a threshold of $x \geq 0.2$ a.u. for both polynomial and sigmoid fit models.

The sigmoid fit, derived from the generalized logistic function [216], is used to fit the post-layout activation function with a moderate and differentiable curve. It achieves a high precision of $r^2 \geq 0.985$ with the b-ML eNeuron and $r^2 \geq 0.995$ with the s-ML eNeuron, but an unsuitable $r^2 \geq 0.158$ with the AH-LIF eNeuron. This latter is due to discrepancies when $x \geq 0.8$ a.u., necessitating a limitation on the used range. The polynomial fit applied to the post-layout activation function features a complex 12^{th} order. Across the three eNeuron types, it closely matches the behavior ($r^2 \geq 0.999$), accurately capturing variations and maintaining consistency in static regions. However, this precise fit may pose training challenges due to abrupt changes in slope near 0. In fact, stochastic gradient descent optimizers (like Adam) use the function's derivative for training, and such discontinuities reduce the convergence of the training algorithm.

Therefore, for training the A-SNN with any of the three eNeuron types, the sigmoid fit is recommended due to its high precision and differentiability at all points. For testing the A-SNN, both sigmoid and polynomial fits are suitable options; however, the polynomial fit is preferred for its closer alignment with the A-SNN's activation function behavior. Regarding the need for higher E_{eff}, restricting the input to the activation functions decreases the level of non-linearity, which may affect the A-SNN's accuracy.

C .Performance on MNIST Problem

The performance of the A-SNN was assessed using the MNIST problem, incorporating the three possible eNeuron types and the excitatory eSynapse. The multi-layered structure of the A-SNN is elaborated in Sec. 4.1.4, which adopts the synthesis framework outlined in Sec. 4.1.3. The training and the testing of the A-SNN are performed with all eNeuron types. Here, two scenarios are considered: one using the full input range and another using a restricted range for significantly higher E_{eff}, indicated by a green vertical bar in Fig. 5.16.

In both scenarios, the A-SNN is trained for 100 epochs using the sigmoid fit model of its post-layout activation function, with results detailed in Sec. 5.2.1.B. After training, synaptic weights are set as statistical variables, taking into account the process variability and the current-mirror mismatch in the eSynapse. The trained A-SNN, incorporating these weights, is then tested using the polynomial fit model of the activation function to precisely evaluate the A-SNN's performance considering realistic variations in circuit parameters.

Figure 5.17 shows the accuracy results for the A-SNN in solving the MNIST problem with the three eNeuron types: b-ML eNeuron in Fig. 5.17(a), s-ML eNeuron in Fig. 5.17(b), and AH-LIF eNeuron in Fig. 5.17(c). The figure shows training performance over epochs with a blue dashed line for the non-restricted input and a green dashed line for the restricted input. Testing performance is marked with red dots for the non-restricted case and green dots for the restricted case, including error bars to illustrate the distribution of the accuracy.

Considering the A-SNN designed with b-ML eNeurons, Fig. 5.17(a) illustrates that in the non-restricted scenario, the training accuracy rapidly improves up to epoch 40 and then gradually rises to 0.85. The testing accuracy experiences a slight decrease of only 3% compared to training accuracy, as the sigmoid and polynomial fits are closely aligned in this scenario. Conversely, restricted training shows more erratic improvement up to an accuracy of 0.7, with significant

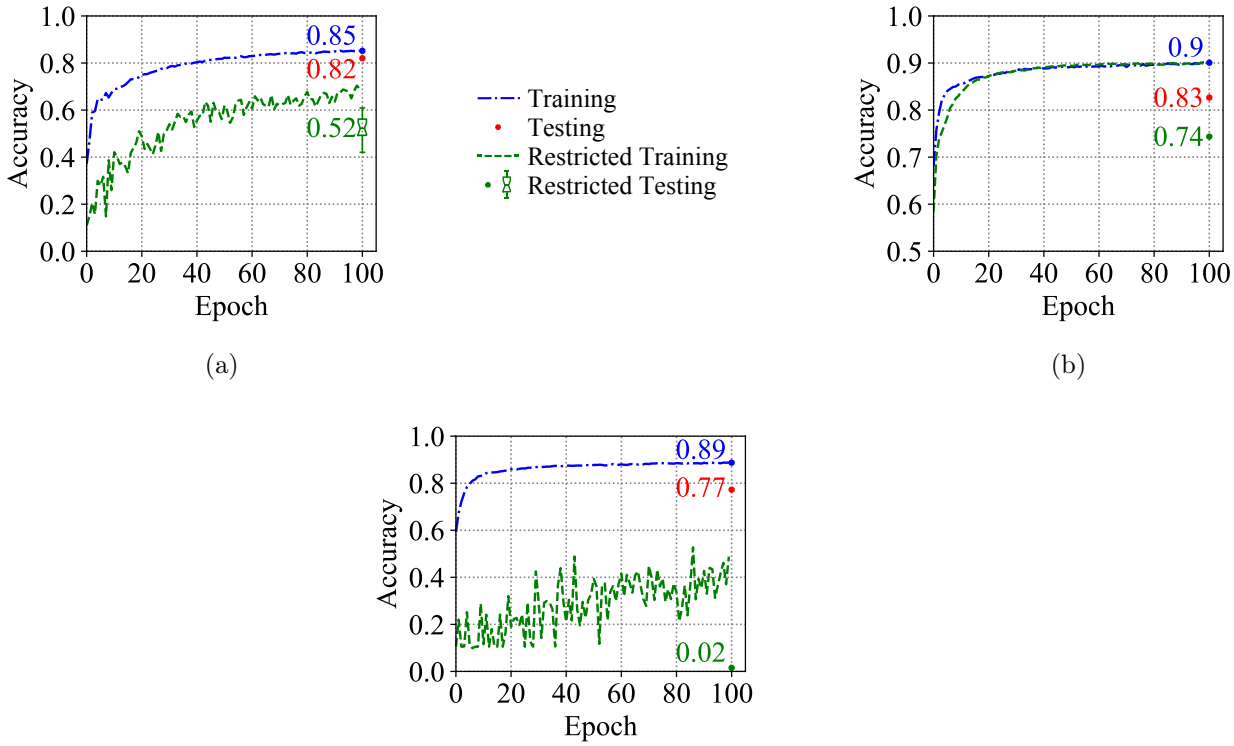


Figure 5.17: Accuracy results for the A-SNN in solving the MNIST problem, designed with (a) b-ML, (b) s-ML, and (c) AH-LIF eNeurons. Restricted and non-restricted scenarios are considered for both training and testing.

drops during testing, averaging a loss of 0.52 and a standard deviation of 0.12.

For the A-SNN designed with s-ML eNeurons, as shown in Fig. 5.17(b), both non-restricted and restricted scenarios exhibit substantial training accuracy improvements up to epoch 40, reaching 0.9 by epoch 100. In this type of A-SNN, the restriction has minimal impact on the non-linearity of the activation function, as discussed in Sec. 5.2.1.B. The testing accuracy falls by 7% in the non-restricted scenario to 0.83, and by over 16% in the restricted case to 0.74.

In the design of A-SNN using AH-LIF eNeurons, as shown in Fig. 5.17(c), the non-restricted scenario shows rapid training improvement until epoch 20, achieving an accuracy of 0.89, which then decreases during testing by 12%. For the restricted case, training performance is erratic, only reaching an accuracy lower than 50% by epoch 100. This lack of convergence likely results from the limited dynamic range and an unsuitable fitting model (see details in Sec. 5.2.1.B).

This analysis demonstrates the trade-off between deep learning and energy efficiency, as outlined in Sec. 5.2.1.A, where achieving higher energy efficiency results in a drop in accuracy due to limitations in deep learning capabilities. To maintain high accuracy in the A-SNN while ensuring high energy efficiency, it is essential to select the suitable eNeuron model and the appropriate fitting model.

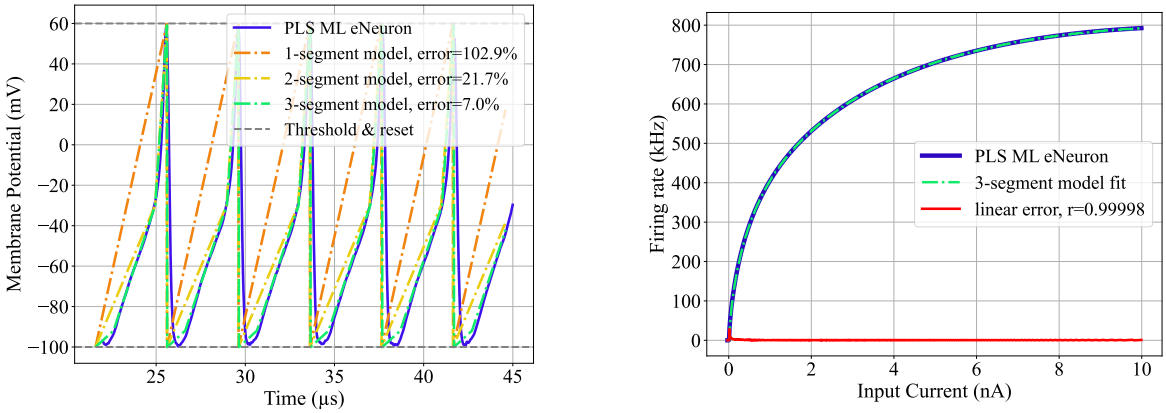
5.2.2 . STDP Learning Impact on A-SNN

This section outlines the application of STDP learning to the A-SNN and details the outcomes. Section 5.2.2.A presents the biomimetic ML (b-ML) eNeuron model results from Sec. 4.2.1, focusing on spike shape and firing rate properties, and from Sec. 4.2.2 for the noise property. Section 5.2.2.B presents the post-layout simulations results that examine the impact of noise on spike timing across various eNeuron circuits. These findings assess the viability of STDP learning in A-SNNs that include these eNeurons. Section 5.2.2.C discusses the performance of the A-SNN in solving XOR and MNIST problems, as described in Sec. 4.2.3.

A .eNeuron Model Results

This section presents the validation results for the b-ML eNeuron model. Although the model is adaptable to other types of eNeurons, the focus here is on this specific eNeuron because it was used for training the A-SNN with STDP due to its high biological plausibility. The b-ML eNeuron model is characterized by three main properties: spike shape, firing rate response, and noise-driven rise deviation, as described in equations (4.10), (4.11), (4.12), (4.16), (4.17), and (4.22), respectively.

Figure 5.18(a) illustrates the spike shape fitting for the b-ML eNeuron, using different levels of segmentation (details in Sec. 4.2.1 and (4.11)): one (orange plot), two (yellow plot), and three (green plot), compared with the post-layout (PLS) spike shape (blue plot). This figure demonstrates that increasing the segmentation fineness enhances the model's accuracy, reducing the error from 102.2% to 7%. For ongoing work on solving XOR and MNIST problems, the model with three segmentations was selected due to its acceptable error margin in approximating the actual spike shape of the b-ML eNeuron. Further refinement in segmentation could reduce the error margin to negligible levels, although it would also increase the computational time when



(a) (b)

Figure 5.18: (a) Action potentials of b-ML eNeuron, showing post-layout spike response and model fits, (b) Firing rate response of b-ML eNeuron versus Brian2 model across input currents. Correlation and models errors are highlighted.

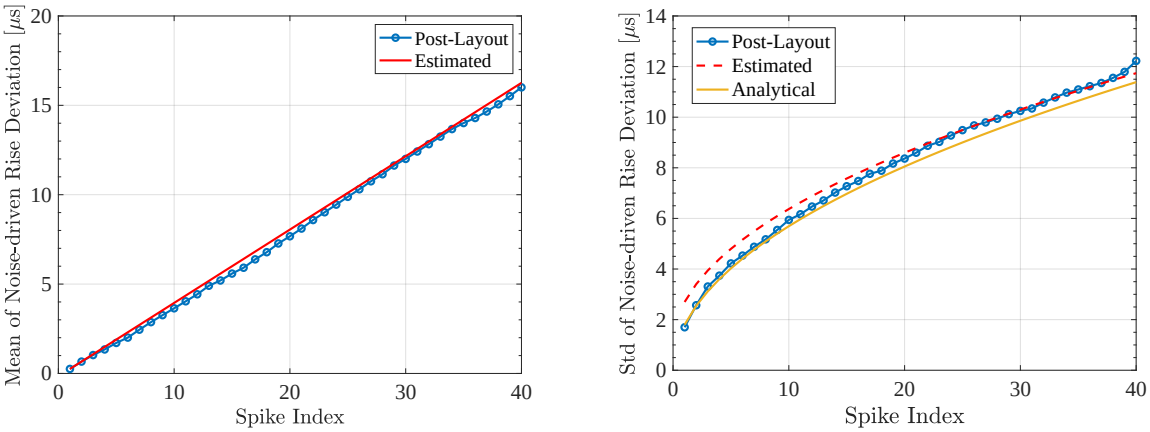


Figure 5.19: (a) Mean noise-driven rise deviation of b-ML eNeuron plotted against spike index, post-layout in blue and estimated in red, (b) Standard deviation of noise-driven rise deviation with post-layout values in blue, estimated in red dashed, and analytical in yellow.

implementing this model in learning A-SNN. Figure 5.18(b) shows the fit of the transfer function of the b-ML eNeuron circuit, defined by its firing rate response to varying synaptic input currents from 0 to 10 nA. The post-layout transfer function is depicted in blue, while the model's fit is shown in green. The fitting function, using interpolation, achieves a near-perfect approximation of the eNeuron's transfer function, with an error rate of $r = 0.99$.

The random noise in the b-ML eNeuron circuit leads to a noise-driven rise deviation that follows a Gaussian random walk distribution across spike events, as depicted in Fig. 4.8(b). The average noise-driven rise deviation, defined in (4.16), demonstrates a linear increase corresponding to spike occurrences. Figure 5.19(a) confirms this trend, displaying the post-layout mean of the noise-driven rise deviation in blue and the estimated mean in red. This average linearly accumulates noise, facilitating adjustments to achieve a zero-mean distribution, as depicted in Fig. 4.8(c).

Regarding the standard deviation of the noise-driven rise deviation, it exhibits a nonlinear increase with each spike. Figure 5.19(b) illustrates this non-linear accumulation with the post-layout standard deviation shown in blue. The estimated equation for this deviation, defined in (4.17), closely matches the post-layout simulations results, as indicated in the red dashed line. The standard deviation of the noise accumulates incrementally, beginning with the initial spike's standard deviation from (4.22). The analytical expression for the standard deviation, shown in yellow, incorporates this initial value and well matches the observed data. These findings affirm the accuracy of the noise model for the eNeuron, and support its integration into A-SNN synthesis with unsupervised STDP learning.

Table 5.3: Integrated Noise Summary in eNeurons

eNeuron	Device	$\overline{V_n^2}$ (V)	% of Total Noise
b-ML	MN_K	0.165	53.42
	MP_{Na}	0.102	19.63
p-ML	MN_K	0.58	60.65
	MP_{Na}	0.34	21.94
AH-LIF	MN_K	0.22	57.7
t-LIF	MN_K	0.38	26.7
	MP_{syn}	0.35	23.32

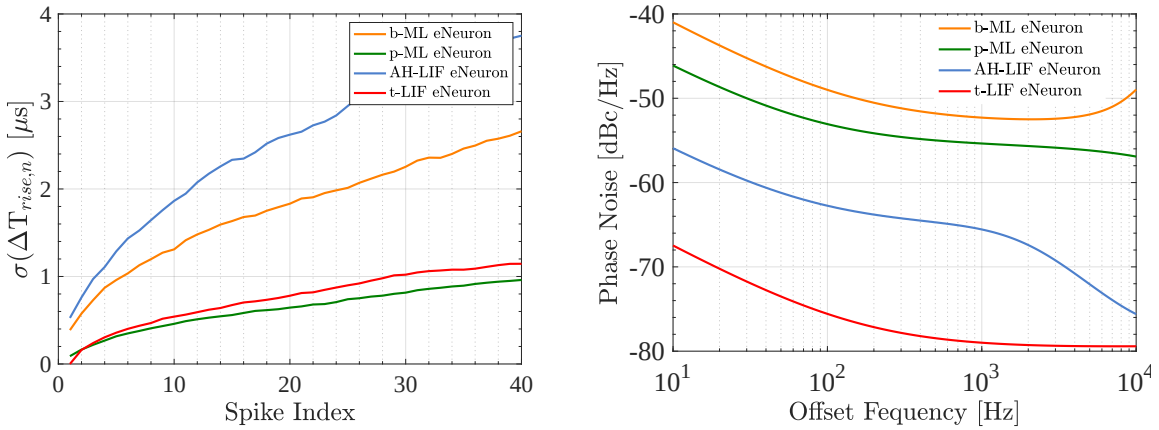
B .Noise-driven Rise Deviation Magnitudes

This section discusses the post-layout noise-driven rise deviation results for four eNeurons: b-ML, p-ML, AH-LIF, and t-LIF, noted for their low power consumption (see Sec. 2.3.2). It is crucial to analyze how the random noise in these circuits affects spike timings, which are essential for synaptic weight adjustments during STDP. The eNeuron layouts with their component sizes are detailed in Appendix D. Consistent technology (BiCMOS 55nm) across all eNeurons ensures a fair comparison. Table 5.3 summarizes the integrated noise across the eNeurons, indicating the main noise contributors are the transistors MN_k, MP_{Na}, and MP_{syn} (modeled in Fig. 4.6). The noise observed in these components is shot noise, validating the hypotheses discussed in Sec. 4.2.2.

Figure 5.20(a) shows the standard deviation of noise-driven rise deviation for the four eNeurons across spike occurrences. These results are derived from 1k-points post-layout trans-noise simulations over a 1 ms window, with each eNeuron stimulated by the same synaptic current (I_{syn} = 50 pA). The graph reveals varying spike timing deviations among the eNeurons, which increase with successive spikes. The b-ML eNeuron (orange line) and AH-MIF eNeuron (blue line) show substantial deviations, reaching up to 3 μs. In contrast, the p-ML eNeuron (green line) and t-LIF eNeuron (red line) maintain deviations below 1 μs up to the 30th spike. The lower deviation in the p-ML eNeuron is attributed to its transistor-only design, minimizing propagation delays from current noise in capacitances. As shown in (4.22), deviation magnitude also depends on the eNeuron's firing frequency. The AH-LIF eNeuron exhibits fewer spikes over time, leading to the highest deviation [155]. Despite variations, all eNeurons exhibit deviations in spike timing that meet or exceed ΔT_s, crucial for synaptic weight updates in STDP.

Figure 5.20(b) shows the phase noise versus offset frequency for a f_{spike} of 40 kHz. Results are from post-layout Virtuoso Spectre PNOISE simulations. Each eNeuron is excited with a different I_{syn} to achieve the same spiking frequency. As the offset frequency increases, phase fluctuations and noise increase. According to (4.23), phase noise $\mathcal{L}(\Delta f)$ is inversely proportional to I_{syn} and directly proportional to C_m. The b-ML eNeuron shows the highest phase noise due to its large C_m. The p-ML eNeuron, despite lacking C_m, exhibits significant phase noise due to low I_{syn}. AH-LIF and t-LIF eNeurons have lower phase noise, as they are excited by higher I_{syn}.

The distribution of noise-driven rise deviation in spike timing for the four eNeurons at the



(a) (b)

Figure 5.20: (a) Standard deviation of noise-driven rise deviation across spike occurrences for b-ML, p-ML, AH-LIF, and t-LIF eNeurons, showing variations that increase with successive spikes, (b) Phase noise versus offset frequency for the same eNeurons.

first and 10th spike occurrences is shown in Fig. 5.21 and Fig. 5.22, respectively. Each eNeuron is excited by different I_{syn} to achieve $f_{spike} = 40$ kHz. These results are based on 1k iterations of post-layout trans-noise simulations. At the 10th spike, the AH-LIF eNeuron has the lowest variation coefficient (76%), while the p-ML eNeuron has the highest (445%), due to its low I_{syn}. According to (4.22), rise deviation increases as I_{syn} decreases. At the 10th spike, rise deviation variations are higher than at the first spike, due to cumulative standard deviation as shown in Fig. 5.19(b). For all eNeurons, noise-driven rise deviation $\Delta T_{rise,n}$ is significant compared to ΔT_s required for STDP synaptic weight updates. Therefore, $\Delta T_{rise,n}$ distributions indicate that spike timing becomes a random variable influenced by noise, leading to random synaptic weight updates in A-SNNs trained with STDP.

The distribution of f_{spike} for the four eNeurons, shown in Fig. 5.23, is based on 1k iterations of post-layout trans-noise simulations. This analysis is crucial to determine whether the effect of the noise on spike timing is averaged out over time in the firing rate. The AH-LIF eNeuron exhibited the most stable f_{spike} with a coefficient of variation of 0.2%, while the b-ML and t-LIF eNeurons showed a slight variation of 0.5%. In contrast, the p-ML eNeuron had a high variation of 6% due to the absence of C_m in its design, making f_{spike} dependent on random parasitic capacitances in the layout, which worsens with process variability.

This study shows that the noise affects the spike timing of low-power eNeurons with high f_{spike}, leading to potential random updates in synaptic weights. Consequently, A-SNNs using STDP learning with these eNeurons may experience random accuracy degradation. However, f_{spike} is less affected by this noise as it averages out over multiple spikes in a time window. The next section highlights the impact of noise on STDP learning for XOR and MNIST problems.

