FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Efficient Neuromorphic Architectures for Visual Perception

Marcelo Almeida de Carvalho

Mestrado em Bioengenharia

Supervisor: Jaime dos Santos Cardoso, PhD

Co-Supervisor: João Diogo Fernandes Freitas Nunes, MSc

July 29, 2022

Efficient Neuromorphic Architectures for Visual Perception

Marcelo Almeida de Carvalho

Mestrado em Bioengenharia

Approved in oral examination by the committee:

Chair: Prof. Miguel Fernando Paiva Velhote Correia, PhD
External Examiner: Timothée Masquelier, PhD
Supervisor: Prof. Jaime dos Santos Cardoso, PhD

July 29, 2022

Abstract

Autonomous Driving Systems (ADS) require some additional concerns regarding safety, regulatory, and hardware specifications. The perception module, the first in the ADS' pipeline, is frequently pointed as the most important since it requires real time reliable responses to act accordingly to the surrounding environment. This module highly relies on machine learning, benefiting the most with precise, yet simple and efficient computer vision algorithms. A promising paradigm in efficient machine learning is that of Spiking Neural Networks (SNNs) and although the field is maturing, with both biologically plausible unsupervised learning and supervised learning demonstrating positive results, it is still not clear if the current state-of-the-art is suitable for real-world applications such as ADS.

In this work, we address several SNN algorithms and assess their suitability for ADS' perception modules, supported on Light Detection and Ranging (LiDAR) data. We start by experimenting with several hyperparameters and evaluate the SNN algorithms on common benchmarking datasets. We then propose a LiDAR spike-encoding scheme and demonstrate that SNNs perform on par with conventional Artificial Neural Networks (ANNs) of approximate architecture on this kind of data.

Overall, some SNN algorithms are still not ready for complex real-world applications, but since LiDAR data is potentially sparse, it could benefit from SNNs which, by nature, are suitable for asynchronous and sparse event streams. Nonetheless, LiDAR spike-encoding is still an open research question, and the top-performing SNN algorithm is to be proposed, although supervised learning, supported on backpropagation and surrogate gradients, seems the most promising direction.

Resumo

Sistemas de condução autónoma requerem cuidados adicionais em relação às especificações de segurança, regulação e hardware. O módulo de perceção, o primeiro módulo do pipeline dos sistemas de condução autónoma, é frequentemente apontado como o mais importante, uma vez que exige respostas em tempo real e confiáveis para agir de acordo com o ambiente envolvente. Este módulo está altamente dependente de machine learning, sendo que seria o maior beneficiário de algoritmos de visão por computador precisos, e ao mesmo tempo simples e eficazes. As Spiking Neural Networks (SNNs) apresentam um paradigma promissor em machine learning eficiente e, embora a área ainda esteja em desenvolvimento, com a aprendizagem supervisionada e não supervisionada a apresentar resultados positivos, ainda não é claro se os resultados do estado da arte são apropriados para aplicações no mundo real como nos sistemas autónomos de condução.

Neste trabalho, abordamos vários algoritmos baseados em SNNs e avaliamos se são apropriados para implementação em módulos de perceção de sistemas autónomos de condução, suportados em dados LiDAR. Começamos por experimentar vários hiperparâmetros diferentes e avaliamos os algoritmos em datasets comuns. De seguida propomos um esquema de codificação em spikes para os dados LiDAR e demonstramos que as SNNs obtêm performances semelhantes com as obtidas pelas redes neuronais artificiais tradicionais com arquiteturas semelhantes neste tipo de dados.

No geral, alguns algoritmos de SNN ainda não estão preparados para aplicações complexas no mundo real, mas uma vez que os dados LiDAR são potencialmente esparsos, podem beneficiar com a aplicação de SNNs que, por natureza, são adequadas para fluxos de eventos assíncronos e esparsos. No entanto, a codificação dos dados LiDAR em disparos é ainda um tema de pesquisa aberto, e o algoritmo de SNN com a melhor performance ainda está para ser proposto, embora a aprendizagem supervisionada, suportada na retropropagação e gradientes substitutos, parece ser a direção mais promissora.