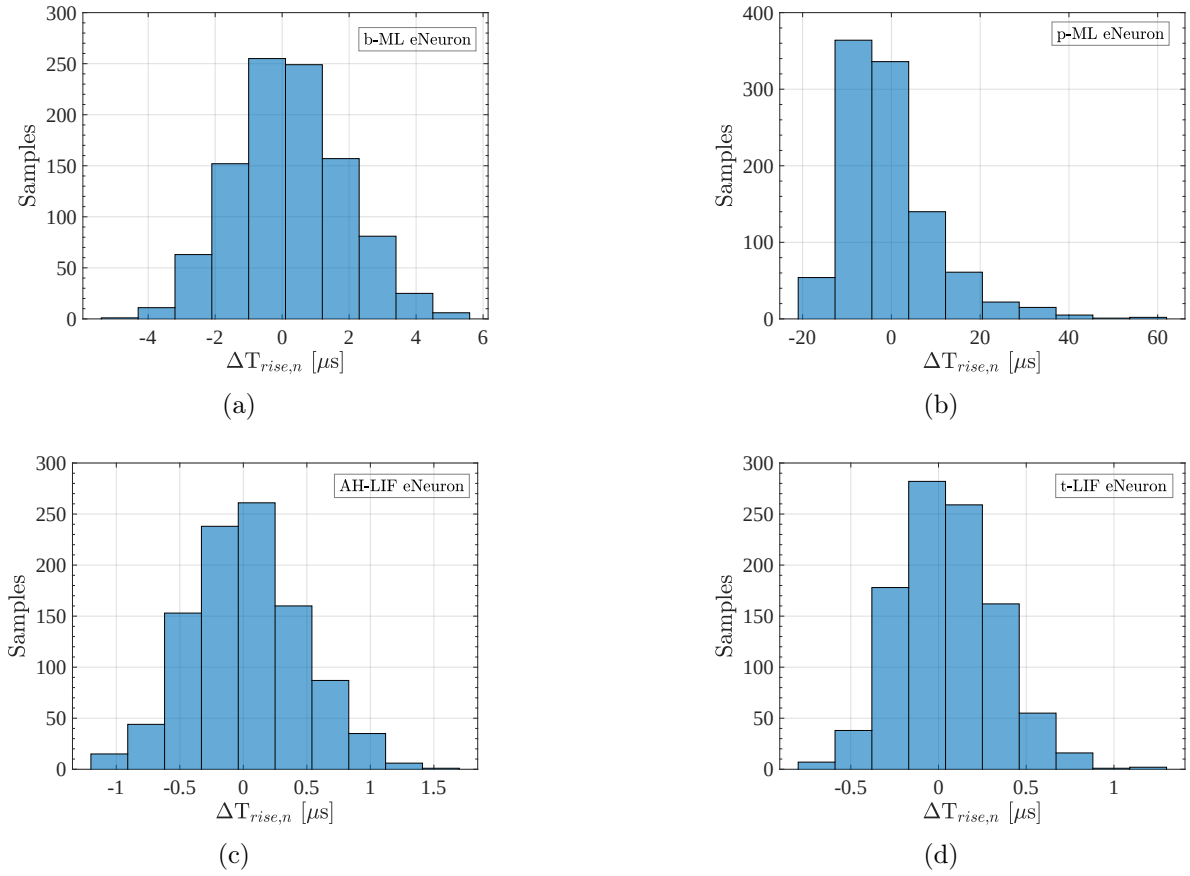


Figure 5.21: Distribution of $\Delta T_{rise,n}$ in first spike timing for 1k iterations of post-layout trans-noise for (a) b-ML eNeuron, (b) p-ML eNeuron, (c) AH-LIF eneuron and (d) t-LIF eNeuron. Each eNeuron is excited by different I_{syn} to achieve f_{spike} = 40 kHz.

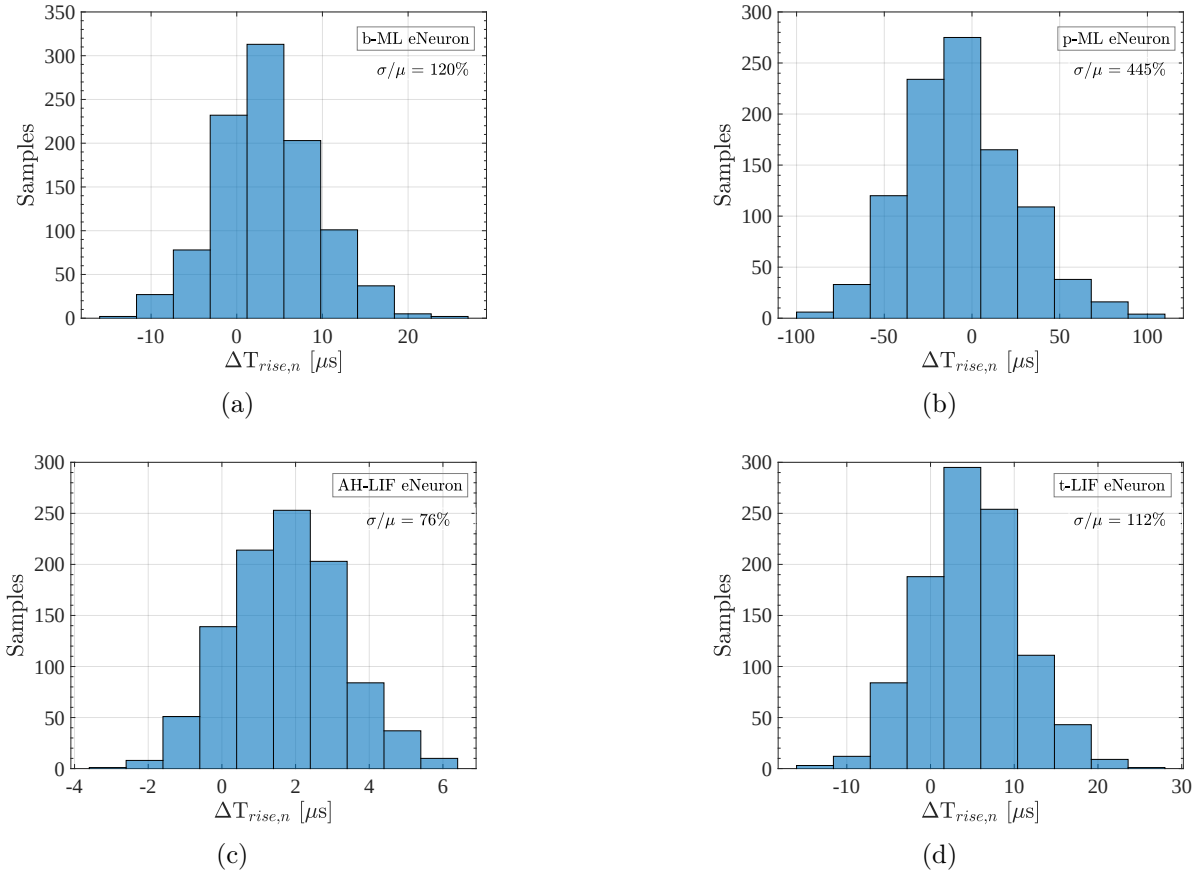


Figure 5.22: Distribution of $\Delta T_{rise,n}$ in 10th spike timing for 1k iterations of post-layout trans-noise for (a) b-ML eNeuron, (b) p-ML eNeuron, (c) AH-LIF eNeuron and (d) t-LIF eNeuron. Each eNeuron is excited by different I_{syn} to achieve $f_{spike} = 40$ kHz.

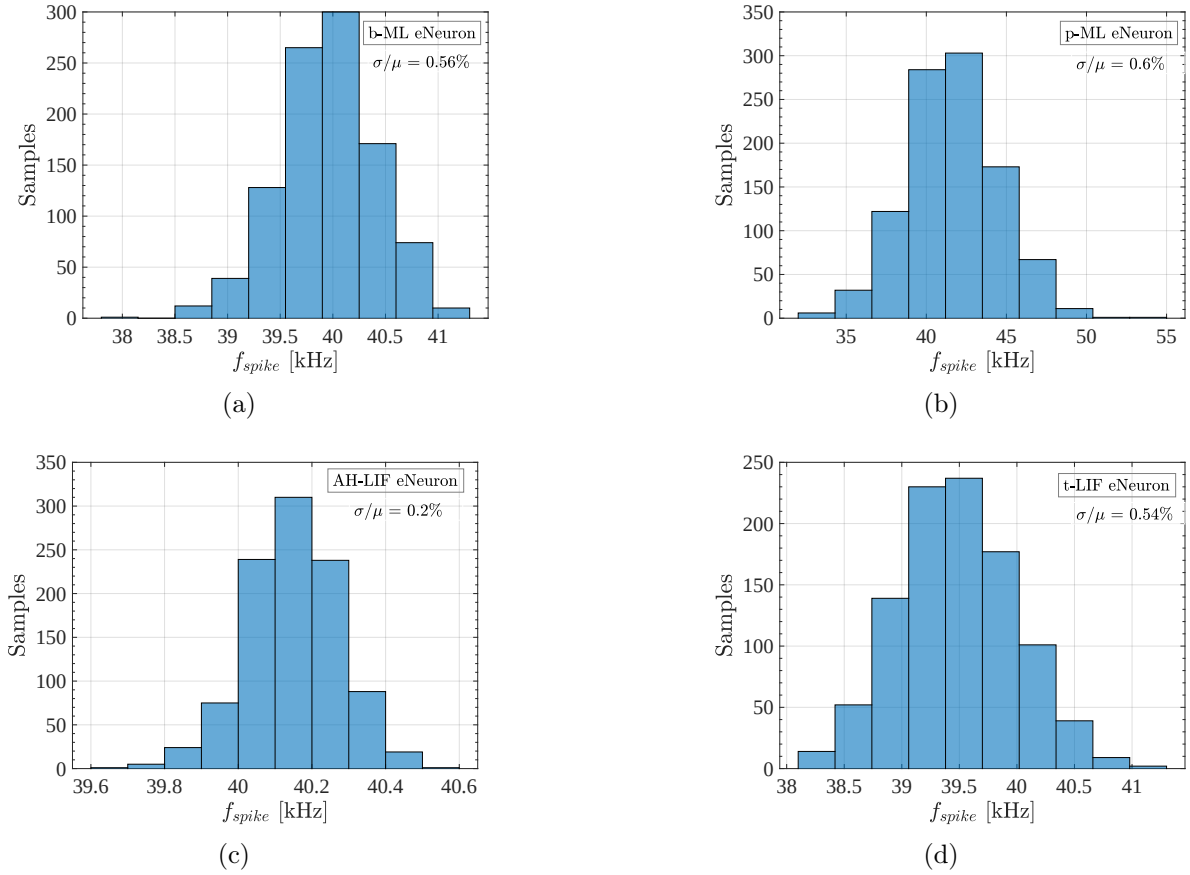


Figure 5.23: Distribution of f_{spike} for 1k iterations of post-layout trans-noise for (a) b-ML eNeuron, (b) p-ML eNeuron, (c) AH-LIF eNeuron and (d) t-LIF eNeuron. Each eNeuron is excited by different I_{syn} to achieve $f_{spike} = 40$ kHz.

C .Performance on XOR and MNIST Problems

The A-SNN is trained and tested according to Alg. 4.2, considering various scenarios of noise incorporation as detailed in Sec. 4.2.3. Two problems were addressed and thoroughly explained in Sec. 4.2.4: the XOR and the MNIST problems.

XOR Problem

Figure 5.24 presents the training and testing performance of the A-SNN across 100 epochs, with accuracy averaged over multiple simulations. The training data is composed of a list of the four binary input pairs ([0,0], [0,1], [1,0], [1,1]), which are presented in a random order, and repeated 10 times during each epoch. The figure contrasts the training accuracy of scenarios involving noiseless eNeurons (blue line, Scenario 1) with those where random noise is present (red and green lines, representing Scenario 2 and Scenario 3, respectively), demonstrating that noise during training leads to poor convergence and low accuracy. The boxplots on the right assess testing accuracy under different testing scenarios: replacing noiseless eNeurons with those having

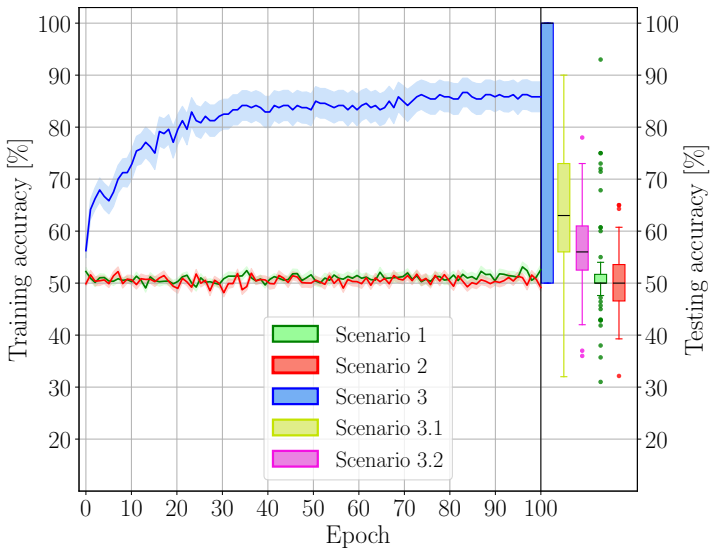


Figure 5.24: Training and testing accuracy of the A-SNN over 100 epochs, for XOR problem and for different scenarios of noise incorporation in the eNeuron model.

a simplified model of random noise (yellow boxplot, Scenario 3.1) results in better accuracy than with a complete model of random noise (pink boxplot, Scenario 3.2). Despite the presence of noise, the network can perform clustering, separating inputs [0,1] and [1,0] from [0,0] and [1,1] during training. By extending the training period beyond the initial XOR problem, the network achieves clustering of all four unique inputs, thereby enabling it to resolve various two-input logic gates simply by reassigning the labels of the output neurons.

Figure 5.25 provides a 3D visualization of the XOR problem as resolved by a trained A-SNN. Each of the plots, labeled (a) through (e), demonstrates how the network interprets varying combinations of binary inputs, which are finely segmented into 10 slices ranging from 0 to 1. The X-input and Y-input axes correspond to the binary inputs [x, y] that feed the two input neurons of the A-SNN. The z-input represents the averaged predictions of the network across numerous simulations. The color gradient on this axis, from blue to yellow, visually encodes the prediction values, where blue indicates a prediction closer to 0 and yellow closer to 1.

Figure 5.25(c) illustrates the network's behavior under Scenario 3, where the network has successfully learned the XOR problem without noise, showing a clear separation between the classes. Figures 5.25(d) and 5.25(e) correspond to Scenarios 3.1 and 3.2, respectively. These plots show the effects of introducing simplified and more complex random noise models during the testing phase, affecting the clarity and definition of the predictive surface. Figures 5.25(a) and 5.25(b) reflect Scenarios 1 and 2, where the network was exposed to noise during the training phase, leading to less effective learning outcomes as evidenced by the less defined predictive surfaces.

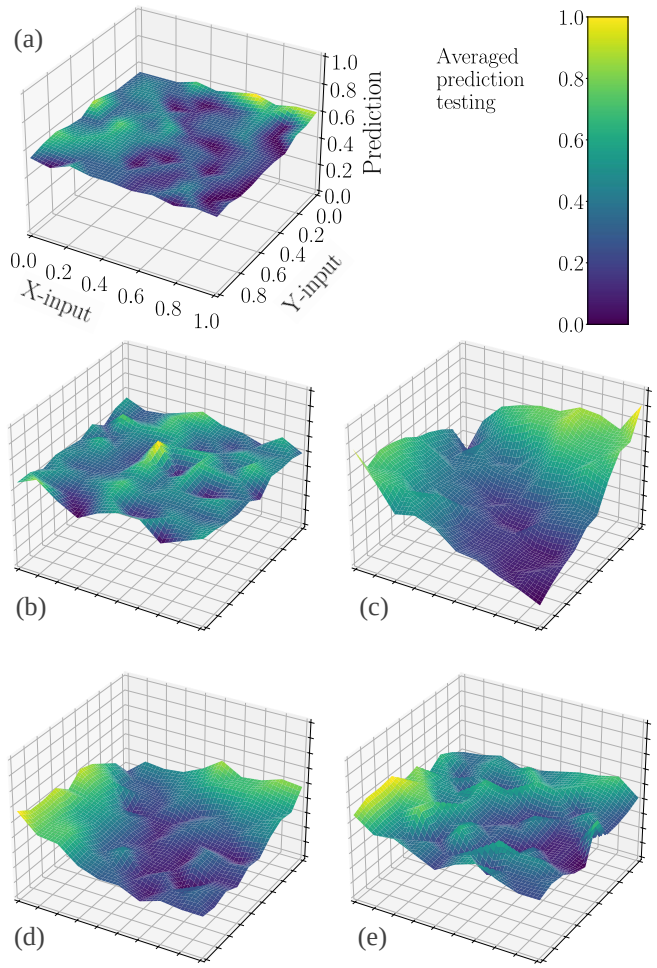


Figure 5.25: 3D visualization of XOR problem resolution by a trained A-SNN, with inputs [x, y] mapped on the X and Y axes and predictions color-coded from blue (0) to yellow (1) on the Z-axis, for (a) Scenario 1, (b) Scenario 2, (c) Scenario 3, (d) Scenario 3.1, and (e) Scenario 3.2.

MNIST Problem

Figure 5.26 illustrates the training and testing accuracy of A-SNN as it processes 10,000 MNIST images across different scenarios. As shown, Scenario 3 demonstrates the most rapid learning, achieving a peak accuracy of 50.6%. However, this scenario's learning plateaus after processing a few thousand samples, indicating a limitation in further improving its accuracy with additional training data. Scenario 1, using a simplified noise model, shows a slightly slower but steady increase in learning, achieving an average accuracy nearly equivalent to Scenario 3 at 49.9%. In contrast, Scenario 2, which applies a complete noise model, demonstrates a poorer performance. This suggests that the full noise model may be disruptive, avoiding the network's effective training. The boxplots on the right provide further insights into testing accuracy variations. Notably, there is a more significant accuracy drop in Scenario 3.2, with an average decrease of 4.5% (indicated by the pink boxplot), compared to a minor 0.2% drop

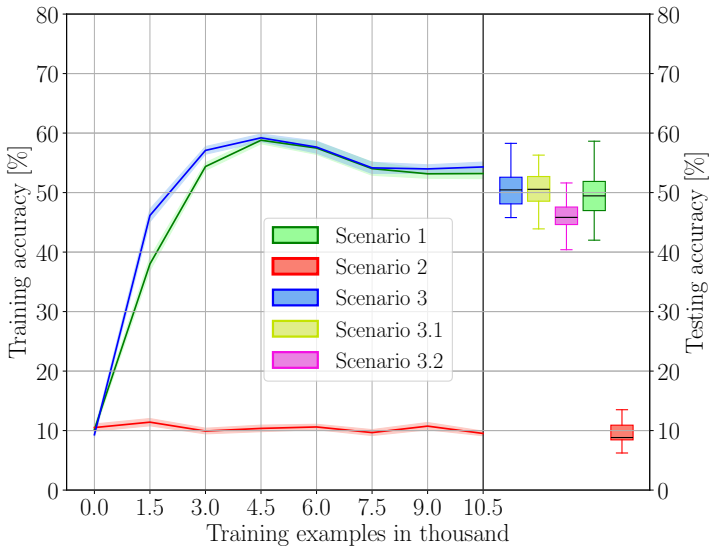
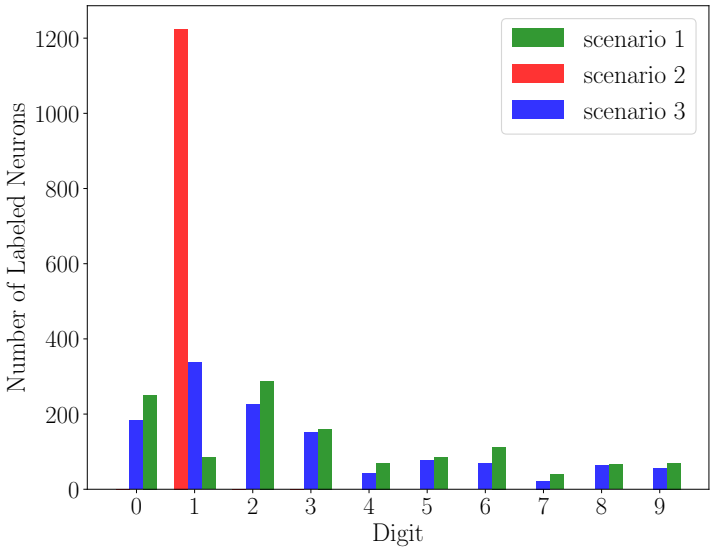


Figure 5.26: Training and testing accuracy of the A-SNN for MNIST problem for different scenarios of noise incorporation in the eNeuron model.

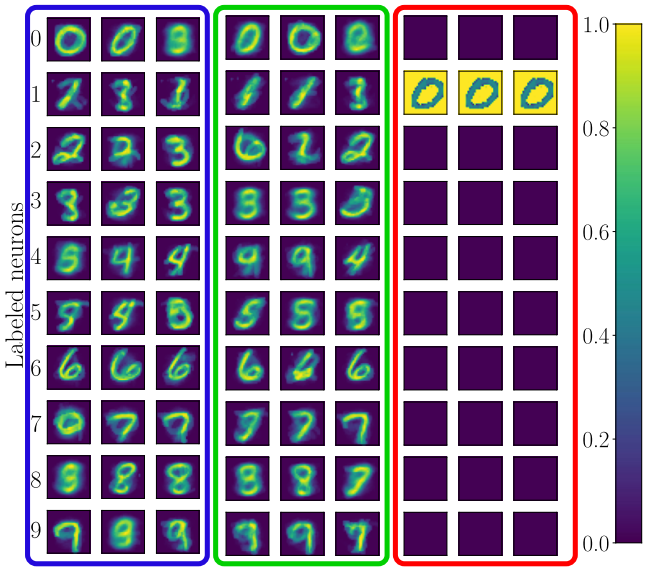
in Scenario 3.1 (shown by the yellow boxplot). This highlights the impact of different noise considerations on the network's robustness in testing phases.

To validate the accuracy results from Fig. 5.26, Fig. 5.27(a) shows the distribution of 1,225 neurons labeled by digit, for each scenario. Scenario 2 (shown in red) has all neurons labeled identically, resulting in repetitive predictions and low accuracy (10%). In contrast, Scenarios 1 (shown green) and 3 (shown blue) present a more diverse labeling, indicating multiple neurons achieving peak firing rates for each digit. Figure 5.27(b) presents the receptive fields of three labeled neurons per digit per scenario, reflecting the patterns learned by the network. The color bar represents the activation level of neurons, ranging from dark blue (0) for minimal activation to yellow (1) for maximum activation. Consistent with the labeling distribution, Scenarios 1 and 3 feature neurons labeled for each digit, although some inaccuracies are noted, such as neurons labeled as '4' showing shapes like '9'. These inconsistencies explain why the SNNs perform better than random guessing (10%) but do not achieve a robust accuracy. Scenario 2 does not respond to any label except for label 1, and its response is incorrect, showing '0'.

The results from the XOR and MNIST problems underscore the significant impact of noise on the performance of the A-SNN with unsupervised STDP learning. The impact is most pronounced in Scenario 2, where the noise model is fully realized through accumulation over spike occurrences. In contrast, Scenario 1, which simplifies noise to an average value at each spike occurrence, shows a lesser impact. When noise is introduced only during the testing phase, as in Scenario 3.1 (simplified noise model) and Scenario 3.2 (complete noise model), the accuracy drop due to random noise remains below 5% in both cases. These results suggest that the drop in accuracy in Scenarios 1 and 2 is likely caused by the training not converging properly. These findings support the implementation of efficient learning methods that can achieve high accuracy, with minimal accuracy degradation due to noise.



(a)



(b)

Figure 5.27: (a) Label distribution of neurons across scenarios, (b) receptive fields of labeled neurons for MNIST problem.

5.3 . Conclusion

This chapter provided a comprehensive overview of the performance and operational results obtained from this thesis. It focused on the efficacy of the end-to-end RF NeuroAS system introduced in Chap. 3, showcasing its capability for energy-efficient source localization. For a 10-degree angular resolution, the system achieved a localization accuracy of 96.7% using the SimLocRF dataset and 93.9% with the MeasLocRF dataset, derived from anechoic chamber measurements. Despite varying signal-to-noise ratios, it maintained high accuracies with a low power consumption of 233 nW. Post-layout simulations of the neuromorphic pre-processing stage of the RF NeuroAS system demonstrated its high effectiveness in converting received power to spike rate and in identifying bit patterns. It achieved a 30 dB dynamic range and displayed robustness across temperature variations. A simplified version of the RF NeuroAS system, implemented in an analog circuit, consistently used just over 1.1 nW, tested across angular resolutions of 5, 12, and 20 degrees.

Subsequently, the chapter examined the integration of deep learning and STDP, both introduced in Chap. 4 into the A-SNN, a fundamental element of the RF NeuroAS system. For deep learning, performance evaluations of the A-SNN using the MNIST problem showed variable accuracies influenced by the eNeuron transfer functions. A maximum accuracy of 90% in training and 83% in testing was achieved when using ML eNeurons. Moreover, high energy efficiency in eNeurons often led to decreased accuracy, reflecting the inherent constraints of deep learning methods. Meanwhile, STDP learning, applied to both the XOR and MNIST problems, demonstrated accuracies that depended critically on the eNeuron circuit noise management. Preliminary results demonstrated that the A-SNN with either simplified or omitted random noise modeling performed better than those with full noise modeling. However, if random noise was neglected during the training phase, the reintroduction of noise modeling in the testing phase caused an average accuracy drop of only 0.2% for the simplified model of random noise and 4.5% for the whole model.

Chapter 6

Conclusion and Perspectives

6.1 . Thesis Conclusions

The rapid growth of the Internet of Things (IoT) necessitates advanced edge AI-based solutions that reduce energy requirements for data transmission, enhance privacy, and make devices more intelligent and autonomous [5, 6]. This evolution requires the development of specialized hardware, efficient algorithms for resource-constrained devices, and innovative neural networks that operate at the edge. Neuromorphic computing, inspired by brain functionality, offers a promising approach to address these needs [11, 12]. Many studies have focused on developing new neuromorphic hardware to overcome the limitations of traditional von Neumann architectures. Other studies have concentrated on bio-plausible algorithms to enhance learning efficiency, while additional efforts aim to evolve neural network models from abstract continuous neuron representations to spiking asynchronous models. These diverse advancements are primarily demonstrated in applications aimed at mimicking human sensory functions, thus diverging from typical IoT requirements. Moreover, few neuromorphic implementations are designed as complete end-to-end systems, often lacking integral pre-processing procedures and requiring digitization. This gap highlights the need for more focus on neuromorphic implementations to leverage its potential for IoT requirements.

This thesis explored the potential of neuromorphic computing to meet the energy efficiency demands of IoT applications, with a particular focus on Radio Frequency (RF) localization. It introduced an end-to-end analog spike-based neuromorphic system (RF NeuroAS) designed for precise and energy-efficient AI-based RF localization. This system was designed to identify source positions within a full 360-degree range on a two-dimensional plane, maintaining a resolution of 1 or 10 degrees, even under noisy conditions. It primarily included a neuromorphic pre-processing stage, followed by an analog-based spiking neural network (A-SNN). The neuromorphic pre-processing stage was implemented in an analog circuit that translated RF signals into spike trains, where the frequency of these spikes correlated with the power of the RF signal. Subsequently, A-SNN was trained and tested using the TensorFlow platform, supported by two datasets for localization developed specifically for this thesis: SimLocRF, derived from MATLAB simulations, and MeasLocRF, obtained from measurements in an anechoic chamber.

This thesis included an in-depth examination of the learning process for the A-SNN, which was a network comprising analog spiking neurons and synaptic models. The primary goal of this study was to develop an adaptable learning strategy for the A-SNN that accounts for the post-layout transfer functions of the eNeuron physical design, the intrinsic random noise from transistors, and the process variability characteristics during weight adjustments. Two learning methods for the A-SNN were explored: software-based deep learning (DL) and biologically plausible spike-timing-dependent plasticity (STDP). For both methods, a theoretical analysis

assessed the feasibility of each approach for the A-SNN. Subsequently, a synthesis was developed and validated using the MNIST and XOR problems.

The results affirmed the efficacy of the RF NeuroAS system in RF localization and validated the analyses of the learning techniques used. For a resolution of 10 degrees and at a 20 dB signal-to-noise ratio (SNR), the RF NeuroAS achieved a localization accuracy of 96.7% with the SimLocRF dataset and approximately 93.9% with the MeasLocRF dataset. The accuracy degraded with decreasing SNR, but the system maintained a low power consumption in the range of nanowatts. A simplified version of the RF NeuroAS, fully implemented in BiCMOS 55 nm technology, demonstrated an ultra-low power consumption of 1.1 nW for a dynamic range of 30 dB. The learning of the A-SNN via DL showed a dependency on the non-linearity of the transfer function and the dynamic range of the A-SNN's neuron model, which could influence the activation function. For the MNIST problem, learning via DL achieved an accuracy of 90% in training and 83% in testing. The learning of the A-SNN via STDP showed a dependency on the random noise of the circuit, potentially affecting spike timing. Simulations for the XOR and MNIST problems showed a strong impact when noise was present during training. However, when noise was introduced only during the testing phase, the accuracy drop was less than 5%.

6.2 . Perspectives

6.2.1 . Full Implementation of RF NeuroAS System for Localization

The RF NeuroAS system, mainly composed of the neuromorphic pre-processing and the A-SNN, made its proof on RF source localization at the system level [16]. It took into consideration the post-layout simulation results from the neuromorphic preprocessing, which had already been fully implemented and validated at the layout level. However, it only considered the post-layout simulation results for the individual neuron and synapse circuits of the A-SNN, without addressing the constraints that arose when the A-SNN was fully implemented with all neuronal and synaptic connections at the layout level. Only a simplified version of the RF NeuroAS, featuring a smaller A-SNN, was implemented in an analog circuit and validated through post-layout simulation results.

The next step will be to fully implement the RF NeuroAS system, starting from the schematic design and progressing to the layout and eventual fabrication. This effort will build on the work submitted in [25], where a complete analog neural network is implemented at the layout level. This study highlighted the tools needed for connecting synapses and neurons at the physical level and provided the detailed synthesis of training and testing for such implementations. It presented post-layout simulation results of the network, which was trained using TensorFlow and deep learning techniques and tested at the layout level. The approach was validated on a simple problem of determining whether a point lies inside a circle. Future work might extend this approach, featured in [25], by developing a more complex analog neural network for tackling more advanced RF localization problems as mentioned in [16]. This perspective would allow for an advanced evaluation of the RF NeuroAS system's capabilities and its application in energy-efficient RF source localization.

6.2.2 . Considering an Advanced RF Environment for Localization

In this thesis, the RF NeuroAS system identified source positions within a specific 2D plane configuration, where the source occupied a position within a circle defined by angle θ_s and radius d_s from the origin. The RF environment was considered a free-space scenario, specifically in an anechoic chamber, meaning that interferences and reflections were not taken into consideration. Moreover, the RF NeuroAS system relied on detecting power levels (i.e. RSS detection) as the key information from received signals.

Future works will push the boundaries to advanced scenarios, such as considering configurations where sources are located in a 3D plane or identifying trajectories other than the circular path considered in this thesis, or including multi-path scenarios. These advanced considerations in the RF environment will require several adaptations. Firstly, the datasets generated (SimLocRF in MATLAB and MeasLocRF from anechoic chamber measurements) will need to be adapted to handle these advanced scenarios. This will involve using new data from real-world scenarios beyond anechoic-chamber measurements. The process will start by placing objects in the anechoic chamber to consider a few reflections and will progress to obtaining data directly from real-world environments outside the chamber. Secondly, the RF NeuroAS system will need to be extended to a more complex version to handle these advanced scenarios. This might involve adding more receivers to capture additional data, developing a higher-performance pre-processing stage to achieve a higher dynamic range, and creating a more complex neural network with additional hidden layers and optimized hyperparameters. Finally, an advanced type of detection will be needed rather than the simple RSS detection, which leads to reduced accuracy in multi-path scenarios, where power levels are significantly affected by interference. Therefore, other measurement techniques, such as time of arrival, angle of arrival, or the methods described in Sec. 2.1.2, shall be considered.

6.2.3 . Intrinsic Random Noise Impact on STDP Learning

The feasibility of unsupervised STDP for the A-SNN was thoroughly examined in this thesis (see details in Sec. 4.2 and related papers [21], [22]), aiming to establish a biologically inspired learning method suitable for A-SNN. It examined how random noise from transistors in analog circuits influences spike timings, which are crucial for STDP-based weight adjustments. The results for both XOR and MNIST problems indicated a significant impact when noise was present during network training, raising the question of whether this noise can be adjusted or even leveraged beneficially for learning. It is recognized that biological neural networks employ noise to improve learning efficiency, suggesting a potentially beneficial role for noise in computational models such as SNNs. Consequently, future work could develop a new methodology inspired by biological techniques to improve the learning capabilities of STDP, particularly in addressing noise within A-SNNs [231]. Additionally, this methodology could draw from advanced techniques developed in current research to either mitigate noise [232] or transform it into a resource for computation and learning [233], [234].

Additionally, this work on STDP learning with A-SNN primarily focused on the post-layout results of the neuron circuit, while using a mathematical model of the synapse. An analog circuit implementation of a synapse adaptable for STDP learning has already been demonstrated

in [152]. The next step could involve incorporating both components—the neuron and the synapse—into the learning framework of the A-SNN using STDP, fully accounting for electrical constraints, particularly noise.

Résumé Étendue en Français

1 . Introduction

Imaginez un monde où les machines communiquent sans heurt, où les usines fonctionnent de manière autonome et où les systèmes intelligents anticipent nos besoins avant même que nous les exprimions. Imaginez des véhicules qui naviguent de manière autonome dans les rues animées des villes, et des bras robotiques qui effectuent des chirurgies complexes avec une précision inégalée. Imaginez une montre qui fait bien plus que compter les minutes — elle compte vos pas, surveille votre santé, gère vos paiements et vous tient au courant avec des notifications. Imaginez un assistant virtuel si intuitif qu'il semble humain, capable de gérer vos tâches quotidiennes et de participer à toute conversation avec une compréhension approfondie. Cette vision, qui semblait relever de la science-fiction, est en train de devenir une réalité grâce aux avancées monumentales du 21ème siècle dans la quatrième révolution industrielle, l'Industrie 4.0, alimentée par l'Internet des objets (IoT) et l'intelligence artificielle (IA).

L'IoT représente un changement révolutionnaire, redéfinissant les modes de vie traditionnels grâce à la technologie avancée. À ce jour, des milliards de ces dispositifs peuplent le globe, et selon les prévisions de IoT Analytics, le nombre de connexions IoT dépassera les 29 milliards d'ici 2027. Cette croissance considérable soutient un large éventail d'applications, impactant divers secteurs tels que la santé, les transports, les dispositifs portables, l'automatisation domestique, les opérations industrielles et l'agriculture. Capitalisant sur les avancées de l'intelligence artificielle (IA) et du cloud computing, l'IA basée sur le cloud centralise le traitement des données dans l'écosystème IoT, élargissant ses applications grâce à l'augmentation des objets connectés. Toutefois, cette croissance rapide exerce une pression sur les capacités du cloud, soulignant ses limites en termes de latence et de bande passante. Pour y remédier, l'IA de périphérie est utilisée pour traiter les données plus près de leur source, réduisant la latence, les besoins énergétiques, et améliorant la sécurité et la confidentialité.

Dans le cadre de l'Industrie 4.0, la tendance actuelle en IA se concentre sur les réseaux de neurones profonds (DNN) pour résoudre des problèmes complexes avec de hautes performances, élargissant ainsi les applications de l'IA dans divers domaines. Ces modèles d'IA se composent de réseaux à couches multiples et densément connectées, utilisant principalement l'algorithme de rétropropagation qui requiert un ajustement précis des paramètres. Leur formation nécessite de grands ensembles de données et d'importantes ressources informatiques. À mesure que les capacités de l'IA s'étendent, les défis associés augmentent également, notamment en termes de demandes computationnelles qui surpassent la loi de Moore, entraînant une consommation énergétique élevée et des besoins accrus en stockage. Pour relever ces défis, le domaine de l'IA se tourne vers des solutions plus écoénergétiques, telles que l'IA en périphérie, qui optimise l'efficacité énergétique en traitant les données directement à la source, réduisant ainsi les transferts fréquents de données. Cependant, cette approche nécessite un matériel spécialisé et des conceptions de réseaux neuronaux innovantes adaptées aux ressources limitées des appareils de périphérie. Une solution prometteuse réside dans le calcul neuromorphique, inspiré par le

cerveau, pour développer des solutions d'IA plus efficaces et économes en énergie.

Le calcul neuromorphique est apparu comme une approche prometteuse pour gérer les demandes computationnelles de l'IA et de l'apprentissage profond, tout en répondant aux besoins d'efficacité énergétique des applications IoT. En imitant les capacités de traitement efficaces, adaptatives et rapides du cerveau humain, le calcul neuromorphique est priorisé pour plusieurs raisons. Les systèmes neuromorphiques atteignent une haute efficacité énergétique et une latence réduite en effectuant des calculs en mémoire, se détachant des architectures von Neumann conventionnelles. Ces systèmes excellent dans le traitement parallèle et les opérations en temps réel en simulant les réseaux neuronaux biologiques par le biais de calculs pilotés par événements. Spécifiquement, ils s'appuient sur les réseaux de neurones à impulsions (SNNs), qui activent les neurones uniquement lorsque cela est nécessaire, optimisant ainsi l'efficacité et s'alignant sur les dynamiques temporelles des neurones à impulsions.

Au cours des dernières années, les implémentations matérielles du calcul neuromorphique ont considérablement progressé, avec des développements couvrant les plateformes numériques, analogiques et mixtes utilisant à la fois les technologies CMOS traditionnelles et émergentes. Les implémentations les plus courantes se sont concentrées sur la réplication des fonctions sensorielles humaines, telles que la reconnaissance visuelle et auditive, divergeant ainsi des besoins spécifiques des applications IoT. Les dispositifs IoT fonctionnent différemment dans leurs fonctions de capteurs, incluant la localisation des dispositifs, la détection de la pression et de la température. Cet écart souligne le besoin critique de systèmes neuromorphiques spécialement conçus pour répondre aux exigences uniques des applications IoT.

2 . Contributions de la thèse

Cette thèse est multidisciplinaire, reliant la microélectronique, l'électromagnétisme et l'intelligence artificielle. Elle vise à répondre au besoin d'applications IoT intelligentes à faible puissance via une approche neuromorphique analogique basée sur des impulsions. L'application principale ciblée par cette thèse est la localisation en fréquence radio (RF) d'un émetteur IoT avec une précision comparable aux solutions basées sur l'IA, tout en étant nettement plus écoénergétique et respectueuse de l'environnement. C'est une application critique, alors que la demande pour des solutions écoénergétiques et précises dans l'IoT intelligent s'intensifie. Les principales contributions de cette thèse peuvent se résumer comme suit :

- **Conception d'un système neuromorphique RF complet (RF NeuroAS)**: Ce système utilise une approche basée sur des impulsions analogiques pour émuler les capacités cérébrales dans l'adressage des défis de localisation RF. Il identifie les positions des émetteurs IoT dans une plage de 360 degrés sur un plan bidimensionnel, maintenant une haute résolution (10 ou 1 degré) même dans des conditions bruyantes. Ce travail comprend la configuration de l'environnement RF, l'extraction de données impliquant la génération de jeux de données, le prétraitement et la conception de réseaux neuronaux. Les communications scientifiques mettant en avant cette contribution sont disponibles dans [16] et [17].

- **Implémentation d'un design de circuit entièrement analogique pour une version simplifiée du système RF NeuroAS**, utilisant la technologie BiCMOS 55 nm. Cela valide la faisabilité des solutions RF NeuroAS basées sur le matériel pour la localisation, prouvant une consommation d'énergie ultra-faible dans les simulations post-layout. Les communications scientifiques mettant en avant cette contribution sont disponibles dans [18], [19], et [20].

- **Étude de faisabilité des techniques d'apprentissage sur des réseaux neuronaux à impulsions basés sur l'analogique**, en considérant les modèles et les contraintes de conception de circuits des composants neurones et synapses. Les techniques d'apprentissage explorées incluent l'apprentissage profond et la plasticité dépendante du temps de l'impulsion. Les communications scientifiques mettant en avant cette contribution sont disponibles dans [21], [22], [23],[24], [25], et [26].

Le Chapitre 2 offre un aperçu du développement et de l'évolution de l'IA, de l'apprentissage profond et du calcul neuromorphique dans la littérature, et décrit les efforts de recherche basés sur l'IA dans le domaine de la localisation. Le Chapitre 3 présente en détail le système RF NeuroAS et chacun de ses composants, tandis que le Chapitre 4 détaille l'analyse de faisabilité des techniques d'apprentissage d'un point de vue théorique. Le Chapitre 5 présente les résultats détaillés obtenus dans cette thèse, correspondant aux performances du RF NeuroAS et aux résultats de faisabilité.

3 . Vers une localisation RF efficace : Une approche neuromorphique analogique basée sur des impulsions

Ce chapitre présente un système neuromorphique analogique complet (RF NeuroAS) conçu pour une localisation RF précise et écoénergétique basée sur l'IA. Ce système est conçu pour identifier les positions sources dans une plage complète de 360 degrés sur un plan bidimensionnel, avec une résolution de 10 degrés, même dans des conditions bruyantes. Une version raffinée du RF NeuroAS est introduite, conçue pour une précision accrue dans la localisation des sources tout en maintenant une faible consommation d'énergie, atteignant une résolution angulaire de 1 degré. Des détails supplémentaires sont disponibles dans l'Annexe A.