Agradecimentos

Ao Professor Doutor Jaime Cardoso, orientador desta dissertação, pelo acompanhamento e oportunidade de integrar este projeto.
Ao João Nunes, co-orientador desta dissertação, por toda a disponibilidade, ajuda e acompanhamento.
Aos meus pais e Cátia.
A todos os elementos do projeto THEIA.
A todos os que me acompanharam ao longo deste percurso.

Ao *Project THEIA: Automated Perception Driving (POCI-01-0247-FEDER-047264)*, a partir do qual surgiu a oportunidade de desenvolver esta dissertação.

Marcelo Almeida Carvalho

"Tudo é ousado
para quem a nada se atreve."

Fernando Pessoa

Contents

List of Figures

List of Tables

Abbreviations

AD	Autonomous Driving
ADS	Autonomous Driving Systems
ANN	Artificial Neural Network
ATIS	Asynchronous Time-based Image
BP	Backpropagation
CNN	Convolutional Neural Network
DECOLLE	Deep Continuous Local Learning
DL	Deep Learning
DoG	Difference of Gaussian
DVS	Dynamic Vision Sensor
FB	Feedback
FF	Feedforward
IF	Integrate and Fire
IT	Inferior Temporal
LGN	Lateral Geniculate Nucleus
LiDAR	Light Detection and Ranging
LIF	Leaky Integrate and Fire
LTD	Long Term Depression
LTP	Long Term Potentiation
ML	Machine Learning
MLP	MultiLayer Perceptron
ODE	Ordinary Differential Equation
RF	Receptive Field
RADAR	Radio Detection and Ranging
SAE	Society of Automotive Engineers
SL	Supervised Learning
SNN	Spiking Neural Network
SOM	Self Organizing Map
SOTA	State-of-the-art
STDP	Spike Timing Dependent Plasticity

UL Unsupervised Learning

V1 Primary Visual Area

Chapter 1

Introduction

1.1 Context

According to the World Health Organization's state of the road safety report in 2018 [1], the number of road traffic deaths was around 1.35 million worldwide, which corresponds to almost 3 deaths per minute. Estimates also show that 20 to 50 million people suffer non-fatal injuries that may cause long-term disabilities. Although some accidents may be owed to weather conditions (slippery roads from rain, or snow), the road condition or even to some defects in the vehicle, human error is pointed to be one of the main factors contributing to this number of accidents, being 5 to 35% of all road deaths alcohol related. In fact, several studies [2, 3, 4] point out the importance of the drivers' emotional state in the prevention of such occurrences, since it may change the way people perceive the situations or even act upon them.

To overcome these problems, the automotive industry has been introducing safer vehicle models through the implementation of drive-assistance technology such as lane keeping assistance and self-braking systems. The ultimate goal is to develop fully Autonomous Driving Systems (ADS), which provide a safer alternative to traditional cars, allowing to decrease the amount of accidents (and, consequently, road traffic deaths) through emotionless software-based decisions. Besides increased safety, ADS would also be responsible for providing environmental and comfort advantages. In that sense, these systems could provide improvements in mobility for people who cannot drive. Furthermore, ADS would be generally faster to get to a destination, since they would be able to constantly adapt the trajectory in order to avoid traffic jams, contributing to a decreased stress in each travel. This faster alternative to get to a destination would also allow cars to use less fuel and, consequently, decrease pollution levels, positively impacting air's quality.

The International Society of Automotive Engineers (SAE) defined 6 automation levels, ranging from 0 to 5, where the level 0 corresponds to vehicles with no automation and level 5 to fully automated vehicles. A schematic representation of the automation levels is illustrated in figure 1.1.