Figure 6.1 illustre l'architecture du RF NeuroAS, divisée en quatre composants clés : la configuration de l'environnement RF, l'extraction des données, le prétraitement neuromorphique et le réseau neuronal analogique à impulsions. La première étape organise l'arrangement des antennes sources et réceptrices dans le champ RF, permettant une identification précise de la source par l'angle de coordonnées θ_s et la distance d_s. La deuxième étape se penche sur l'extraction des données de l'environnement RF, menant à la création de deux bases de données analytiques et expérimentaux pour la localisation : SimLocRF, dérivé des simulations MATLAB, et MeasLocRF, obtenu à partir de mesures dans une chambre anéchoïque. La troisième étape est le prétraitement neuromorphique, mis en œuvre dans un circuit analogique qui traduit les signaux RF en trains d'impulsions, où la fréquence de ces impulsions est corrélée à la puissance du signal RF. La dernière étape est le réseau neuronal à impulsions basé sur l'analogique (A-SNN), incorporant les

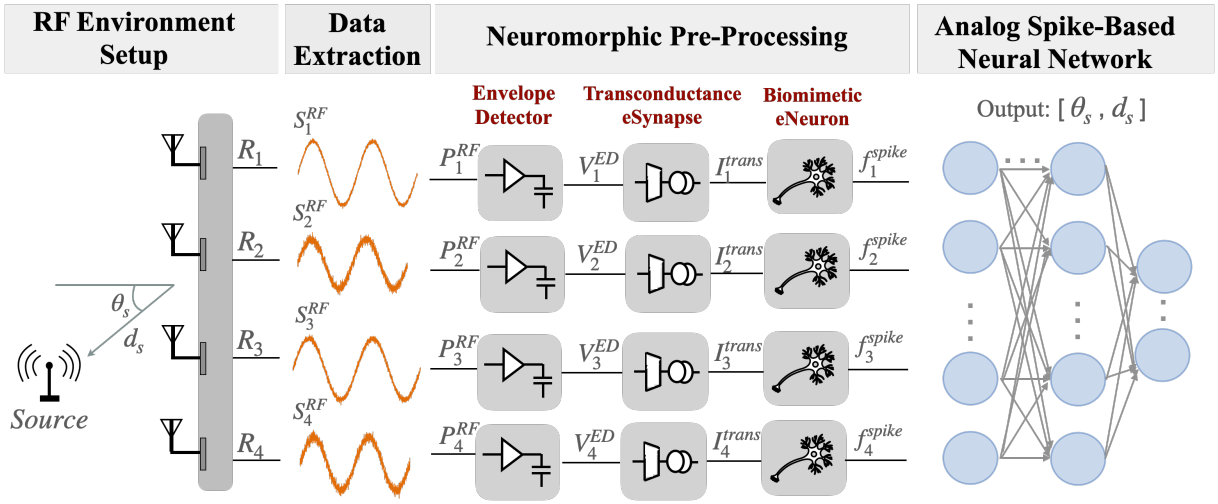


Figure 6.1: Architecture au niveau système de RF NeuroAS pour la localisation de source, comprenant quatre étapes : configuration de l'environnement RF, extraction des données, prétraitement neuromorphique, et réseau neuronal analogique à impulsions. Les sorties incluent la position de la source, donnée sous forme d'angle θ_s et de distance d_s.

fonctions des neurones à impulsions analogiques (eNeurons) et des synapses (eSynapses). A-SNN est formé et testé en utilisant la plateforme TensorFlow, avec les deux bases de données.

4 . Étude de faisabilité des réseaux neuronaux à impulsions analogiques : Des eNeurons à impulsion aux techniques d'apprentissage

Ce chapitre comprend une étude approfondie du processus d'apprentissage pour le A-SNN, qui est un réseau composé de neurones à impulsions analogiques (eNeurons) et de synapses. L'objectif principal de cette étude était de développer une stratégie d'apprentissage adaptable pour l'A-SNN, qui tienne compte des fonctions de transfert post-layout de la conception physique de l'eNeuron, du bruit aléatoire intrinsèque des transistors et des caractéristiques de variabilité du processus lors des ajustements de poids. Deux méthodes d'apprentissage pour le A-SNN ont été explorées : l'apprentissage profond (DL) et la plasticité dépendante du temps de l'impulsion (STDP) biologiquement plausible.

4.1 . Approche d'apprentissage profond pour A-SNN

Avant de développer un cadre de synthèse pour le A-SNN, une étude de faisabilité a été menée pour examiner si l'apprentissage profond pouvait être appliqué ou non sur A-SNN. L'applicabilité de l'apprentissage profond suggère une comparaison entre les propriétés du DNN et du A-SNN, principalement sur la structure et les caractéristiques de la fonction d'activation. En considérant un DNN, si le réseau est entièrement composé d'activations linéaires, quel que soit sa profondeur (c'est-à-dire le nombre de couches), il peut finalement être simplifié à une seule fonction linéaire de l'entrée à la sortie. Par conséquent, le réseau pourrait être équivalamment représenté par

juste une couche d'entrée et une couche de sortie, annulant effectivement les avantages potentiels de la technique d'apprentissage profond. En se basant sur cette propriété, l'étude de faisabilité se penche sur l'expression analytique de la sortie du A-SNN, et prouve que : chaque fois que l'eNeuron et l'eSynapse au sein du A-SNN ont des fonctions de transfert linéaires, cela conduit à un réseau composé uniquement de fonctions d'activation linéaires et donc l'apprentissage profond n'est pas faisable.

Par conséquent, pour l'applicabilité de l'apprentissage profond sur A-SNN, la plage appropriée où l'eNeuron et l'eSynapse ont de la non-linéarité dans leurs fonctions de transfert a été examinée. Le cadre de synthèse a été développé dans Algo. 4.1 en impliquant l'entraînement et le test du A-SNN dans TensorFlow, en tenant compte des contraintes et caractéristiques du circuit. La synthèse du A-SNN à travers l'apprentissage profond comprend plusieurs étapes clés, comme suit.

1- Considérer le modèle eNeuron et eSynapse. Cela est réalisé en chargeant les fonctions de transfert des circuits eNeurons et eSynapses à partir des simulations post-layout et identifier en fonction d'elles la fonction d'activation du A-SNN, utilisée pendant le processus d'apprentissage profond.

2- Considérer la variabilité du processus lors des ajustements de poids. Cela est réalisé en considérant un poids synaptique variable pour la phase de test du A-SNN, où la moyenne est le poids synaptique obtenu après la phase d'entraînement. La représentation statistique des poids synaptiques vise à tenir compte des variances que le matériel pourrait introduire en raison de la variabilité du processus (mismatchs), assurant une évaluation précise du A-SNN.

Ce cadre est illustré à travers la résolution du problème MNIST, comme démonstration des performances du A-SNN lors de l'application de la technique d'apprentissage profond. Il comprend l'applicabilité de DL suivant trois différents A-SNN : un avec l'eNeuron biomimétique (b-ML eNeuron), un avec une version simplifiée biomimétique (s-ML eNeuron), et un avec l'eNeuron le moins biomimétique (AH-LIF eNeuron).

4.2 . Approche de l'apprentissage temporel (STDP) pour A-SNN

Pour explorer le potentiel de la STDP avec A-SNN, un modèle complet d'eNeuron doit être considéré pour la synthèse. Trois propriétés principales doivent être extraites des simulations post-layout et intégrées dans le modèle : son comportement d'impulsion dans le temps, sa fréquence d'impulsions en fonction de l'excitation d'entrée, et la variabilité du bruit inhérent des transistors au sein du circuit analogique. Cette dernière est particulièrement importante pour les applications d'apprentissage STDP, où le bruit peut causer des variations dans le timing des impulsions, un facteur crucial pour ajuster les poids synaptiques par STDP. Ainsi, une analyse approfondie du bruit de l'eNeuron est menée pour modéliser ce bruit et estimer son ampleur dans le circuit ainsi que son impact sur le timing des impulsions. Il est identifié par une distribution de *Gaussian random walk* sur les occurrences des impulsions dans le temps, avec un écart-type dépendant des paramètres du circuit, du courant synaptique d'entrée, et du taux d'impulsions.

Suite à cela, un cadre de synthèse est développé pour A-SNN basé sur cette technique d'apprentissage temporel, présenté dans Algo. 4.2. Il organise les phases d'entraînement et de test du A-SNN au sein du simulateur Brian 2, par une technique STDP non supervisée. Ce cadre

est démontré à travers la résolution des problèmes XOR et MNIST, comme une démonstration des performances du A-SNN lors de l'application de l'apprentissage STDP bio-plausible.

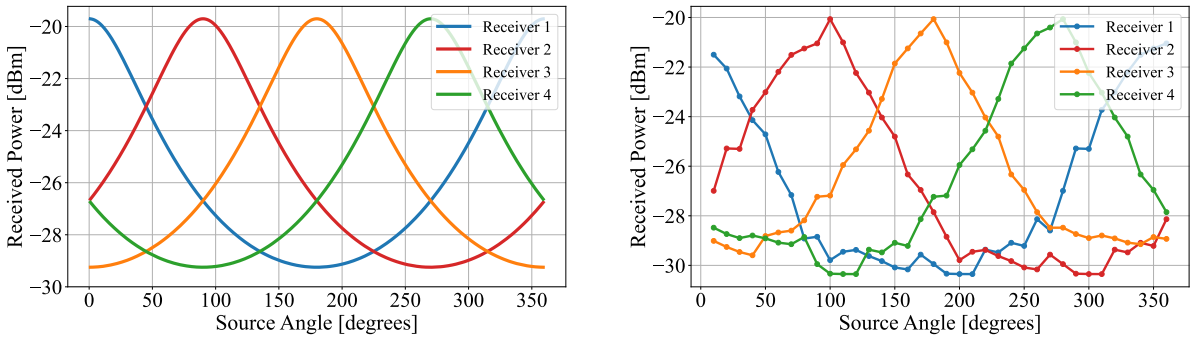
5 . Résultats

Ce chapitre vise à présenter les résultats de performance dérivées de cette thèse, soulignant l'efficacité du RF NeuroAS pour réaliser une localisation de source écoénergétique et les résultats pratiques de l'utilisation des techniques d'apprentissage au sein du A-SNN.

5.1 . Performance de RF NeuroAS

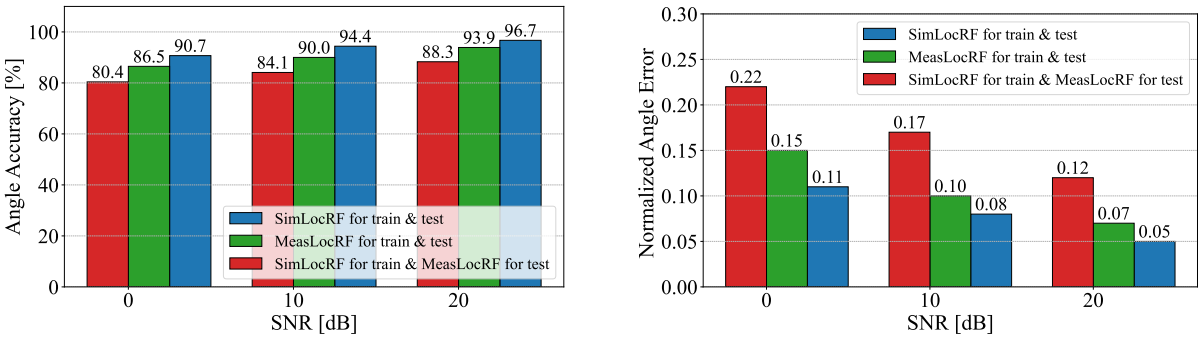
Cette section présente les résultats du système RF NeuroAS, en commençant par la validation des caractéristiques des bases de données utilisés pour son apprentissage, et en présentant ensuite sa performance dans la localisation de sources RF. Figure 6.2 illustre les motifs de puissance reçus par les quatre récepteurs sur une gamme complète de 360 degrés, obtenus à partir de la base de données SimLocRF (Fig. 6.2(a)) et la base de données MeasLocRF (Fig. 6.2(b)). Dans les deux cas, la source est positionnée à 0,5 mètre de l'origine et émet un signal de 10 dBm dans un environnement SNR de 20 dB. Les données simulées dans la Fig. 6.2(a) présentent une courbe cohérente de puissance reçue de -20 à -29 dBm tout au long de la rotation de la source, dans la plage dynamique correspondante de l'étape de prétraitement neuromorphique. Inversement, les données expérimentales dans la Fig. 6.2(b) montrent des fluctuations de -20 à -31 dBm en raison des irrégularités de la chambre anéchoïque, telles que les propriétés des matériaux, l'équipement pratique et les variations des motifs d'antenne.

Figure 6.3 montre la précision angulaire (Fig. 6.3(a)) et l'erreur angulaire (Fig. 6.3(b)) à travers trois scénarios d'entraînement et de test et à trois niveaux de SNR différents. Lors de l'utilisation des bases de données SimLocRF ou MeasLocRF, 70% des données sont allouées pour la phase d'entraînement et de validation, tandis que les 30% restants sont utilisés pour



(a) (b)

Figure 6.2: Illustration de la puissance reçue (en dBm) aux quatre récepteurs en fonction de l'angle de la source, variant de 0 à 360 degrés par incréments de 10 degrés. Le panneau (a) montre les données de SimLocRF, tandis que le panneau (b) présente les données de MeasLocRF.



(a) (b)

Figure 6.3: Performance du A-SNN à une résolution de 10 degrés en termes de (a) précision angulaire (en%) et (b) erreur angulaire normalisée (en degrés) de la position de la source, pour trois scénarios différents de formation et de test, et pour trois niveaux de SNR.

les tests. La précision angulaire la plus élevée, 96.7%, est obtenue lorsque le réseau est formé et testé avec la base de données SimLocRF dans l'environnement le moins bruyant (SNR = 20 dB). L'exactitude tend à diminuer avec la baisse du SNR et lors du passage à la base de données MeasLocRF, car les fluctuations augmentent. La précision la plus basse, 80.4%, se produit dans l'environnement le plus bruyant à 0 dB SNR, en particulier lorsque le réseau est formé avec des données SimLocRF et testé sur de nouvelles données MeasLocRF non vues.

L'erreur angulaire est calculée par $1/(N_f \times RE) \cdot \sum_{i=1}^{N_f} |\theta s_i - \hat{\theta} s_i|$, où θs_i et $\hat{\theta}_{s_i}$ sont les angles de source réels et prédits, respectivement, à travers N_f échantillons de test, et RE est la résolution fixée à 10 degrés. À un SNR de 20 dB, les valeurs de NAE les plus basses enregistrées — 0.02, 0.07, et 0.12 pour les premier, deuxième et troisième scénarios respectivement — démontrent une haute précision dans l'estimation angulaire. De plus, même au niveau de SNR difficile de 0 dB, les valeurs de NAE les plus élevées de 0.11, 0.15, et 0.22 pour les scénarios respectifs restent dans une fourchette acceptable.

Selon la méthodologie d'estimation de la consommation d'énergie, le système RF NeuroAS, composé de 60 eNeurons ML et de 492 eSynapses, consomme seulement 255 nW de puissance. Ces résultats démontrent la robustesse du réseau neuronal en termes de performance et sa capacité à maintenir une architecture simple et à faible puissance, bien adaptée à la localisation RF efficace. De plus, une version simplifiée du RF NeuroAS, avec un réseau plus petit entièrement implémenté en technologie BiCMOS 55 nm, a démontré une consommation d'énergie ultra-faible de 1.1 nW pour une gamme dynamique de 30 dB.

5.2 . Avancées dans les techniques d'apprentissage pour les réseaux neuronaux à impulsions analogiques

L'apprentissage du A-SNN via DL montre une dépendance à la non-linéarité de la fonction de transfert et à la gamme dynamique du modèle de neurone du A-SNN, qui pourrait influencer la fonction d'activation. L'apprentissage via DL est validé à travers le problème MNIST, en utilisant A-SNN avec différents types d'eNeurons. Figure 6.4 illustre la précision du A-SNN

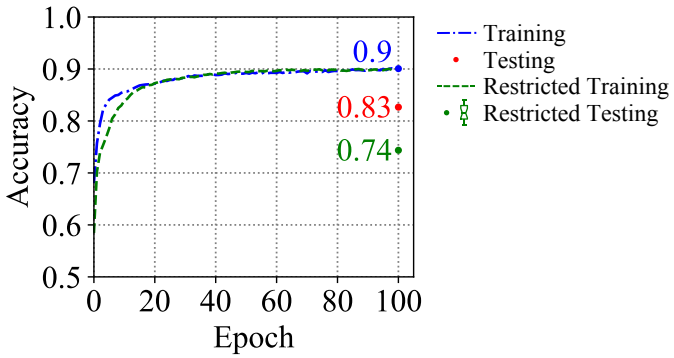


Figure 6.4: Résultats de précision pour le A-SNN dans la résolution du problème MNIST, en utilisant le eNeuron s-ML. Les scénarios restreints et non restreints sont considérés pour l'entraînement et le test.

pendant les phases d'entraînement et de test pour le problème MNIST, utilisant l'eNeuron s-ML, et pour deux scénarios : un non restreint où toute la gamme de la fonction de transfert de l'eNeuron est prise en compte, et l'autre restreint où seule une partie de la gamme dynamique est prise en compte pour une efficacité énergétique élevée. Les scénarios non restreints et restreints montrent des améliorations substantielles de la précision de l'entraînement jusqu'à l'époque 40, atteignant 0.9 à l'époque 100. Pour le scénario non restreint, l'apprentissage via DL a atteint une précision de 90% en entraînement et 83% en test. La précision des tests chute de 16% dans le cas restreint à 0.74.

L'apprentissage via STDP a montré une dépendance au bruit aléatoire du circuit, affectant potentiellement le timing des impulsions. Figure 6.5 illustre la précision de l'A-SNN pendant et après l'entraînement, lorsqu'il traite 10 000 images MNIST dans différents scénarios. Comme montré, le scénario 3 (modèle sans bruit de l'eNeuron) démontre l'apprentissage le plus rapide, atteignant une précision maximale de 50.6%. Le scénario 1, utilisant un modèle de bruit simplifié, montre une augmentation de l'apprentissage légèrement plus lente mais régulière, atteignant une précision moyenne presque équivalente à celle du scénario 3, soit 49.9%. En revanche, le scénario 2, qui applique un modèle de bruit complet, montre une performance inférieure. Cela suggère que le modèle de bruit complet peut être perturbateur, empêchant un entraînement efficace du réseau. Les boxplots à droite fournissent des informations supplémentaires sur les variations de précision après l'entraînement. Il est notable qu'il y a une baisse de précision plus significative dans le scénario 3.2 (modèle de bruit complexe uniquement en phase de test), avec une diminution moyenne de 4.5% (indiquée par le boxplot rose), par rapport à une baisse mineure de 0.2% dans le scénario 3.1 (modèle de bruit simplifié uniquement en phase de test) (montrée par le boxplot jaune). Cela souligne l'impact des différentes considérations de bruit sur la robustesse du réseau en phases de test.

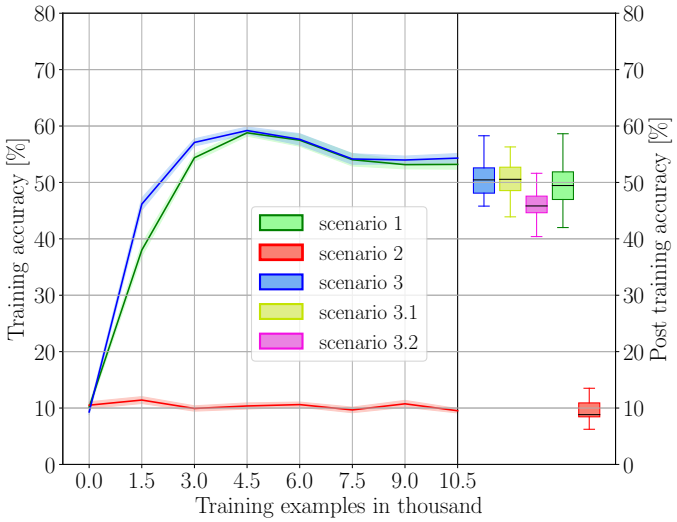


Figure 6.5: Résultats de précision pour le A-SNN dans la résolution du problème MNIST, en utilisant le eNeuron s-ML. Les scénarios restreints et non restreints sont considérés pour l'entraînement et le test.

6 . Conclusion

La croissance rapide de l'Internet des Objets (IoT) nécessite des solutions avancées basées sur l'IA en périphérie qui réduisent les besoins énergétiques pour la transmission des données, améliorent la confidentialité et rendent les dispositifs plus intelligents et autonomes. Le neuromorphisme, inspirée par les fonctionnalités du cerveau, propose une approche prometteuse pour répondre à ces besoins. Cette thèse explore le potentiel du neuromorphique pour satisfaire les demandes d'efficacité énergétique des applications IoT, avec un focus particulier sur la localisation par fréquence radio (RF). Elle introduit un système neuromorphique analogique complet, RF NeuroAS, conçu pour une localisation RF précise et économe en énergie. Ce système est capable d'identifier des positions sources dans une gamme complète de 360 degrés sur un plan bidimensionnel, avec une résolution de 1 ou 10 degrés, même dans des conditions bruyantes. La thèse examine également en profondeur le processus d'apprentissage pour le réseau neuronal analogique, un élément clé de RF NeuroAS. Deux méthodes d'apprentissage pour le A-SNN ont été explorées: l'apprentissage via le deep learning (DL) et la plasticité dépendante du temps des impulsions (STDP). Les résultats démontrent la robustesse du RF NeuroAS en termes de performance et sa capacité à maintenir une architecture simple et à faible puissance, bien adaptée à la localisation RF efficace. De plus, les résultats confirment l'efficacité de l'apprentissage via le DL et la STDP, tout en mettant en évidence les contraintes et les limitations associées. Ces résultats marquent des avancées dans les applications IoT économes en énergie grâce au neuromorphique, promettant des percées dans l'IoT intelligent à faible consommation d'énergie inspirées par les mécanismes du cerveau.

Appendices

Appendix A

RF NeuroAS System with 1-degree Angular Resolution

1 . System Architecture

The RF NeuroAS, an analog spiking-rate neuromorphic system designed for efficient RF source localization, is discussed in detail in Chap. 3. This system can pinpoint the source positions with a 10-degree angular resolution. An enhanced version of the RF NeuroAS system, featuring a refined 1-degree resolution, is presented in this appendix. Like its 10-degree counterpart, this system also employs received signal strength (RSS) detection and spiking-rate computing, thus bypassing the need for signal digitization. The system-level architecture of the RF NeuroAS, optimized for 1-degree resolution source localization, is depicted in Fig. A.1. It includes four principal stages: setting up the RF configuration for transmitter and receiver placement, extracting data for signal capture and dataset generation, neuromorphic pre-processing to transform RF signals into spike trains, and an analog spiking-rate neural network (A-SNN) to estimate source positions.

The RF configuration setup for the 1-degree resolution RF NeuroAS system mirrors that of the 10-degree version, extensively outlined in Sec. 3.1. This setup includes one mobile source and four stationary receivers arranged on a 2D plane. The source's position is determined by coordinates, defined by a distance (d_s) and an angle (θ_s) from the origin of the plane, with the distance being either 0.1, 0.3, or 0.5 meters and the angle ranging from 0 to 360 degrees in 1-degree increments. Receivers are strategically placed at the midpoint of each boundary of the plane to maximize coverage and enhance localization accuracy within the 2D configuration. Following this setup, signals are collected from various source positions under different noise conditions to create two critical datasets. These datasets are essential for evaluating the decision-making efficiency of the RF NeuroAS system. The first, SimLocRF, is a simulated dataset generated in MATLAB, and the second, MeasLocRF, consists of empirical data obtained through measurements, both created using the methodology described in Sec. 3.2. The third stage of the system, the neuromorphic pre-processing stage,is identical to that used in the 10-degree system and detailed in Sec. 3.3, converting the RF signals from these datasets into spike trains using eNeurons.

The final stage,the A-SNN, marks the primary distinction between the two versions of the RF NeuroAS system, featuring here a more complex structure to achieve higher resolution. As illustrated in Fig. A.1, the network integrates four ML eNeurons from the neuromorphic pre-processing stage as its input layer. It includes three densely connected hidden layers, each composed of only 8 eNeurons, culminating in an output layer. These ML eNeurons are chosen for their biomimetic properties and broad dynamic range, which boost learning efficiency [23]. The

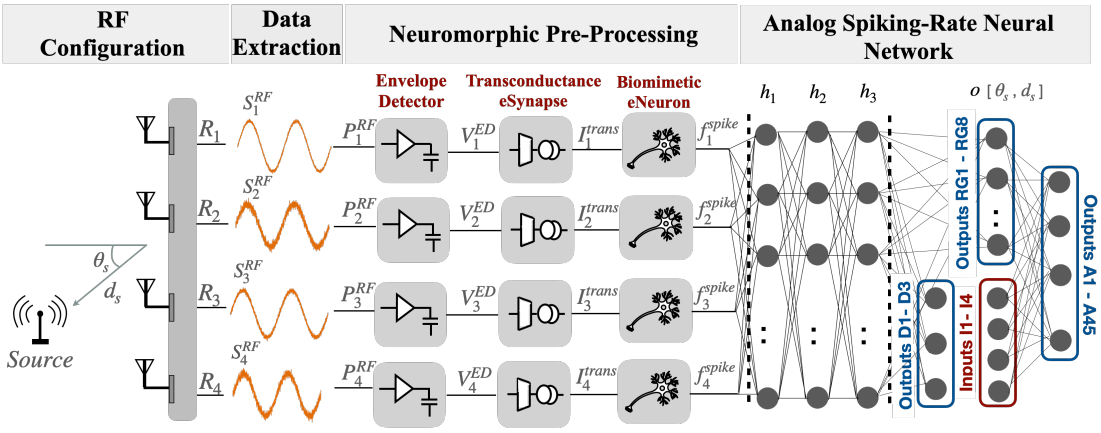
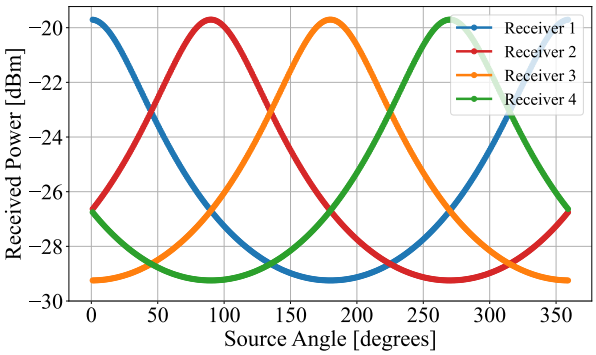


Figure A.1: System-level architecture of RF NeuroAS for 1-degree angular resolution in source localization, including four stages: RF configuration setup, data extraction, neuromorphic pre-processing, and the analog spiking neural network. The final outputs are the source's position, specified by distance d_s and angle θ_s.

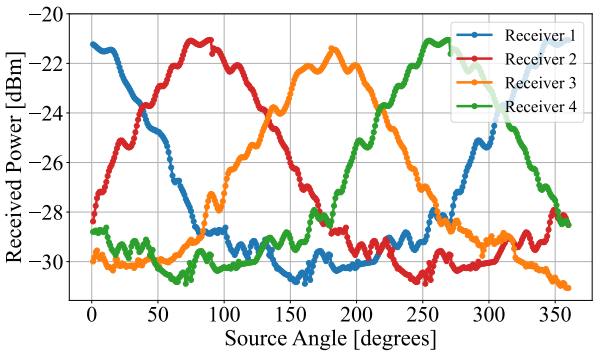
A-SNN uses a classification model to pinpoint source coordinates, starting with the identification of the distance d_s using three eNeurons D1-D3. For angle detection, the network innovatively uses fewer eNeurons for greater efficiency. Instead of allocating 360 eNeurons to cover the 360 possible angles, it first determines the correct region r_s on the plane , where the source is in the plane, through output eNeurons RG1-RG8. The specific angle within that region a_s is then identified using 45 eNeurons (A1-A45), achieving 1-degree resolution while conserving eNeuron resources. The source angle θ_s is computed as $r_s \times n_r + a_s$, where n_r, set at 45 degrees, denotes the range of each region. The system further refines angle prediction through a concatenation of input features from the input layer and the identified output regions. The learning process of the A-SNN is conducted via deep learning techniques, throughly described in Sec. 3.4.2. This approach integrates the analog characteristics of eNeurons and eSynapses within the TensorFlow platform, using their spiking rate post-layout transfer functions as activation functions. The network undergoes training and testing on the SimLocRF and MeasLocRF datasets within TensorFlow.

2 . Results

This section details the outcomes from the RF NeuroAS system configured for 1-degree angular resolution. Initially, it reviews the validation of dataset features used in system training, followed by the examination of its performance in RF source localization. Figure A.2 displays the power patterns captured by the four receivers over a complete 360-degree span, obtained from the SimLocRF dataset (Fig. A.2(a)) and the MeasLocRF dataset (Fig. A.2(b)). In both scenarios, the source, located 0.5 meters away from the origin, emits a 10 dBm signal in a setting with a 20 dB SNR. The power patterns from the simulated data in Fig. A.2(a) show a stable curve ranging from -20 to -29 dBm as the source rotates, which aligns with the dynamic range defined



(a)

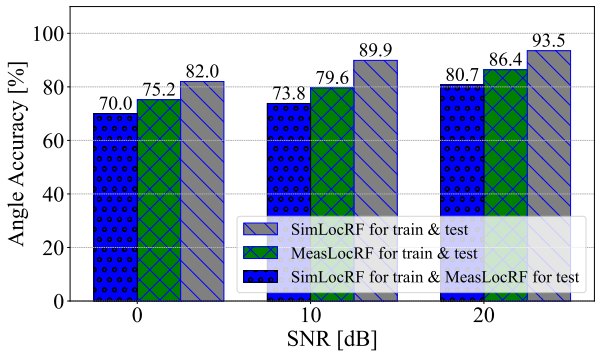


(b)

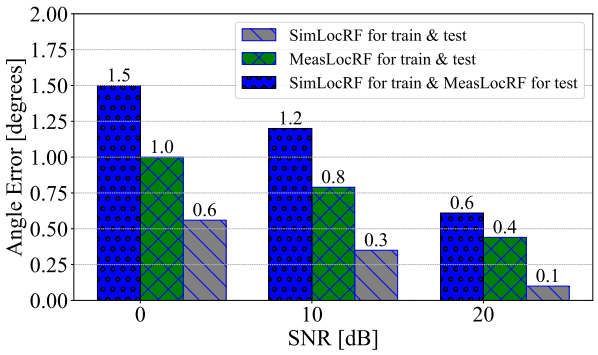
Figure A.2: Illustration of received power (in dBm) versus source sngle (0 to 360 Degrees) for four receivers: Panel (a) shows data from the SimLocRF dataset, and Panel (b) shows data from the MeasLocRF dataset.

by the neuromorphic pre-processing stage, as explained in Sec. 5.1.3. In contrast, the empirical data in Fig. A.2(b) reveal variations in received power between -20 and -31 dBm, attributable to inconsistencies in the anechoic chamber, such as variations in material properties, equipment performance, and antenna patterns.

Figure A.3 illustrates the angle accuracy (Fig. A.3(a)) and angle error (Fig. A.3(b)) under three different training and testing scenarios and at three distinct SNR levels. The datasets SimLocRF and MeasLocRF are partitioned such that 70% of the data is used for training and validation, with the remaining 30% designated for testing. Accuracy for both distance and region identification remains high (above 90%) due to the limited number of classes involved. The highest angle accuracy of 93.5% is achieved with the SimLocRF dataset in a low-noise environment (SNR = 20 dB). Angle accuracy decreases with lower SNR and when using the MeasLocRF dataset due to increased data variability. The minimum accuracy observed is 70%, which occurs in the noisiest setting (0 dB SNR) when training on SimLocRF data and testing on unseen MeasLocRF data.



(a)



(b)

Figure A.3: Performance of the A-SNN (1-degree resolution) in source positioning: Panel (a) details angle accuracy (%) and Panel (b) shows angle error (degrees) across three training and testing scenarios and three SNR levels.

Angle error is quantified by the formula $\sum_{i=1}^{N_f} |\theta s_i - \hat{\theta} s_i|$, where θs_i and $\hat{\theta}_{s_i}$ represent the actual and predicted source angles, respectively, across N_f test samples. As depicted in Fig. A.3(b), angle error typically remains below 1 degree in most cases, confirming the network's capability to maintain 1-degree angular resolution. In more challenging scenarios, such as in 0 dB SNR or when exclusively using the MeasLocRF dataset for testing, the angle error may slightly exceed 1 degree, indicating a drop in system resolution.

According to the power consumption estimation method described in [23], the RF NeuroAS system, which consists of 84 ML eNeurons and 788 eSynapses, consumes only 403 nW of power. This consumption is double that of the 10-degree resolution system as detailed in Sec. 5.1.4. These findings underscore the neural network's effectiveness in maintaining a low-power, efficient architecture suitable for precise RF localization.

Appendix B

Software Frameworks

Algorithm B.1 SimLocRF Dataset Generation in MATLAB, detailed in Sec. 3.2 - Part 1

```
1: 1. Initialize Parameters
2: 1.1. RF Environment Parameters
3:   parameters.f                                   ▷ Operating frequency
4:   parameters.c                                   ▷ Speed of light
5:   parameters.lambda                              ▷ Wavelength
6:   parameters.Pt                                  ▷ Transmit power
7:   parameters.Gt, parameters.Gr                   ▷ Transmit and Receive antenna gain
8: 1.2. RF Configuration Parameters
9:   parameters.num_antennas                        ▷ Number of antennas
10:  parameters.num_angles                          ▷ Number of angles
11:  parameters.d_s, parameters.theta_s             ▷ Source coordinates
12:  parameters.d_r, parameters.theta_r             ▷ Receivers coordinates
13: 1.3. Data Resolution and Quality Parameters
14:  parameters.snr_level                           ▷ Signal-to-Noise ratio
15:  parameters.resolution                          ▷ Angular resolution 1, 10 in degrees
16:  parameters.num_instances_per_angle             ▷ Data points per angle
17: 2. Generate Dataset
18:  [features, labels] = generateDataset(parameters)
19: 2.1.1. Generate Features as Powers
20:  features_Powers = zeros(num_angles * num_instances_per_angle,
     num_antennas)
21:  i = 1:parameters.num_angles
22:    n = 1:parameters.num_instances_per_angle
23:      j = 1:length(received_Power)               ▷ received_Power from Friis equation
24:        received_Signal = generateSignal(received_Power, f)
25:        noisy_received_Signal = addNoise(received_Signal, snr_level)
                                                    ▷ Add noise from AWGN function
26:        noisy_received_Power = mean((noisy_received_Signal)^2)
27:        features_Powers[instance] = watts2dbm(noisy_received_Power)
```

Algorithm B.1 SimLocRF Dataset Generation in MATLAB, detailed in Sec. 3.2 - Part 2

29: **2.1.2. Generate Features as Spiking Frequencies from Pre-Processing Stage**
30: `[Power, spiking_Frequency] = loadPostLayoutData()`
▷ Load Data from pre-processing (NWR) Stage
31: `features_Frequencies = interp1(Power, spiking_Frequency, features_Powers)` ▷ Interpolate Features
32: **2.2. Generate labels**
33: `labels = zeros(parameters.num_angles * parameters.num_instances_per_angle, parameters.num_labels)`
▷ Assigns two labels: angle and distance (distances omitted here for simplicity, values discussed in thesis = 0.1, 0.3, 0.5 m)
34: `for` i `= 1 to parameters.num_angles`
35: `for` n `= 1 to parameters.num_instances_per_angle`
36: **2.2.1. Identify Region Index and Angle Index within Region**
37: `region_index = floor(theta_s[`i`] / parameters.region_range)`
▷ Range of angles per region given by: parameters.region_range = parameters.num_angles / parameters.num_regions
38: `angle_within_region = rem(theta_s[`i`] , parameters.region_range)`
39: `angle_within_region_index = floor(angle_within_region / parameters.resolution)`
40: `if (region_index >= parameters.num_regions) then`
41: `region_index = 0` ▷ Adjust for angle wrap-around
42: **2.2.2. Assign Label**
43: `labels[`i `* `n`, :] = [region_index, angle_within_region_index]`

Appendix C

BiCMOS SiGe 55 nm Technology

The BiCMOS SiGe 55 nm technology from ST Microelectronics (B55) offers low-threshold voltage, low power MOS transistors, referred to as nlvtlp and plvtlp [228]. These transistors have a high I_{ON}/I_{OFF} ratio of approximately 150k, making them ideal for microwave applications, including self-oscillation frequencies (f_{max}) exceeding 110 GHz. Additionally, B55 technology allows for the integration of high-quality passive components such as inductors, transmission lines, and varactors, with quality factors greater than 10. The technology includes 8 metal levels and a top copper metal layer that is 3 μm thick. It provides mature BSIM4 and PSP models that account for process variability and temperature variations from -40°C to 175°C, based on silicon measurements. For parameter extraction in B55, designers need to analyze models that fit transistor characteristics in either strong inversion (SI), moderate inversion (MI), or weak inversion regions (also named subthreshold region) (WI). Since this thesis focuses on designing circuits in the subthreshold region for low power consumption, this appendix will concentrate on the analysis within this region.

1 . Parameters Extraction

In this section, parameter extraction follows the methodology presented in [229], which is inspired by the g_m/I_D-based procedure outlined in [230]. This approach relies on the applied basic (long channel) compact model (ACM). In the ACM model, the drain current of a transistor is expressed as the sum of two currents: the forward and reverse currents, as follows

$$I_D = I_F - I_R = I_S(i_f - i_r), \tag{C.1}$$

where I_S is the specific current, and i_f and i_r are the forward and reverse inversion coefficients respectively. I_S is given by

$$I_S = \mu \cdot n \cdot C'_{ox} \frac{\phi_T^2}{2} \frac{W}{L}, \tag{C.2}$$

where μ is the mobility, n is the slope factor, $C'ox$ is the oxide capacitance per unit area, ϕ_T is the thermal voltage ($kT/q \approx 26$ mV at 27 °C), and W/L is the aspect ratio.

The relationship between the terminal voltages V_G, V_S, V_D (all referenced to the bulk) and the inversion coefficients is

$$\frac{V_P - V_{S(D)}}{\phi_t} = \sqrt{1 + i_{f(r)}} - 2 + \ln\left(\sqrt{1 + i_{f(r)}} - 1\right), \tag{C.3}$$

where the pinch-off voltage V_P can be approximated by $V_P \approx (V_G\text{-}V_T)/n$, with V_T being the threshold voltage at $V_S = 0$. The ratio g_m/I_D can be expressed as

$$\frac{g_m}{I_D} = \frac{2}{n\phi_t\left(\sqrt{1+i_f}+\sqrt{1+i_r}\right)}. \tag{C.4}$$

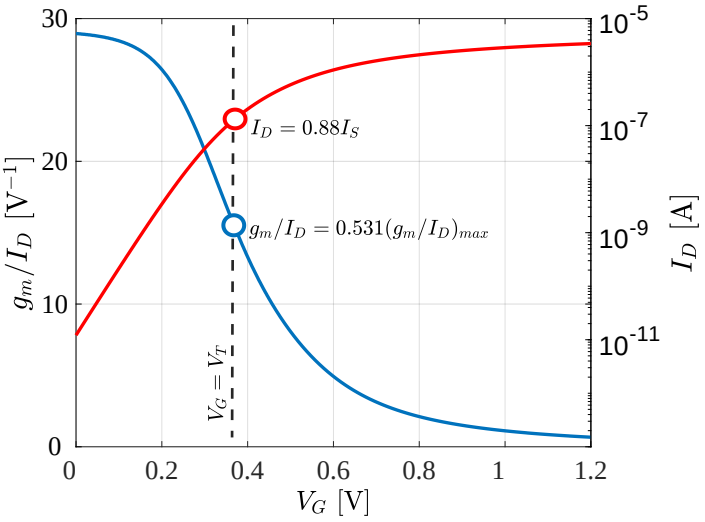


Figure C.1: Plot illustrating the variation of g_m/I_D (transconductance-to-current ratio) and I_D (drain current) as functions of V_{GB} (gate-to-bulk voltage), used for parameters extraction.

Based on these equations and the plots of g_m/I_D and I_D with respect to V_{GB} (see Fig. C.1), the parameters to be extracted are n, V_T, and I_S using transistor voltages as: $V_{SB} = 0$, $V_{DB} = \phi_T/2$, and sweeping V_{GB}.

- From $(g_m/I_D)_{max} = 1/n\phi_T$, where $i_f > 0$ and $i_r > 0$, $(g_m/I_D)_{max}$ is equal to $1/n\phi_T$.
- Using (C.3), for $V_{SB} = 0$ and $V_{GB} = V_T$ (which implies $V_P = 0$), $i_f = 3$ is obtained.
- Using (C.3), for $V_{DS} = \phi_T/2$, $V_{SB} = 0$, and $V_{GB} = V_T$, $i_r = 2.12$ is obtained.
- For these values of i_f and i_r, $g_m/I_D = 0.531/\ n\phi_T$ from (C.4) and $I_D = 0.88I_S$ from (C.1).
- Extract the parameters:
  - $n = 1/\ (\phi_T\ ((g_m/I_D)_{max}) = 1.33$,
  - V_T such that $g_m/I_D\ (V_{GB} = V_T) = 0.531/n\phi_T$ is determined, $V_T = 0.37$ V.
  - $I_S = I_D\ (V_{GB} = V_T)\ /0.88 = 1.5$ x 10^{-7} A.

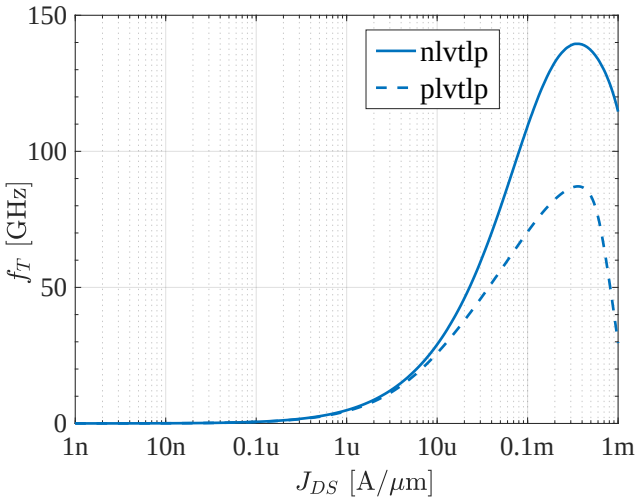
2 . Transistor Performance

In this thesis, the circuit designs use low-threshold voltage, low power NMOS and PMOS transistors referred to as "nlvtlp" and "plvtlp", respectively. The performance characteristics of these transistors are depicted in Fig. C.2: the transition frequency (f_T) and the transconductance efficiency (g_m/I_D), both as functions of the drain-to-source current density (J_{DS}).