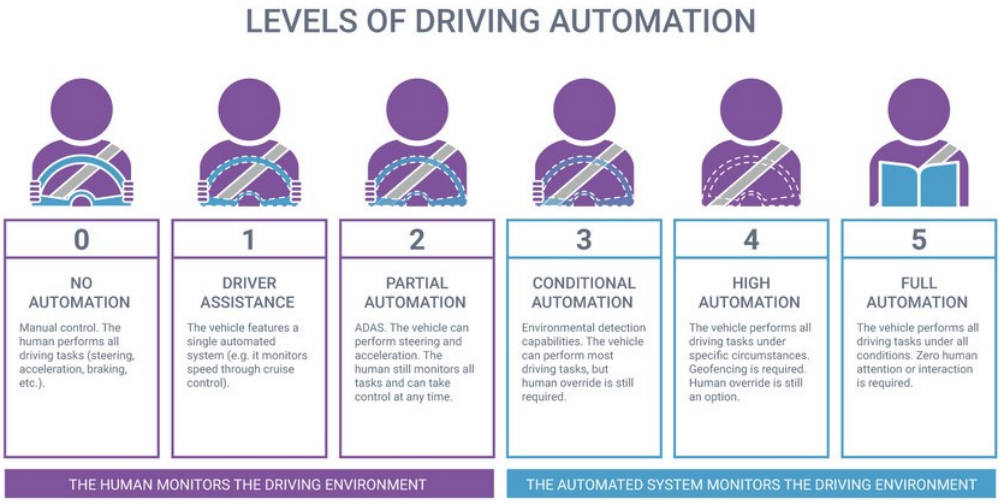

Figure 1.1: The six levels of driving automation for on-road vehicles. Source:[5]

It is considered to be level 0 all vehicles in which the driver is fully responsible for the acceleration and steering even if there are some safety features such as emergency braking in case of an imminent collision. In level 1, although the driver is still the main responsible for the driving experience, the vehicle already has some higher degree of automation such as lane keeping or adaptive cruise control systems. Level 2 is characterized by a partial automation in which the driver must be always ready to step in if necessary but, in general, the vehicle is able to control its own acceleration and steering. The next level is the one in which the driver can disengage from the act of driving, although only in certain conditions and its presence behind the wheel is still mandatory. In levels 4 and 5, the vehicle is capable of driving on its own, being that in level 5 the human driver becomes a mere passenger in the vehicle.

Several automotive companies have been exploring increasingly autonomous solutions for both commercial and transportation purposes. Some examples of such companies are Tesla, Waymo, Motional, Refraction AI, Voyage, nuTonomy and Cruise. For many years, Tesla was the leading company in the autonomous vehicle industry. Tesla is a fully electric vehicle manufacturer with a well known and developed autopilot system that focuses on assisting the driver in the most burdensome tasks such as driving long distances in highways and other similar roads. Their system is equipped with emergency braking, collision alerts, blind spot monitoring and adaptive cruise control even though the driver oversees the steering task [6] , making it a level 2 autonomous system.

Waymo is another major player in the autonomous vehicle industry that started as a Google's division of self-driving exploration in 2009 and was established as an independent company in 2016. Waymo is currently focused on two main projects: Waymo One, an autonomous ride-hailing service that is established in the Phoenix metropolitan area and Waymo Via, which aims to develop autonomous solutions for goods delivery, mostly targeting large trucks [7]. Although Waymo solutions fall under SAE definition of level 4 automated systems, currently there are no solutions

with a level of automation higher than 3 commercially available in the market for personal use. This is due to the fact that is much more difficult to develop a vehicle that must be prepared for all kinds of conditions than one that must be prepared for the conditions of specific areas such as the solutions created by Waymo.

ADS are composed of several interconnected modules that allow the system to make decisions, from perception to control. The perception module is responsible for understanding and interpreting the surrounding environment, making it of utmost importance to the system. This module must, thus, present high accuracy and reliability since all other modules are dependent on its output. To aid in the perception task, ADS are normally equipped with data acquisition sensors and devices such as RGB stereo cameras, Radio Detection and Ranging (RADAR), and ultrasound sensors. Recently, some autonomous vehicles began to also include Light Detection and Ranging (LiDAR) sensors, which provide 3D point cloud information of the surroundings, being used mainly in object avoidance and mapping. Nevertheless, the sensors that compose the ADS depend on the developing company and, while some manufacturers find LiDAR to be crucial to a better autonomous driving performance, others opt for a more traditional vision-based approach. On the one hand, cameras are cheap high-resolution sensors that allow us to distinguish shapes, colors and recognize 2D information, which is essential to reading lanes and pavement markings. Their similarities with the human eye are ideal to make the autonomous driving experience as close as possible to the one produced by a human driver. Despite these advantages, the similarity with the human eye also constitutes a major disadvantage under severe weather conditions and dark environments (i.e., at night). On the other hand, LiDAR sensors allow a 3D reconstruction of the environment and provide a 360-degree visibility independent of the weather conditions but with lower resolution than traditional cameras. Besides that, some companies might opt to not include LiDAR sensors due to their high cost and sophistication, meaning an increase in the computational power needed to process the hundreds of thousands of points obtained by the sensor. RADAR sensors are the middle term between the two aforementioned sensors, with lowest resolution among them but cheaper than the LiDAR and still accurate under adversarial weather conditions. This is still the default sensor for emergency braking in autonomous vehicles.

The perception module of an autonomous vehicle is responsible for the monitorization of the surrounding environment. Although that monitorization may be composed by segmentation, classification, and recognition, this work focuses on the classification task. Nowadays, Convolutional Neural Networks (CNNs) present State-of-the-art (SOTA) results for perception tasks [8] but, even though they try to approximate the human brain behavior through mathematical models, the similarities are mostly abstract, since they usually use non-linear but continuous activation functions as neuron approximators, while biological neurons carry information through non-differentiable, discontinuous, spike-based signals. Spiking Neural Networks (SNNs) are pointed as the successor of conventional neural networks, that came to close the gap between artificial intelligence and human brain's behavior, taking advantage of event-driven computations and great latency responses [9].

1.2 Motivation

The perception module of ADS is of utmost importance as failures would propagate through the whole pipeline, with possible fatal consequences. Thus, a fast perception's response is crucial to allow the system to act in accordance with the highly dynamic and ever-changing surrounding environment. Nowadays, perception in autonomous driving is highly reliable on deep CNN algorithms to detect and classify objects, even though these models require high computational power. Moreover, hardware constraints and the need for low latency during inference, demands simple and efficient models, that are often compressed. This means there is some loss of performance when tackling efficiency. But faster inference and lower computational power are thus desirable features of ADS. A promising direction is that of SNNs, a more biologically plausible alternative to Artificial Neural Networks (ANNs). SNNs process information asynchronously and in the form of spikes thus being more efficient and showing that event-based tasks, such as is the case of many ADS, could benefit from these kind of networks. Nonetheless, it is still not clear if current SNN algorithms are suitable for ADS and, to the best of our knowledge, there is no standard spike-encoding strategy for common ADS' data, such as LiDAR point clouds. Moreover, some SNN algorithms are highly susceptible to the choice of hyperparameters and some works do not detail the set of hyperparameters they use which hinders reproducibility. Overall, although the field is evolving, SNNs' research is still in its infancy and it remains to clarify which real-world tasks would benefit the most from neuromorphic computing and, subsequently, biologically plausible SNN algorithms.

1.3 Goals

This thesis thus aims to assess the performance of SOTA SNN algorithms on ADS data, namely LiDAR and Dynamic Vision Sensor (DVS) data. We start by studying different biologically plausible mechanisms through the replication of several SOTA algorithms. We evaluate these models on common computer vision benchmarks. We intend to experiment with several hyperparameters and provide source code to ensure the reproducibility of our methodology. We also aim to propose a LiDAR spike-encoding scheme and compare the performance of SNNs with conventional ANNs on this type of data. The goal would be to assess if SOTA SNN solutions are ready for real-world scenarios, in particular for autonomous driving applications. These contributions with point clouds can also be leveraged in other areas where 3D information and accurate predictions are crucial, such as medical applications.

1.4 Document structure

In this first chapter, the context of this thesis is presented, followed by the motivation of the study of SNNs (1.2) and why they are important in the context presented before. The main goals of this work are presented in section 1.3.

In chapter 2, a brief background on the field of Deep Learning (DL) is presented to illustrate the evolution of this field and the motivation for the appearance of biological plausible networks. In section 2.2, the fundamentals of SNNs are discussed. Some relevant biological considerations are also analyzed in the following sections.

Chapter 3 presents current neuromorphic solutions regarding sensors and datasets that can be beneficial to the application of SNNs while in chapter 4 point clouds are discussed, targeting some relevant datasets and developed works for LiDAR data classification.

Chapters 5 and 6 propose and discuss the developed work, presenting details about the implementations and obtained results.

Finally, chapter 7 summarizes the most important considerations developed through the course of this thesis and proposes future steps for further development of the work.

Chapter 2

Background

This chapter is based on our work published in IEEE Access through the course of this thesis. For a more detailed and comprehensive overview, we refer the reader to [10].