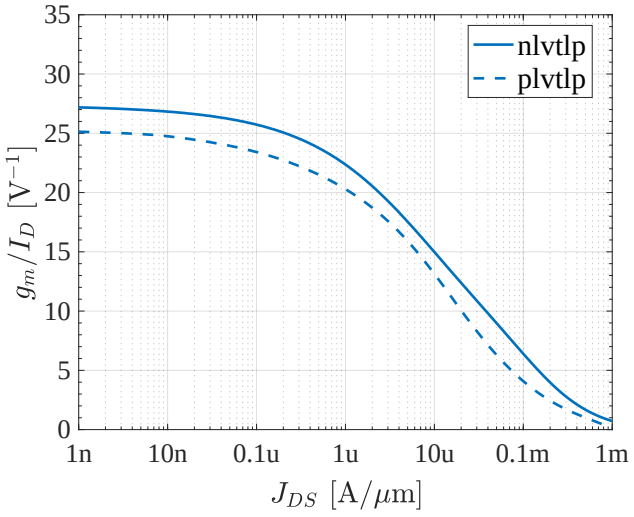
The transition frequency (f_T), shown in Fig. C.2(a), represents the frequency at which the current gain of a transistor drops to one, a crucial parameter for high-frequency applications. For both nlvtlp and plvtlp transistors, f_T increases with J_{DS} initially, indicating enhanced performance at higher currents due to improved carrier transport. However, beyond a certain current density, f_T begins to decline, likely due to parasitic effects and other high-current phenomena that degrade the transistors' performance. Notably, the nlvtlp transistor exhibits a higher peak frequency, suggesting superior performance in high-frequency applications compared to the plvtlp transistor.

Figure C.2(b) illustrates the transconductance efficiency (g_m/I_D), a key parameter for low-power and analog applications, as it measures the efficiency of converting input voltage signals into output current. Both transistor types show a decrease in g_m/I_D with increasing J_{DS}, indicating reduced efficiency at higher current densities, likely due to increased scattering and thermal effects. It is important to note that when the g_m/I_D ratio exceeds 25 V^{-1}, the transistor is typically operating in the subthreshold region. In this region, the transistor operates below the threshold voltage, resulting in high transconductance efficiency due to the exponential relationship between the gate voltage and the drain current. This high g_m/I_D ratio is advantageous for ultra-low power applications, as it allows for significant current generation with minimal gate drive voltage.

These performance characteristics highlight the advantages of the nmvtlp and plvtlp transistors in high-speed and low-power applications within the technology used, suitable for this thesis objective.



(a)



(b)

Figure C.2: B55 characteristics extraction for low-threshold voltage, low power NMOS (nlvtlp) and PMOS (plvtlp) transistors, being (a) the transition frequency f_T and (b) the g_m/I_D versus the drain-to-source current density J_{DS}.

Appendix D

Layouts and Components Size

1 . Simplified RF NeuroAS System

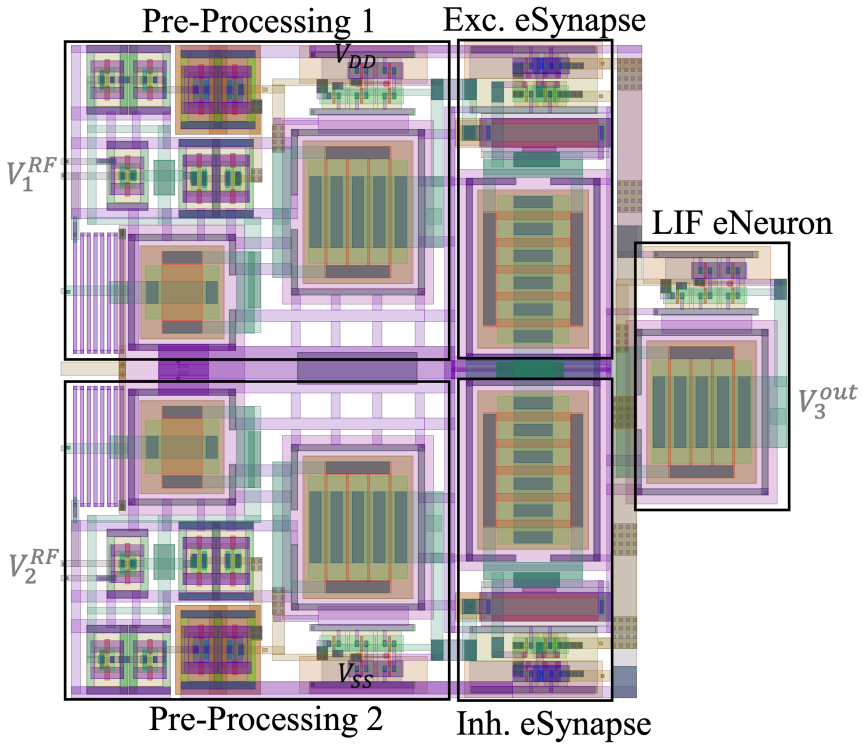
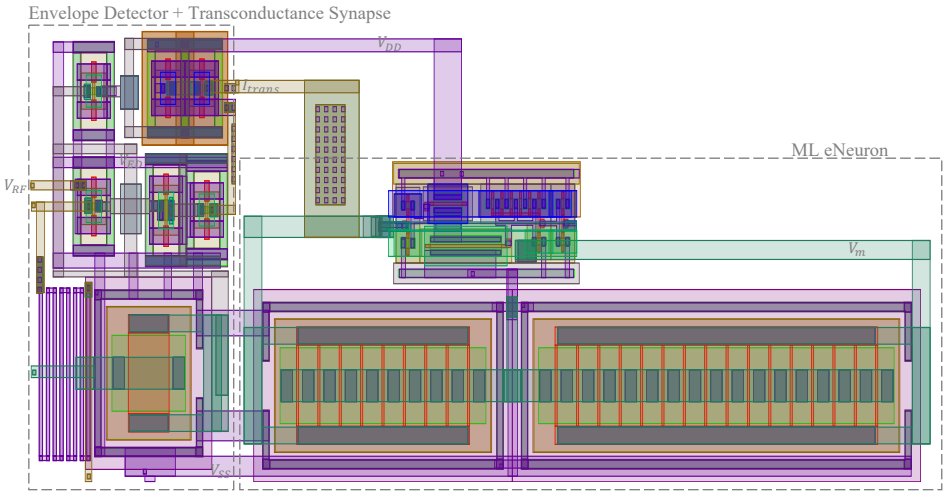
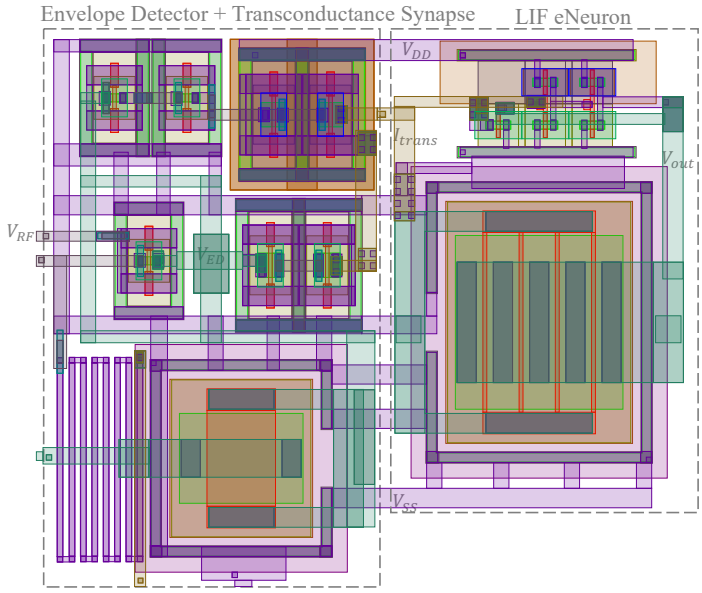


Figure D.1: Layout of Simplified RF NeuroAS system, depicted in Fig. 3.10, with dimensions $18.3 \times 20.24\ \mu m^2$.

1.1 . ML and LIF-based NWR Systems



(a)



(b)

Figure D.2: Layout of (a) ML-based and (b) LIF-based NWR systems, with circuits shown in Fig. 3.7(a) and Fig. 3.7(b) respectively, occupying areas of 9.8 × 25.09 μm^2 and 9.03 × 10.52 μm^2, respectively.

Table D.1: Sizing of transistors in W×L (nm) and of capacitances in number of cells × unity capacitance (fF) for the Simplified RF NeuroAS system, and ML and LIF-based NWRs.

Envelope Detector and Transconductance eSynapse			
M_{CG}	135×60	M_c	135×60
$C_{LP,ED}$	1×8.5	$M_{LP,ED}$	135×60
M_1	405×60	M_2*	135×60
M_3*	135×60	M_4*	135×60
Morris-Lecar (ML) eNeuron			
MP_1	135×60	MN_1	200×60
MP_2	1200×60	MN_2	135×60
MP_3	200×60	MN_3	135×60
MP_{Na}	800×60	MN_K	1500×60
C_m	1×9.83	C_K	1×5.53
Leaky Integrate-and-Fire (LIF) eNeuron			
MP_1	135×60	MN_1	135×60
MP_2	135×60	MN_2	135×60
C_f	1×5.038	MN_3	135×60
Bifunctional eSynapse			
$M_{LP,S}$	500×3000	$C_{LP,S}$	1×11.9
$M_{T,S}$	135×60	MP_{S1}	135×60
MN_{S1}	135×60	MP_{S2}	135×60
MN_{S2}	135×60	MP_{S3}	135×60

* Width of the transistor multiplied by 3 for LIF-based NWR

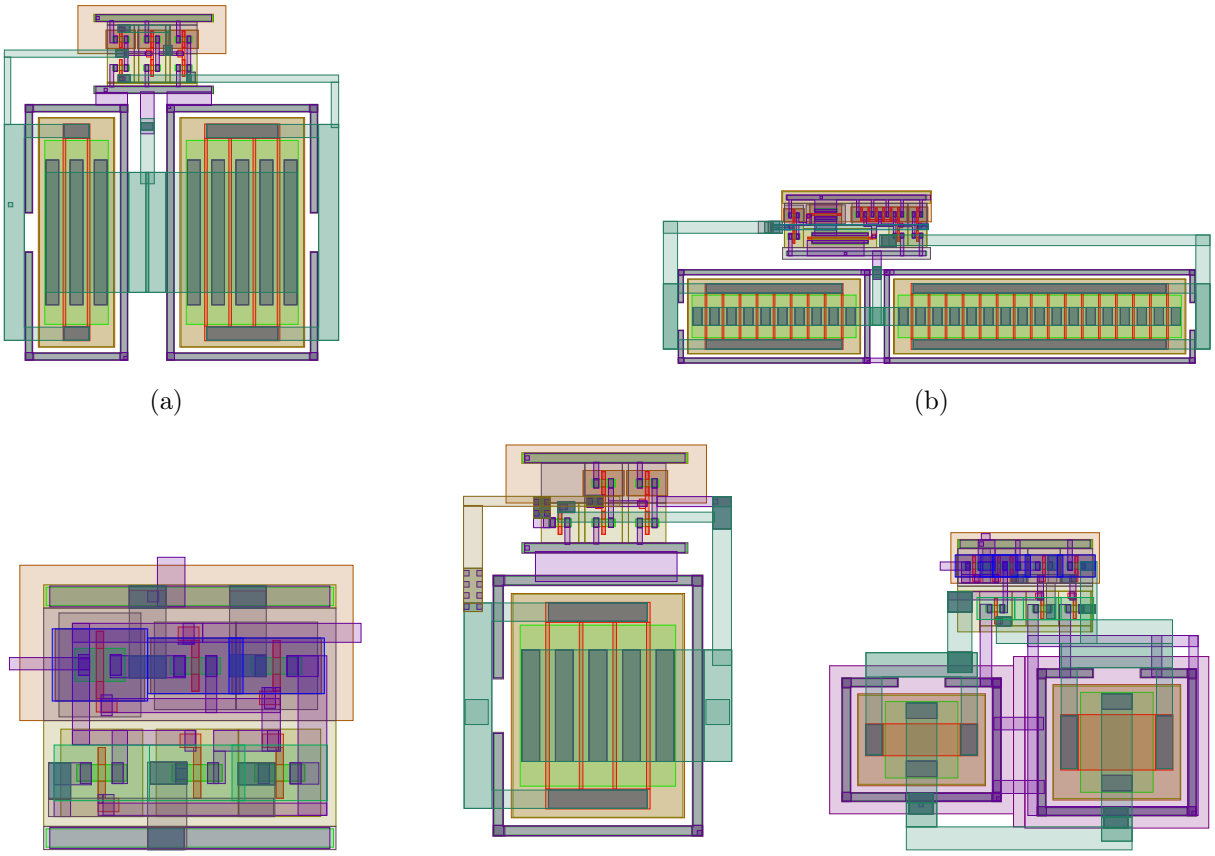


Figure D.3: Layouts of eNeurons under test: (a) s-ML using 8.2 × 7.5 μm^2 [153], (b) b-ML using 5.7 × 17.3 μm^2 [156], (c) p-ML using g 2.3 × 2.7 μm^2 [158], (d) AH-LIF using 6.6 × 4.3 μm^2 [155], and t-LIF using 6.5 × 7.8 μm^2 [157]. Circuit designs are illustrated in Fig. 2.9.

Table D.2: Sizing of transistors in W×L (nm) and of capacitances (fF) for the eNeurons

AH-LIF eNeuron			
MP_1, MN_1, MP_2, MN_2, MN_3			135×65
C_f	5.0		
s-ML eNeuron			
MP_1, MN_1, MP_2, MN_2, MP_{Na}, MN_K			135×65
C_m	4.0	C_K	8.0
b-ML eNeuron			
MP_1	135×60	MN_1	200×60
MP_2	1200×60	MN_2	135×60
MP_3	200×60	MN_3	135×60
MP_{Na}	800×60	MN_K	2000×60
C_m	5.5	C_K	9.8
p-ML eNeuron			
MP_1, MN_1, MP_2, MN_2, MP_{Na}, MN_K			135×65
t-LIF eNeuron			
MP_1, MN_1, MP_2, MN_2, MP_{Na}, MN_K			135×65
C_m	5.5	C_{res}	9.8

3 . Excitatory eSynapse

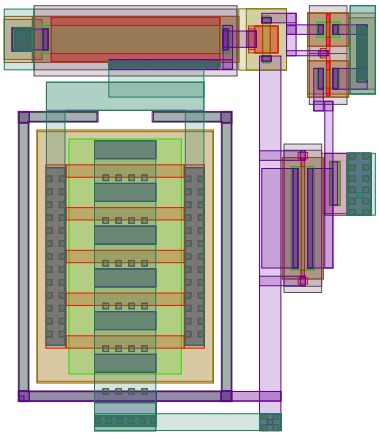


Figure D.4: Layout of Excitatory eSynapse, with circuit shown in Fig. 2.10, occupying 6.6 × 7.5 μm^2.

Table D.3: Sizing of transistors in W×L (nm) and of capacitances (fF) for the Excitatory eSynapse

Excitatory eSynapse			
MP_R	500 × 3000	MN_1	135 × 480
MP_1	270 × 60	MP_2	324 × 60
C	11.9		

References

[1] K. Ashton, "That 'Internet of Things' Thing", Online at https://www.rfidjournal.com/that-internet-of-things-thing

[2] D. C. Nguyen, et al., "6G Internet of Things: A Comprehensive Survey", *IEEE Internet of Things Journal*, vol. 9, n. 1, 2022. doi: 10.1109/JIOT.2021.3103320

[3] IoT Analytics Research, State of IoT 2023: the number of global IoT connections grew by 18% in 2022 to 14.3 billion active IoT endpoints. Online at https://iot-analytics.com/number-connected-iot-devices/

[4] J. Pan, and J. McElhannon, "Future Edge Cloud and Edge Computing for Internet of Things Applications", *IEEE Internet of Things Journal*, vol. 5, n.1, 2018. doi: 10.1109/JIOT.2017.2767608

[5] R. Singh, and S. S. Gill, "Edge AI: a survey", *Internet ofThings and Cyber-Physical Systems*, vol. 3, 2023. doi: 10.1016/j.iotcps.2023.02.004

[6] Y. Shi, et al., "Communication-Efficient Edge AI: Algorithms and Systems", *IEEE Communications Surveys & Tutorials*, vol. 22, n. 4, 2020. doi: 10.1109/COMST.2020.3007787

[7] Y. Lecun, et al., "Deep learning", *Nature*, vol. 521, n. 7553, 2015. doi: 10.1038/nature14539

[8] J. Sevilla, et al., "Compute Trends Across Three Eras of Machine Learning", *International Joint Conference on Neural Networks*, Padua, Italy, July, 2022. doi: 10.1109/IJCNN55064.2022.9891914.

[9] X. W. Chen, and X. Lin, "Big Data Deep Learning: Challenges and Perspectives", *IEEE Access*, vol. 2, 2014. doi: 10.1109/ACCESS.2014.2325029.

[10] T. Yarally, et al., "Uncovering Energy-Efficient Practices in Deep Learning Training: Preliminary Steps Towards Green AI", *IEEE ACM 2nd International Conference on AI Engineering - Software Engineering for AI*, Melbourne, Asutralia, May. 2023. doi: 10.1109/CAIN58948.2023.00012

[11] C. Schuman, et al., "Opportunities for neuromorphic computing algorithms and applications", *Nature Computational Science*, vol. 2, n. 1, 2022. doi: 10.1038/s43588-021-00184-y

[12] A. Shrestha, et al., "A survey on neuromorphic computing: Models and hardware", *IEEE Circuits and Systems Magazine*, vol. 22, n. 2, 2022. doi: 10.1109/MCAS.2022.3166331

[13] C. Schuman, et al., "A Survey of Neuromorphic Computing and Neural Networks in Hardware", 2017. http://arxiv.org/abs/1705.06963

[14] S. C. Liu, and T. Delbruck, "Neuromorphic sensory systems", *Current Opinion in Neurobiology*, vol. 20, n. 3, 2010. doi: 10.1016/j.conb.2010.03.007

[15] T. Wan, et al., "Neuromorphic sensory computing", *Science China Information Sciences*, vol. 65, n. 4, 2022. doi: 10.1007/s11432-021-3336-8

[16] Jouni, Z., Wang, S., Benlarbi-Delai, A., Ferreira, P. M., "Energy-Efficient Neuromorphic RF Localization using Analog Spiking Neurons", *IEEE Transactions on Circuits and Systems for Artificial Intelligence*, vol. x, n. x

[17] Jouni, Z., Wang, S., Benlarbi-Delai, A., Ferreira, P. M.,"Analog Spiking-Rate Neuromorphic System for Efficient RF Source Localization", *IEEE 31th Conference on Circuits and Systems (ICECS)*, pp

[18] Jouni, Z., Soupizet, T., Wang, S., Benlarbi-Delai, A., Ferreira, P. M., "RF Neuromorphic Spiking Sensor for IoT Devices", *Analog Integrated Circuits and Signal Processing*", vol. 117, pp 3-20, 2023, doi: 10.1007/s10470-023-02164-w

[19] Jouni, Z., Soupizet, T., Wang, S., Benlarbi-Delai, A., Ferreira, P. M., "1.2 nW Neuromorphic Enhanced Wake-Up Radio", *IEEE 35th SBC Symposium on Integrated Circuits and Systems Design (SBCCI)*, Porto Alegre, Brazil, 2022, pp. 1-6, doi: 10.1109/SBCCI55532.2022.9893247

[20] Ferreira, P. M., Jouni, Z., Wang, S., Klisnick, G., Benlarbi-Delai, A., "Neuromorphic-Enhanced Wake-up Radio for a Context-Aware IoT", *IEEE Design & Test*, vol. x, n.x

[21] Jouni, Z., Prats, T., Wang, S., Benlarbi-Delai, A., Ferreira, P. M., "Jitter Noise Impact on Analog Spiking Neural Networks: STDP Limitations", *IEEEE 36th SBC Symposium on Integrated Circuits and Systems Design (SBCCI)*, Rio de Janeiro, Brazil, 2023, pp. 1-6, doi: 10.1109/SBCCI60457.2023.10261661

[22] Bossard, Y., Jouni, Z., Wang, S., Ferreira, P. M., "Analog Spiking Neuron Model for Unsupervised STDP-based learning in Neuromorphic Circuits", *Journal of Integrated Circuits and Systems*, vol. x, n.x

[23] Soupizet, T., Jouni, Z., Wang, S., Benlarbi-Delai, A., Ferreira, P. M.," Analog Neural Network Synthesis for the MNIST", *Journal of Integrated Circuits and Systems*", vol. 18, n. 1, pp 1-12, 2023, doi: 10.29292/jics.v18i1.663

[24] Soupizet, T., Jouni, Z., Sulzbach, J. F., Benlarbi-Delai, A., Ferreira, P. M., "Deep Neural Network Feasibility Using Analog Spiking Neurons", *IEEE 35th SBC Symposium on Integrated Circuits and Systems Design (SBCCI)*, Porto Alegre, Brazil, 2022, pp. 1-6, doi: 10.1109/SBCCI55532.2022.9893216

[25] Prats, T., Ramos, P., Jouni, Z., Wang, S., Benlarbi-Delai, A., Ferreira, P. M., "Energy-Efficient and Robust-Trained Analog Neural Network Synthesis: a Design Framework for Multilayer Perceptron Problem", *IEEE Open Journal of Circuits and Systems*, vol. x, n.x.