2.1 Introduction

DL is a subfield of Machine Learning (ML) highly inspired by the capabilities of the human brain. DL algorithms usually learn from experience to perform human-like tasks such as perception, which includes object segmentation and classification. The history of DL began in 1943, when Walter Pitts and Warren McCulloch presented the first mathematical model describing an ANN [11]. But it was not until 1960 that the first Backpropagation (BP) model was proposed in the context of Control Theory by Henry J. Kelley [12], laying the foundations for further refinements that granted Yann LeCun the first practical demonstration of this method in 1989 onto reading handwritten digits. In that year, CNNs had already been established as the promising artificial intelligence solution for pattern recognition. Significant advances in hardware, with the introduction of GPU-enabled DL, led to bigger, more complex, and robust algorithms, increasing their processing speed and efficiency. In 2012, DL reached the next level with the ImageNet challenge, where AlexNet [13] achieved a 15.3% top-5 error rate (against the 26.2% top-5 error rate of the second place) in the 1000 classes classification task of the ImageNet [14] dataset. As a result, deeper and more complex networks have been studied and, nowadays, the SOTA top-5 error rate in the ImageNet classification problem decreased to less than 1% [15]. However, the increase in performance has been correlated with an increase in required resources (e.g. processing power and energy requirements). In fact, the improvement of 14.5% in top-5 error rate in the ImageNet dataset classification from the AlexNet to the current SOTA model, is only achievable with an increase of 833M parameters. With this ever-increasing number of parameters and models' complexity, DL may become unsustainable [16]. To that end, efficient DL algorithms are a growing

concern with SNNs being one of the most promising paradigms. Moreover, due to their intrinsic properties (e.g. sparseness of spike trains, event-drive, etc.), SNNs are naturally suitable for many systems, including ADS. Nonetheless, there are a few challenges with several SNN algorithms, like the non-scalable Spike Timing Dependent Plasticity (STDP) or the non-differentiable activation functions.

2.2 Fundamentals of Spiking Neural Networks

SNNs, commonly referred as the third generation of neural networks, came as an energy-efficient alternative to conventional ANN, namely due to their lower computational costs that derive from asynchronous computations and biological plausibility. SNN architectures consist of interconnected neurons by synapses that allow the flow of information from presynaptic to postsynaptic neurons. Unlike conventional ANNs, however, information is transmitted in the form of discrete spikes over time. In fact, SNNs' activation functions are time dependent.

As in biological neural networks, every time a neuron receives an impulse, its membrane potential (v_{mem}) is increased by a certain amount δv. When v_{mem} gets to the neuron's threshold value (v_{thresh}), the neuron emits a spike. As a result, the neuron's membrane potential will reset to the resting potential (v_{rest}) and the neuron will go through a refractory period in which it cannot produce another impulse. The emitted signal will propagate and reach subsequent neurons as a synaptic current proportional to the synaptic weight or conductance. Despite their similarities, artificial neurons in SNNs are fundamentally different from their biological counterparts as they rely on mathematical approximations. In 1952, Hodgkin and Huxley [17] proposed the first mathematical model describing axons' electrical activity, by treating the axon as a simple electrical circuit (figure 2.1). This study, using a squid giant nerve fiber, won them the 1963 Nobel prize in Physiology and Medicine and, despite being the most biologically plausible neuron model, it is computationally complex as it is composed by a non-linear system of four Ordinary Differential Equations (ODEs) [18]. After that, other models appeared in the attempt to describe the axons' current flow such as Izhikevich [19], CSRM and SRM0 [20] neuron models. Nevertheless, the most widely employed neuron model is the Leaky Integrate and Fire (LIF) as it provides a simple and yet complete description of neurons' behavior. Similarly to the Hodgkin-Huxley model, a LIF neuron can be described as a parallel combination of a "leaky" resistor and a capacitor [21] subject to a synaptic current as illustrated in figure 2.2. The membrane potential (v_{mem}) can thus be mathematically defined as

$$C \cdot \frac{dv_{mem}}{dt} = -g_L \cdot (v_{mem}(t) - E_L) + I(t) \tag{2.1}$$

and depends on the capacitor value C, the resistor's conductance (g_L), the resting potential (E_L), and the current source ($I(t)$). Regarding the circuit, it is known that $g_L = \frac{1}{R}$ and $\tau_m = RC$, with τ_m being the membrane time constant. Multiplying both sides of equation 2.1 by R we get the final LIF model expression, represented in equation 2.2 .

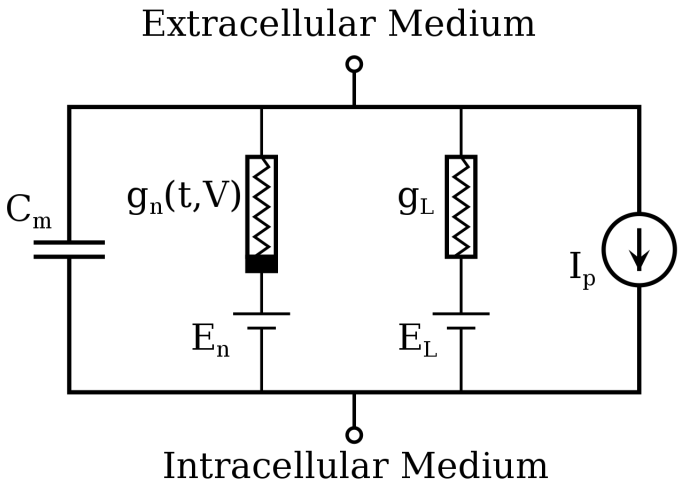

Figure 2.1: Equivalent circuit for the Hodgkin-Huxley neuron model. Capacitor C_m is used to emulate the lipid bilayer while leak ion channels are represented by non-linear (g_n) and linear (g_L) conductances. Source: [22]

$$\tau_m \cdot \frac{dv_{mem}}{dt} = -(v_{mem}(t) - E_L) + RI(t) \tag{2.2}$$

Due to the leaky behavior of this type of neuron, the membrane potential leans towards its resting potential (E_L). Simpler versions of LIF neuron model can be used for specific applications such as in Delorme et al. [23] that used Integrate and Fire (IF) neurons to assess an alternative to conventional coding schemes. Without the leaking factor, neurons can be defined as

$$C \cdot \frac{dv_{mem}}{dt} = I(t) \tag{2.3}$$

Being the equivalent circuit composed by a single capacitor.

Based on the work of Wei Zhang and David J. Linden [24] on neuron intrinsic excitability, Diehl and Cook [25] introduced another variant of the LIF neuron model with an adaptive membrane threshold that became known as Adaptive LIF neuron model. In this model, each time a neuron fires, the membrane threshold increases by θ, allowing to control neurons' firing rate and promoting competition among them.

As information is transmitted in the form of spikes, all these neuron models have a binary activation function $A(t)$ described by equation 2.4.

$$A(t) = \begin{cases} 0, & \text{if } v_{mem} < v_{thresh} \\ 1, & \text{if } v_{mem} \geq v_{thresh} \end{cases} \tag{2.4}$$

A major drawback with this approach relates to the non-differentiability of this activation function, which means that the BP algorithm cannot be directly applied as in conventional ANNs, and we must rely on biologically plausible Unsupervised Learning (UL), such as STDP, or mathematical approximations as later discussed.

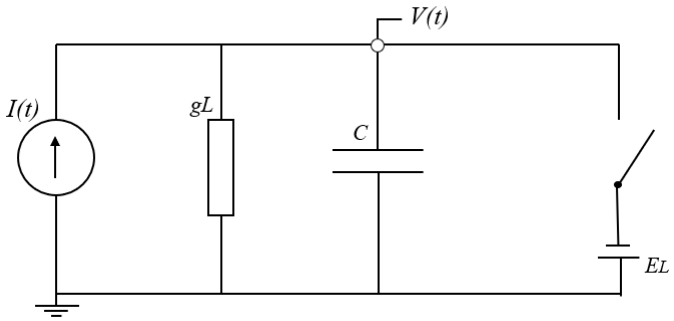

Figure 2.2: Equivalent electrical circuit of the LIF neuron model. The resistor acts as the synaptic weights that allow the current to pass and charge the capacitor. When the capacitor reaches a certain threshold value, it will discharge, similarly to the neuron emitting a spike. Source: [10]

2.3 Biological Neural models

The first step to develop biological plausible ANNs is the understanding of humans' visual system that comprises the sensorial organ (the eye) and parts of the central nervous system that are responsible for interpreting information in the acquired images.