[26] Prats, T., Jouni, Z., Wang, S., Benlarbi-Delai, A., Ferreira, P. M., "Revisiting the Ultra-Low Power Electronic Neuron Towards a Faithful Biomimetic Behavior", *IEEE 36th SBC Symposium on Integrated Circuits and Systems Design (SBCCI)*, Rio de Janeiro, Brazil, 2023, pp. 1-6, doi: 10.1109/SBCCI60457.2023.10261961

[27] Sulzbach, J. F., Wang, S., Jouni, Z., Benlarbi-Delai, A., Klisnick, G., Ferreira, P. M., "Sub-nJ per Decision Schmitt Trigger Comparator for Neuromorphic Spike Detection in 55 nm Technology", *IEEE 7th Forum on Research and Technologies for Society and Industry Innovation (RTSI)*, Paris, France, 2022, pp. 1-6, doi: 10.1109/rtsi55261.2022.9905228

[28] Z. Jouni, S. Wang, and P. M. Ferreira, "Analog Neuromorphic System for Low-Power RF Localization in IoT," Open Softw., 2024, doi: 10.5281/zenodo.12808148.

[29] Y. Bossard, Z. Jouni, S. Wang, and P. M. Ferreira, "Analog Spiking Neuron Model for Unsupervised STDP-based Learning in Neuromorphic Circuits," Open Softw., 2024, doi: 10.5281/zenodo.12839336.

[30] F. Khelifi, et al., "A Survey of Localization Systems in Internet of Things", *Mobile Networks and Applications*, vol. 24, n. 3, 2019. doi: 10.1007/s11036-018-1090-3

[31] A. F. G. Ferreira, et al., "Localization and Positioning Systems for Emergency Responders: A Survey", *IEEE Communications Surveys and Tutorial*, vol. 19, n. 4, 2017. doi: 10.1109/COMST.2017.2703620

[32] T. D. McAllister, et al., "Localization of Health Center Assets Through an IoT Environment (LoCATE)", *Systems and Information Engineering Design Symposium*, Charlottesville, USA, Apr. 2017. doi: 10.1109/SIEDS.2017.7937703

[33] C. Bradley, et al., "Security analysis of an IoT system used for indoor localization in healthcare facilities", *IEEE Systems and Information Engineering Design Symposium*, Charlottesville, USA, Apr. 2018. doi: 10.1109/SIEDS.2018.8374726

[34] C. Laoudias, et al., "A survey of enabling technologies for network localization, tracking, and navigation", *EEE Communications Surveys and Tutorials*, vol. 20, n. 4, 2018. doi: 10.1109/COMST.2018.2855063

[35] F. Zafari, et al., "A Survey of Indoor Localization Systems and Technologies", *IEEE Communications Surveys and Tutorials*, vol. 21, n. 3, 2019. doi: 10.1109/COMST.2019.2911558

[36] M. Maheepala, et al., "Light-Based Indoor Positioning Systems: A Review", *IEEE Sensors Journal*, vol. 20, n. 8, 2020. doi: 10.1109/JSEN.2020.2964380

[37] H. C. So, and L. Lin, "Linear least squares approach for accurate received signal strength based source localization", *IEEE Transactions on Signal Processing*, vol. 59, n. 8, 2011. doi: 10.1109/TSP.2011.2152400

[38] N. Patwari, et al., "Relative location in wireless networks", *IEEE VTS 53rd Vehicular Technology Conference*, Rhodes, Greece, May. 2001. doi: 10.1109/VETECS.2001.944560

[39] J. Zhang, and D. Tao, "Empowering Things with Intelligence: A Survey of the Progress, Challenges, and Opportunities in Artificial Intelligence of Things", *IEEE Internet of Things Journal*, vol. 8, n. 10, 2021. doi: 10.1109/JIOT.2020.3039359

[40] M. Haenlein, and A. Kaplan, "A brief history of artificial intelligence: On the past, present, and future of artificial intelligence", *California Management Review*, vol. 61, n. 4, 2019. doi: 10.1177/0008125619864925

[41] Anyoha Rockwell, "The History of Artificial Intelligence", *Harvard University*, 2017. https://sitn.hms.harvard.edu/flash/2017/history-artificial-intelligence/

[42] W. S. McCulloch, and W. Pitts, "A logical calculus of the ideas immanent in nervous activity", *The Bulletin of Mathematical Biophysics*, vol. 5, n. 4, 1943. doi: 10.1007/BF02478259

[43] Y. LeCun, et al., "Backpropagation applied to handwritten zip code recognition", *Neural computation*, vol. 1, n. 4, 1989.

[44] F. Rosenblatt, "Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms", *Spartan Books*, 1962.

[45] M. Minsky, and S.Papert, "An introduction to computational geometry", *The MIT Press*, 1969. doi: 10.7551/mitpress/11301.001.0001

[46] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators", *Neural Networks*, vol. 2, n. 5, 1989. doi: 10.1016/0893-6080(89)90020-8

[47] B. Widrow, and M. A. Lehr, "30 years of adaptive neural networks: perceptron, Madaline, and backpropagation," *Proceedings of the IEEE*, vol. 78, n. 9, 1990, doi: 10.1109/5.58323

[48] D. Rumelhart, G. Hinton, and R. Williams, "Learning representations by back-propagating errors", *Nature*, vol. 323, 1986. doi: 10.1038/323533a0

[49] Y. LeCun, et al., "Learning algorithms for classification: a comparison on handwritten digit recognition," *Neural networks: the statistical mechanics perspective*, vol. 1, 1995. doi: 10.1142/2808

[50] R. J. Williams and D. Zipser, "A Learning Algorithm for Continually Running Fully Recurrent Neural Networks," *Neural Computation*, vol. 1, n. 2, 1989. doi: 10.1162/neco.1989.1.2.270.

[51] A. Vaswani, et al., "Attention is all you need", *Curran Associates, Inc.*, vol. 30, 2017.

[52] L. Yang, and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice", *Neurocomputing*, vol. 415, 2020. doi: 10.1016/j.neucom.2020.07.061

[53] J. Flinn, H. Levy, "Project Adam: Building an Efficient and Scalable Deep Learning Training System", *Proceedings of the 11th USENIX Conference on Operating Systems Design and Implementation*, Broomfield, CO, Oct. 2014.

[54] L. Ciampiconi, et al., "A survey and taxonomy of loss functions in machine learning", *arXiv*, 2023. doi: 10.48550/arXiv.2301.05579

[55] R. Mahima, et al., "A Comparative Analysis of the Most Commonly Used Activation Functions in Deep Neural Network", *4th International Conference on Electronics and Sustainable Communication Systems, ICESC*, Coimbatore, India, Jul. 2023. doi: 10.1109/ICESC57686.2023.10193390

[56] Wikipedia: Sigmoid Function. https://en.wikipedia.org/wiki/Sigmoid_function

[57] Y. LeCun, et al., "Efficient backpropagation. In Neural networks: Tricks of the trade "(pp. 9-50), *Springer*, Berlin, Heidelberg, 1998. doi: 10.1007/978-3-642-35289-8

[58] C. M. Bishop, "Neural Networks for Pattern Recognition," *Oxford University Press*, 1995

[59] S. Narayan, et al., "The generalized sigmoid activation function: Competitive supervised learning", *Information Sciences*, vol. 99, n. 1. doi: 10.1016/S0020-0255(96)00200-9

[60] Wikipedia: Softmax Function. https://en.wikipedia.org/wiki/Softmax_function

[61] J. S. Bridle, "Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition" (pp. 227-236), *Neurocomputing: Algorithms, Architectures and Applications*, *Springer*, Berlin, Heidelberg, 1990. doi: 10.1007/978-3-642-76153-9 28

[62] R. Memisevic, et al.,"Gated Softmax Classification", *Curran Associates, Inc.*, vol. 23, 2010.

[63] Wikipedia: Hyperbolic Functions. https://en.wikipedia.org/wiki/Hyperbolic_functions

[64] S. Hochreiter, and J. Schmidhuber, "Long short-term memory", *Neural Computation*, vol. 9, n. 8, 1997.

[65] A.D. Jagtap, and G.E. Karniadakis, "How important are activation functions in regression and classification? A survey, performance comparison, and future directions", 2022. https://doi.org/10.48550/arXiv.2209.02681

[66] Wikipedia: Rectifier (neural networks). https://en.wikipedia.org/wiki/Rectifier_(neural_networks)

[67] H. Ide, and T. Kurita, "Improvement of learning for CNN with ReLU activation by sparse regularization," *2017 International Joint Conference on Neural Networks*, Anchorage, AK, USA, May, 2017. doi: 10.1109/IJCNN.2017.7966185.

[68] M. Abadi, et al., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems", 2024, www.tensorflow.org

[69] M. Abadi, et al., "TensorFlow: A system for large-scale machine learning", *12th USENIX Symposium on Operating Systems Design and Implementation*, Savannah, GA, USA, Nov. 2016.

[70] A. Paszke, et al., "PyTorch: An imperative style, high-performance deep learning library", *Advances in neural information processing systems*, 2019.

[71] F. Chollet, "Keras", https://keras.io, 2015.

[72] E. Bisong, "Google Colaboratory", *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, Apress, Berkeley, CA, 2019. doi: 10.1007/978-1-4842-4470-8

[73] E. Liberty, et al., "Elastic machine learning algorithms in Amazon SageMaker", *ACM Queue*, vol. 18, n. 1, 2020.

[74] Azure Machine Learning. https://azure.microsoft.com/en-in/products/machine-learning

[75] G. E. Hinton, et al., "A fast learning algorithm for deep belief nets", *Neural computation*, vol. 18, n. 7, 2006. doi: 10.1162/neco.2006.18.7.1527

[76] R. Raina, et al., "Large-scale deep unsupervised learning using graphics processors", *Proceedings of the 26th annual international conference on machine learning*, Jun. 2009. doi: 10.1145/1553374.1553486

[77] B. Catanzaro, et al., "Fast support vector machine training and classification on graphics processors", *Machine Learning, Proceedings of the 25thInternational Conference*, Helsinki, Finland, June, 2008. doi: 10.1145/1390156.1390170

[78] Graves, et al., "Speech recognition with deep recurrent neural networks", *IEEE international conference on acoustics, speech and signal processing*, Vancouver, BC, Canada, May 2013. doi: 10.1109/ICASSP.2013.6638947

[79] K. Simonyan, and A. Zisserman, "Very deep convolutional networks for large-scale image recognition", *arXiv*, 2014. doi: https://doi.org/10.48550/arXiv.1409.1556

[80] A. Krizhevsky, et al., "ImageNet classification with deep convolutional neural networks", *Advances in Neural Information Processing Systems*, vol. 15, 2012.

[81] D. Silver, et al., "Mastering the game of Go with deep neural networks and tree search", *Nature*, vol. 529, 2016. doi: 10.1038/nature16961

[82] T. B. Brown, et al., "Language models are few-shot learners", *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[83] S. Bubeck, et al., "Sparks of artificial general intelligence: Early experiments with GPT-4", *arXiv*, 2020. doi: 10.48550/arXiv.2303.12712

[84] Hello GPT-4o, https://openai.com/index/hello-gpt-4o/

[85] E. Strubell, et al., "Energy and Policy Considerations for Deep Learning in NLP", *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, July 2019. doi: 10.18653/v1/P19-1355

[86] D. Blalock, et al., "What is the State of Neural Network Pruning?", *Proceedings of the Machine Learning and Systems*, 2020.

[87] U. M. Khaire, et al., "Stability of feature selection algorithm: A review", *Journal of King Saud University - Computer and Information Sciences*, vol. 34, n.4, 2022. doi: 10.1016/j.jksuci.2019.06.012

[88] A. G. Howard, et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications", *arXiv*, 2017. doi: 10.48550/arXiv.1704.04861

[89] Y. E. Wang, et al., "Benchmarking TPU, GPU, and CPU platforms for deep learning", *arXiv*, 2019. doi: 10.48550/arXiv.1907.10701

[90] J. Lee, and H. bahn, "Characterization of Memory Access in Deep Learning and Its Implications in Memory Management", *Computers, Materials, and Continua*, vol. 76, n. 1, 2023. doi: 10.32604/cmc.2023.039236

[91] J. von Neumann, "First draft of a report on the EDVAC", *IEEE Annals of the History of Computing*, vol. 15, no. 4, doi: 10.1109/85.238389.

[92] NVIDIA A100 Tensor Core GPU. https://www.nvidia.com/en-us/data-center/a100/

[93] J. Backus, "Can programming be liberated from the von Neumann style? A functional style and its algebra of programs", *Communications of the ACM*, vol. 21, n. 8, 1978. doi: 10.1145/359576.359579

[94] C. Mead, "Neuromorphic electronic systems", *Proceedings of the IEEE*, vol. 78, n. 10, 1990. doi: 10.1109/5.58356

[95] C. Mead, "Publisher Correction: How we created neuromorphic engineering", *Nature Electronics*, vol. 3, n. 502, 2020. doi: 10.1038/s41928-020-0462-4

[96] C. Frenkel, et al., "Bottom-Up and Top-Down Neural Processing Systems Design: Neuromorphic Intelligence as the Convergence of Natural and Artificial Intelligence", *arXiV*. doi: abs/2106.01288

[97] G. Indiveri, and Y. Sandamirskaya, "The Importance of Space and Time for Signal Processing in Neuromorphic Agents: The Challenge of Developing Low-Power, Autonomous Agents That Interact with the Environment", *IEEE Signal Processing Magazine*, vol. 36, n. 6, 2019. doi: 10.1109/MSP.2019.2928376

[98] V. Sze, et al., "Hardware for machine learning: Challenges and opportunities," *IEEE Custom Integrated Circuits Conference*, Austin, TX, USA, 2017. doi: 10.1109/CICC.2017.7993626

[99] C. D. James, et al., "A historical survey of algorithms and hardware architectures for neural-inspired and neuromorphic computing applications", *Biologically Inspired Cognitive Architectures*, vol. 19, 2017. doi: 10.1016/j.bica.2016.11.002.

[100] P. A. Merolla, et al., "A million spiking-neuron integrated circuit with a scalable communication network and interface", *Science*, vol. 345, 2014. doi:10.1126/science.1254642

[101] F. Akopyan, et al., "TrueNorth: Design and tool flow of a 65mW 1 million neuron programmable neurosynaptic chip", *IEEE transactions on computer-aided design ofintegrated circuits and systems*, vol. 34, n. 10, 2015. doi: 10.1109/TCAD.2015.2474396

[102] M. Davies, et al., "Loihi: A neuromorphic manycore processor with on-chip learning", *IEEE Micro*, vol. 38, n. 1, 2018. doi: 10.1109/MM.2018.112130359.

[103] G. Orchard, et al., "Efficient neuromorphic signal processing with Loihi 2", *IEEE Workshop on Signal Processing Systems*, Coimbra, Portugal, 2021. doi: 10.1109/SiPS52927.2021.00053.

[104] J. Schemmel, et al., "A wafer-scale neuromorphic hardware system for large-scale neural modeling," *IEEE International Symposium on Circuits and Systems*, Paris, France, 2010. doi: 10.1109/ISCAS.2010.5536970.

[105] J. Schemmel, et al., "Accelerated Analog Neuromorphic Computing", *Springer International Publishing*. doi: 10.1007/978-3-030-91741-8_6

[106] S. B. Furber,et al., "The SpiNNaker Project," *Proceedings of the IEEE*, vol. 102, n. 5,2014. doi: 10.1109/JPROC.2014.2304638.

[107] Introducing SpiNNaker2 – The Future of Hybrid Brain Inspired High Performance Computing https://spinncloud.com/introducing-spinnaker2-the-future-of-hybrid-brain-inspired-high-performance-computing/

[108] G. E. Alexander, and M. D. Crutcher, "Functional architecture of basal ganglia circuits: neural substrates of parallel processing", *Trends in neurosciences*, vol. 13, n. 7, 1990. doi: 10.1016/0166-2236(90)90107-L.

[109] M. E. Raichle, and D. A. Gusnard, "Appraising the brain's energy budget", Proceedings of the National Academy of Sciences", vol. 99, n. 16, 2002. http://www.jstor.org/stable/3059371

[110] C. S. Von Bartheld, et al., "The search for true numbers of neurons and glial cells in the human brain: A review of 150 years of cell counting", *Journal of Comparative Neurology*,vol. 524, n. 18, 2016.doi: 10.1002/cne.24040

[111] J. Zhang, "Basic neural units of the brain: neurons, synapses and action potential", *arXiv*, 2019. doi: 10.48550/arXiv.1906.01703

[112] S. Herculano-Houzel, "The human brain in numbers: a linearly scaled-up primate brain", *Frontiers in human neuroscience*, 2009. doi: 10.3389/neuro.09.031.2009

[113] R. Sylwester, "A celebration of neurons: An educator's guide to the human brain", *Book*, 1995.

[114] B. Hille, and W. A. Catterall, "Basic Neurochemistry: Molecular, Cellular and Medical Aspects", *Book*, 1999.

[115] Basic Structure of the Human Nervous System. https://mikerbio.weebly.com/structure--function.html

[116] What is an action potential? https://www.moleculardevices.com/applications/patch-clamp-electrophysiology/what-action-potential

[117] M. HGrider, R. Jessu, and R. Kabir, "Physiology, action potential", *Book*, 2019.

[118] T. C. Sudhof, and R. C. Malenka, "Understanding synapses: past, present, and future", *Neuron*, vol. 60, n. 3, 2008. doi: 10.1016/j.neuron.2008.10.011

[119] T. Trappenberg, "Fundamentals of computational neuroscience", *OUP Oxford Book*, 2009.

[120] J. CarewEccles, "The physiology of synapses", *Academic Press*, 2013.

[121] Excitatory synapse. https://en.wikipedia.org/wiki/Excitatory_synapse

[122] E. M. Izhikevich, "Which model to use for cortical spiking neurons?", *IEEE Transactions on Neural Networks*, vol. 15, n. 5, 2004. doi: 10.1109/TNN.2004.832719

[123] L. F. Abbott, "Lapicque's introduction of the integrate-and-fire model neuron (1907)", *Brain research bulletin*, vol. 50, n. 5-6, 1999. doi: 10.1016/s0361-9230(99)00161-6

[124] A. L. Hodgkin, and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve", *The Journal of physiology*, vol. 117, n. 4, 1952. doi: 10.1113/jphysiol.1952.sp004764

[125] R. FitzHugh, "Impulses and physiological states in theoretical models of nerve membrane", *Biophysical journal*, vol. 1, n. 6, 1961. doi: 10.1016/s0006-3495(61)86902-6

[126] J. Nagumo, S. Arimoto, and S. Yoshizawa, "An active pulse transmission line simulating nerve axon", *Proceedings ofthe IRE*, vol. 50, n. 10, 1962. doi: 10.1109/JRPROC.1962.288235.

[127] E. M. Izhikevich, and R. FitzHugh, "Fitzhugh-Nagumo model", *Scholarpedia*, vol. 1, n. 9, 2006. doi:10.4249/scholarpedia.1349

[128] C. Morris, and H. Lecar, "Voltage oscillations in the barnacle giant muscle fiber", *Biophys J.*, vol. 35, n. 1, 1981. doi: 10.1016/S0006-3495(81)84782-0.

[129] H. Lecar, "Morris-Lecar model", *Scholarpedia*, vol. 2, n. 10, 2007. doi: doi:10.4249/scholarpedia.1333

[130] E. M. Izhikevich, "Simple model of spiking neurons", *IEEE Transactions on Neural Networks*, vol. 14, n. 6, 2003. doi: 10.1109/TNN.2003.820440.

[131] S. Yu, and P. Y. Chen, "Emerging Memory Technologies: Recent Trends and Prospects", *IEEE Solid-State Circuits Magazine*, vol. 8, n. 2, 2016. doi: 10.1109/MSSC.2016.2546199.

[132] V. Milo, et al., "Memristive and CMOS Devices for Neuromorphic Computing", *Materials*, vol. 13, n. 1. doi: 10.3390/ma13010166.

[133] E. A. Vittoz, "Analog VLSI Implementation of Neural Networks", *IEEE Integrated Circuits and Systems*, Lausanne, Switzerland, 1989. doi: 10.1109/ISCAS.1990.112524

[134] D. B. Strukov, et al., "The missing memristor found", *Nature*, vol. 453, n. 7191, 2008. doi:10.1038/nature06932

[135] S. Kumar, et al., "Dynamical memristors for higher-complexity neuromorphic computing", .*Nature Reviews Materials*, vol. 7, 2022. doi: 10.1038/s41578-022-00434-z

[136] I. Zutic, et al., "Spintronics: Fundamentals and applications", *Reviews of Modern Physics*,vol. 76, n. 2, 2004. doi:10.1103/RevModPhys.76.323

[137] B. Dieny et al., "Opportunities and challenges for spintronics in the microelectronics industry," *Nature Electronics*, vol. 3, 2020. doi: 10.1038/s41928-020-0461-5

[138] N. Locatelli, V. Cros, and J. Grollier, "Spin-torque building blocks", *Nature Materials*, vol. 13, 2014. doi: 10.1038/nmat3823

[139] A. Joubert, et al., "Hardware spiking neurons design: Analog or digital?", *IEEE International Joint Conference on Neural Networks* , Brisbane, QLD, Australia, 2012. doi: 10.1109/IJCNN.2012.6252600

[140] L. Gatet, H. Tap-Beteille, and F. Bony, "Comparison Between Analog and Digital Neural Network Implementations for Range-Finding Applications", *IEEE Transactions on Neural Networks*, vol. 20, n. 3, 2009. doi: 10.1109/TNN.2008.2009120

[141] S. Ratté, et al., "Impact of Neuronal Properties on Network Coding: Roles of Spike Initiation Dynamics and Robust Synchrony Transfer", *Neuron*, vol. 78, n. 5, 2013. doi: 10.1016/j.neuron.2013.05.030.

[142] S. Prescott, "Excitability: Types I, II, and III", *Springer Science*, 2014. doi: 10.1007/978-1-4614-7320-6_151-1

[143] G. Indiveri, et al., "A VLSI array of low-power spiking neurons and bistable synapses with spike-timing dependent plasticity", *IEEE Transactions on Neurals Networks*, vol. 17, n. 1, 2006. doi: 10.1109/TNN.2005.860850.

[144] G. Indiveri, "A low-power adaptive integrate-and-fire neuron circuit", *International Symposium on Circuits and Systems*, Bangkok, Thailand, 2003. doi: 10.1109/ISCAS.2003.1206342.

[145] G. Indiveri, "Neuromorphic Silicon Neuron Circuits", *Frontiers in Neuroscience*, vol. 5, 2011. doi: 10.3389/fnins.2011.00073

[146] V. Rangan, et al., "A subthreshold aVLSI implementation of the Izhikevich simple neuron model", *IEEE Engineering in Medicine and Biology Society*, 2010. doi: 10.1109/IEMBS.2010.5627392

[147] A. S. Demirkol, and S. Ozoguz, "A low power VLSI implementation of the Izhikevich neuron model", *IEEE 9th Interregional NEWCAS Conference*, Bordeaux, France, 2011. doi: 10.1109/NEWCAS.2011.5981282

[148] O. O. Dutra, et al., "A sub-threshold halo implanted MOS implementation of Izhikevich neuron model", *IEEE SOI-3D-Subthreshold Microelectronics Technology Unified Conference*, Monterey, CA, USA, 2013. doi: 10.1109/S3S.2013.6716556

[149] M. Ronchini, et al., "Tunable Voltage-Mode Subthreshold CMOS Neuron," *IEEE Computer Society Annual Symposium on VLSI*, Limassol, Cyprus, 2020. doi: 10.1109/ISVLSI49217.2020.00053

[150] S. Moriya, et al., "A Fully Analog CMOS Implementation of a Two-variable Spiking Neuron in the Subthreshold Region and its Network Operation", *International Joint Conference on Neural Networks*, 2022. doi: 10.1109/IJCNN55064.2022.9891920

[151] S. A. Aamir, et al., "A highly tunable 65-nm CMOS LIF neuron for a large scale neuromorphic system", *42nd European Solid-State Circuits Conference*, Lausanne, Switzerland, 2016. doi: 10.1109/ESSCIRC.2016.7598245.

[152] B. Joo, J. W. Ha,n and B. -S. Kong, "Energy- and Area-Efficient CMOS Synapse and Neuron for Spiking Neural Networks With STDP Learning", *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, n. 9, 2022. doi: 10.1109/TCSI.2022.3178989.

[153] I. Sourikopoulos, et al., "A 4-fJ/Spike Artificial Neuron in 65 nm CMOS Technology", *Frontiers in Neuroscience*, vol. 11 (MAR). doi: 10.3389/fnins.2017.00123

[154] C. Mead, "Analog VLSI and Neural Systems", *Book*, 1989.

[155] F. Danneville, et al., "Sub-35 pW Axon-Hillock artificial neuron circuit", *Solid-State Electronics*, vol. 153. doi: 10.1016/j.sse.2019.01.002ï

[156] P.M. Ferreira, et al., "Energy efficient fJ/spike LTS e-Neuron using 55-nm node", *32nd Symposium on Integrated Circuits and Systems Design, SBCCI*, Sao Paulo, Brazil, Aug. 2019. doi:10.1145/3338852.3339852

[157] M. Besrour, et al., "Analog Spiking Neuron in 28 nm CMOS," *IEEE Interregional NEWCAS Conference*, Quebec City, QC, Canada, 2022. doi: 10.1109/NEWCAS52662.2022.9842088.

[158] H. Takaloo, A. Ahmadi, and M. Ahmadi, "Design and Analysis of the Morris–Lecar Spiking Neuron in Efficient Analog Implementation", *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 70, n. 1, 2023. doi: 10.1109/TCSII.2022.3203929

[159] S. Kumar, et al., "Dynamical memristors for higher-complexity neuromorphic computing", *Nature Reviews Materials*, 2022. doi: 10.1038/s41578-022-00434-z

[160] G. M. Matrone, et al., "A modular organic neuromorphic spiking circuit for retina-inspired sensory coding and neurotransmitter-mediated neural pathways", *Nature Communication*, 2024. doi: 10.1038/s41467-024-47226-3

[161] C. Bartolozzi, and G. Indiveri, "Synaptic dynamics in analog VLSI", *Neural Computation*, vol. 19, n. 10, 2007. doi: 10.1162/neco.2007.19.10.2581

[162] N. Qiao, and G. Indiveri, "Analog circuits for mixed-signal neuromorphic computing architectures in 28 nm FD-SOI technology", *IEEE SOI-3D-Subthreshold Microelectronics Technology Unified Conference*, Burlingame, CA, USA, 2017. doi: 10.1109/S3S.2017.8309203

[163] L. Khacef, et al., "Spike-based local synaptic plasticity: a survey of computational models and neuromorphic circuits", *Neuromorphic Computing and Engineering*, vol. 3, n. 4, 2023. doi: 10.1088/2634-4386/ad05da

[164] F. Danneville, et al., "Sub-0.3V CMOS neuromorphic technology and its potential application", *International Conference on Content-Based Multimedia Indexing*, Lille, France, 2021. doi: 10.1109/CBMI50038.2021.9461899

[165] E. F. Morales, and H. J. Escalante, "Chapter 6 - A brief introduction to supervised, unsupervised, and reinforcement learning", *Academic Press*, 2022. doi: 10.1016/B978-0-12-820125-1.00017-8.

[166] SuperVize Me: What's the Difference Between Supervised, Unsupervised, Semi-Supervised and Reinforcement Learning? https://blogs.nvidia.com/blog/supervised-unsupervised-learning/

[167] Q. Liu, and Y. Wu, "Supervised Learning", *Encyclopedia of the Sciences of Learning, Springer*, 2012. doi: 10.1007/978-1-4419-1428-6_451

[168] H. B. Barlow, "Unsupervised learning", *Neural computation*, vol. 1, n. 3, 1989. doi: 10.1162/neco.1989.1.3.295

[169] O. Chapelle, B. Scholkopf, and A. Zien, "Semi-Supervised Learning", *IEEE Transactions on Neural Networks*, vol. 20, n. 3, 2009. doi: 10.1109/TNN.2009.2015974.

[170] X. Liu, et al., "Self-Supervised Learning: Generative or Contrastive", *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, n. 1, 2023. doi: 10.1109/TKDE.2021.3090866

[171] A. K. Shakya, et al., "Reinforcement learning algorithms: A brief survey", *Expert Systems with Applications*, vol. 231, 2023. doi: 10.1016/j.eswa.2023.120495.

[172] T. P. Lillicrap, et al., "Backpropagation and the brain", *Nature Reviews Neuroscience*, vol. 21, 2020. doi: 10.1038/s41583-020-0277-3

[173] B. A. Richards, et al., "A deep learning framework for neuroscience", *Nature Neuroscience*, vol. 22, 2029. doi: 10.1038/s41593-019-0520-2

[174] J.C.R. Whittington, and R. Bogacz, "Theories of Error Back-Propagation in the Brain", *Trends in Cognitive Sciences*, vol. 23, n. 3, 2019. doi: 10.1016/j.tics.2018.12.005

[175] Q. Yu, H. Li, and K. C. Tan, "Spike Timing or Rate? Neurons Learn to Make Decisions for Both Through Threshold-Driven Plasticity", *IEEE Transactions on Cybernetics*, vol. 49, n. 6, 2019. doi: 10.1109/TCYB.2018.2821692

[176] D. O. Hebb, "The organization of behavior; a neuropsychological theory", *Wiley*, 1949.

[177] D. E. Feldman, "The spike-timing dependence of plasticity", *Neuron*, vol. 745, n. 4, 2012. doi: 10.1016/j.neuron.2012.08.001

[178] Y. Andrade-Talavera, et al., "Timing to be precise? An overview of spike timing-dependent plasticity, brain rhythmicity, and glial cells interplay within neuronal circuits", *Molecular Psychiatry*, vol. 28, 2023. doi: 10.1038/s41380-023-02027-w

[179] B. Scellier, and Y. Bengio, "Equilibrium propagation: Bridging the gap between energy-based models and backpropagation", *Frontiers in computational neuroscience*, vol. 11, n. 24, 2017. doi: 10.3389/fncom.2017.00024

[180] J.P. Pfister, and W. Gerstner, "Triplets of spikes in a model of spike timing-dependenr plasticity", *Journal of Neuroscience*, vol. 26, n. 38, 2006. doi: 10.1523/JNEUROSCI.1425-06.2006

[181] E. Martin, et al., "Eqspike: spike-driven equilibrium propagation for neuromorphic implementations", *iScience*, vol. 24, n. 3, 2021. doi: 10.1016/j.isci.2021.102222

[182] M. Stimberg, R. Brette, and D.F Goodman, "Brian 2, an intuitive and efficient neural simulator", *eLife*, vol. 8, 2019. doi: 10.7554/eLife.47314.001

[183] Brian2: https://briansimulator.org/

[184] NEST: https://neuralensemble.org/NEST/

[185] S. J. van Albada, et al., "Performance Comparison of the Digital Neuromorphic Hardware SpiNNaker and the Neural Network Simulation Software NEST for a Full-Scale Cortical Microcircuit Model", *Frontiers in Neuroscience*, vol. 12, 2018. doi: 10.3389/fnins.2018.00291

[186] SNNTORCH Documentation. https://snntorch.readthedocs.io/en/latest/

[187] G. Indiveri, "A neuromorphic VLSI device for implementing 2D selective attention systems", *IEEE Transactions on Neural Networks*, vol. 12, n. 6, 2001, doi: 10.1109/72.963780

[188] B. Heruseto, et al., "Embedded Analog CMOS Neural Network inside high speed camera", *Asia Symposium on Quality Electronic Design*, Kuala Lumpur, Malaysia, 2009. doi: 10.1109/ASQED.2009.5206280.

[189] M. Giulioni, et al., "Real time unsupervised learning of visual stimuli in neuromorphic VLSI systems", *Scientific Reports* vol. 5, 2015. doi: 10.1038/srep14730

[190] A. van Schaik, and S.C.Chii Liu, "AER EAR: a matched silicon cochlea pair with address event representation interface", *IEEE International Symposium on Circuits and Systems*, Kobe, Japan, 2005. doi: 10.1109/ISCAS.2005.1465560

[191] P. M. Ferreira, et al., "Neuromorphic analog spiking-modulator for audio signal processing", *Analog Integrated Circuits and Signal Processing*,vol. 106, n. 1, 2021. doi: 10.1007/s10470-020-01729-3

[192] F. Perez-Pena, A. Linares-Barranco, and E. Chicca, "An approach to motor control for spike-based neuromorphic robotics", *IEEE Biomedical Circuits and Systems Conference*, Lausanne, Switzerland, 2014. doi: 10.1109/BioCAS.2014.6981779

[193] M. Guilioni, et al., "Event-Based Computation of Motion Flow on a Neuromorphic Analog Neural Platform", *Frontiers in Neuroscience*, vol. 10, 2016. doi: 10.3389/fnins.2016.00035

[194] E. Donati, and G. Indiver, "Neuromorphic bioelectronic medicine for nervous system interfaces: from neural computational primitives to medical applications", *Progress in Biomedical Engineering*, vol. 5, n. 1, 2023. doi: 10.1088/2516-1091/acb51c

[195] E. Donati, et al., "Discrimination of EMG Signals Using a Neuromorphic Implementation of a Spiking Neural Network", *IEEE Transactions on Biomedical Circuits and Systems*, vol. 13, n. 5, 2019. doi: 10.1109/TBCAS.2019.2925454

[196] Q. Sun, et al., "Implementation Study of an Analog Spiking Neural Network for Assisting Cardiac Delay Prediction in a Cardiac Resynchronization Therapy Device", *IEEE Transactions on Neural Networks*, vol. 22, n. 6, 2011. doi: 10.1109/TNN.2011.2125986

[197] E. Donati, and G. Valle, "Neuromorphic hardware for somatosensory neuroprostheses", *Nature Communications*, vol. 15, 2024. doi: 10.1038/s41467-024-44723-3

[198] P. Diehl, and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity", *Frontiers in Computational Neuroscience*, vol. 9, 2015. doi: 10.3389/fncom.2015.00099

[199] S. Wang, P. Maris Ferreira, and A. Benlarbi-Delai, "Behavioral Modeling of Nonlinear Power Amplifiers Using Spiking Neural Networks", *20th IEEE Interregional NEWCAS Conference*, Quebec City, QC, Canada, 2022. doi: 10.1109/NEWCAS52662.2022.9842167

[200] W. Guo, et al., "An End-To-End Neuromorphic Radio Classification System With an Efficient Sigma-Delta-Based Spike Encoding Scheme", *IEEE Transactions on Artificial Intelligence*, vol. 5, n. 4, 2024. doi: 10.1109/TAI.2023.3306334

[201] J. Chen, N. Skatchkovsky, and O. Simeone, "Neuromorphic Wireless Cognition: Event-Driven Semantic Communications for Remote Inference", *IEEE Transactions on Cognitive Communications and Networking*, vol. 9, n. 2, 2023. doi: 10.1109/TCCN.2023.3236940

[202] H. G. Stratigopoulos, T. Spyrou, and S. Raptis, "Testing and Reliability of Spiking Neural Networks: A Review of the State-of-the-Art", *EEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems*, Juan-Les-Pins, France, 2023. doi: 10.1109/DFT59622.2023.10313541

[203] H. Dai, et al., "Multi-layer neural network for received signal strength-based indoor localisation", *IET Communications*, vol. 10, n. 6, 2016. doi: 10.1049/iet-com.2015.0469

[204] X. Song, et al., "A Novel Convolutional Neural Network Based Indoor Localization Framework with WiFi Fingerprinting", *IEEE Access*, vol. 7, 2019. doi: 10.1109/ACCESS.2019.2933921

[205] A. Kucuk, et al., "Real-Time Convolutional Neural Network-Based Speech Source Localization on Smartphone", *IEEE Access*, vol. 7, 2019. doi: 10.1109/ACCESS.2019.2955049

[206] J. Ko, H. Kim, and J. Kim, "Real-Time Sound Source Localization for Low-Power IoT Devices Based on Multi-Stream CNN", *Sensors*, vol. 22, n. 12, 2022. doi: 10.3390/s22124650

[207] S. Ivanov, V. Kuptsov, V. Badenko, and A. Fedotov, "RSS/TDoA Based Source Localization in Microwave UWB Sensors Networks Using Two Anchor Nodes", *Sensors*, vol. 22, n. 8, 2022. doi: 10.3390/s22083018

[208] B. Xu, X. Zhu, and H. Zhu, "An efficient indoor localization method based on the long short-term memory recurrent neuron network", *IEEE Access*, vol. 7, 2019. doi: 10.1109/ACCESS.2019.2937831

[209] F. Mahdavi, H. Zayyani, and R. Rajabi, "RSS Localization Using an Optimized Fusion of Two Deep Neural Networks", *IEEE Sensors Letters*, vol. 5, n. 12, 2021. doi: 10.1109/LSENS.2021.3125911

[210] F. Moro, et al., "Neuromorphic object localization using resistive memories and ultrasonic transducers", *Nature Communications*, vol. 13, n. 1, 2022. doi: 10.1038/s41467-022-31157-y

[211] A. Ross, et al., "Multilayer spintronic neural networks with radiofrequency connections", *Nature Nanotechnology*, vol. 18, n. 11, 2023. doi: 10.1038/s41565-023-01452-w

[212] N. Leroux, et al., "Classification of multi-frequency RF signals by extreme learning, using magnetic tunnel junctions as neurons and synapses", *APL Machine Learning*, vol. 1, n. 3, 2023. doi: doi.org/10.1063/5.0155447

[213] N. Nikolova, et al., "Chapter Six - Substrate-Integrated Antennas on Silicon", *Advances in Imaging and Electron Physics, Elsevier*, vol. 174, 2012. doi: 10.1016/B978-0-12-394298-2.00006-5

[214] G. Cybenkot, et al., "Approximation by Superpositions of a Sigmoidal Function*", *Mathematics of Control, Signals, and Systems*, vol. 2, 1989.

[215] X. Glorot, A. Bordes, and Y. Bengio, "Deep Sparse Rectifier Neural Networks", *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

[216] F. J. Richards, "A Flexible Growth Function for Empirical Use", *Journal of Experimental Botany*, vol. 10, n. 2, 1959. doi: 10.1093/jxb/10.2.290

[217] Z. Li, et al., "A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects", *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, n. 12, 2022. doi: 10.1109/TNNLS.2021.3084827

[218] T. Grasser, "Noise in Nanoscale Semiconductor Devices", *Springer Nature*, 2020.

[219] Random walk, Wikipédia. https://en.wikipedia.org/wiki/Random_walk

[220] A. A. Abidi, "Phase Noise and Jitter in CMOS Ring Oscillators", *IEEE Journal of Solid-State Circuits*, vol. 41, n. 8, 2006. doi: 10.1109/JSSC.2006.876206

[221] K-W. Cheng, and S-E. Chen, "An Ultralow-Power OOK/BFSK/DBPSK Wake-Up Receiver Based on Injection-Locked Oscillator", *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, n. 7, 2021. doi: 10.1109/TVLSI.2021.3073166

[222] L. Reyes, and F. Silveira, "Gain, Signal-to-Noise Ratio and Power Optimization of Envelope Detector for Ultra-Low-Power Wake-Up Receiver, *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 66, n. 10, 2019. doi: 10.1109/TCSII.2019.2932767

[223] V. Mangal, and P. R. Kinget, "Sub-nW Wake-Up Receivers With Gate-Biased Self-Mixers and Time-Encoded Signal Processing", *IEEE Journal of Solid-State Circuits*, vol. 54, n. 12, 2019. doi: 10.1109/JSSC.2019.2941010

[224] P-H. Wang, et al., "A Near-Zero-Power Wake-Up Receiver Achieving -69 dBm Sensitivity", *IEEE Journal of Solid-State Circuits*, vol. 53, n.6, 2018. doi: 10.1109/JSSC.2018.2815658

[225] K-W. Cheng, and S-E. Chen, "An Ultralow-Power Wake-Up Receiver Based on Direct Active RF Detection", *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 64, n. 7, 2017. doi: 10.1109/TCSI.2017.2664919

[226] M. A. Karami, and K. Moez, "An Integrated RF-Powered Wake-Up Wireless Transceiver with -26 dBm Sensitivity", *IEEE Internet of Things Journal*, vol. 4662, n. c, 2021. doi: 10.1109/JIOT.2021.3116208

[227] R.H. Walden, "Analog-to-Digital Converter Survey and Analysis", *IEEE Journal on Selected Areas in Communications*, vol. 17, n. 4, 1999

[228] P. Chevalier, et al., "A 55 nm triple gate oxide 9 metal layers SiGe BiCMOS technology featuring 320 GHz fT / 370 GHz fMAX HBT and high-Q millimeter-wave passives," *Int. Electron Devices MeetingI (EDM)*, San Francisco, CA, USA, Dec 2014. doi: 10.1109/IEDM.2014.7046978

[229] M. Siniscalchi, et al., "Modeling a nanometer FD-SOI transistor with a basic all-region MOSFET model", *IEEE Latin America Electron Devices Conference*, San Jose, Costa Rica, 2020. doi: 10.1109/LAEDC49063.2020.9073239

[230] O. F. Siebel, M. C. Schneider, C. Galup-Montoro, "MOSFET threshold voltage: Definition, extraction, and some applications", *Microelectronics Journal*, vol. 43, n. 5, 2012. doi: 10.1016/j.mejo.2012.01.004

[231] A. A. Faisal, L. P. Selen, and D. M. Wolpert, "Noise in the nervous system", *Nat Rev Neurosciences*, vol. 9, n. 4, 2008. doi: 10.1038/nrn2258

[232] S. Park, D. Lee and S. Yoon, "Noise-Robust Deep Spiking Neural Networks with Temporal Information", *58th ACM/IEEE Design Automation Conference (DAC)*, San Francisco, CA, USA, 2021. doi: 10.1109/DAC18074.2021.9586195

[233] G. Ma, R. Yan, and H. Tang, "Exploiting Noise as a Resource for Computation and Learning in Spiking Neural Networks", *Cell Press*, vol. 4, n. 10, 2023. doi: 10.1016/j.patter.2023.100831

[234] D. Querlioz and V. Trauchessec, "Stochastic resonance in an analog current-mode neuromorphic circuit", *IEEE International Symposium on Circuits and Systems (ISCAS)*, Beijing, China, 2013. doi: 10.1109/ISCAS.2013.6572166