As we already know, human brain responds to electrochemical signals (i.e., spikes). Thus, the light that enters through the eyes' set of lenses must be converted into electrical spikes. That conversion takes place in the retina, a three-layered structure responsible for the interface between the eyes and the human brain. The outermost retinal layer is composed by millions of photoreceptors – rods and cones – that act as transducers, converting light into electrical signals [26]. Photoreceptors communicate with ON-center and OFF-center bipolar cells, located on the middle retinal layer, through the release of glutamate, a neurotransmitter. In the presence of light, photoreceptors remain hyperpolarized and decrease their release of glutamate, stimulating the ON-center and inhibiting OFF-center cells. Bipolar cells present a surrounding with inverse properties from the center, which means that ON-center cells will possess an OFF surrounding and OFF-center cells present an ON surrounding [27], allowing the cell to efficiently encode contrasts in luminance, rather than absolute intensities. Bipolar cells will connect to inner retinal surface cells, ganglion cells, that are responsible for more subtle feature selectivity such as color, size, direction, and even speed of motion [28]. Those cells, in turn, will transmit the information to the brain through complex circuits that, ultimately, will conduct it to the Lateral Geniculate Nucleus (LGN) and visual cortex for further processing and interpretation.

The brain circuits that propagate the information from retinal ganglion cells to the LGN can be decomposed in several simpler circuits and mechanisms that allow the information to flow through the network (Feedforward (FF) excitation circuits), regulate neurons' outputs (FF and Feedback (FB) inhibitory circuits), and differentiate neurons' Receptive Fields (RFs) (lateral inhibition) [29]. When the information reaches the LGN, it will be processed and further transmitted to the Primary Visual Area (V1), which is the first stage of cortical processing and presents itself in a retinotopic

organization to ensure that its neurons' RFs completely represent the visual scene. Moreover, it is known that even the blind spots of the retina are mapped into V1 [30]. The output of the primary visual cortex neurons will stimulate extrastriate areas, V2 and V4, and the information is then passed to the Inferior Temporal (IT) cortex [31], where the neurons' RFs reach their complexity peak. In fact, it is believed that mammalians' brain presents larger and more complex RFs as we move from the retina, where color or size are encoded, to the primary visual cortex, where the neurons present selectivity for orientations and spatial frequencies, and to the IT cortex, where RFs correspond to highly complex features such as faces or other complex objects [32] .

Most of the descripted biological considerations were already addressed in the literature regarding biological plausible neural networks, such as the use of ON-OFF Difference of Gaussian (DoG) filters to simulate the way retinal bipolar cells respond to light stimuli [33, 34, 35]. On the other hand, DVS data is able to mimic retina outputs, allowing researchers to focus on higher processing aspects of the human brain, such as the implementation of lateral inhibition for promoting competition among neurons and increase their RFs differences [36, 25, 37]. Furthermore, 2D Gabor functions' representations [38, 39] are frequently used to simulate V1 neurons' RFs, due to their sensitivity to edges, orientation, and texture, operating at different frequencies and scales [40].

Even though the complexity of our brain is still far from being fully understood by the scientific community, there are already several biological methods that are currently implemented in biologically plausible ANNs that allow to increase their robustness and performance at the same time the efficiency and fast inference are also pursued.

2.4 Information encoding

As aforementioned, the first step in biologically plausible neural networks is to encode information as spike trains. In fact, that is one of the major differences between traditional ANNs and SNNs in image-based tasks, since the previous one uses pixel intensities to extract critical features, while the latter need to map each pixel intensity to a discrete spike domain before feature extraction, for traditional color-based images. Besides, information encoding is a crucial tool to shape models' behavior, considerably contributing for energy-efficient networks that aim to close the gap between ANNs and the human brain functioning. Moreover, the mammalians' brain performs task-specific encoding, which suggests ML practitioners should also take advantage of different encoding methods, depending on the application.

Information encoding techniques can be divided into two critical groups: rate and temporal coding. As the names suggest, broadly speaking, rate coding relies on spike trains' firing rate to encode pixel intensities while temporal coding relies on the precise timing of the spikes to encode information, allowing to improve network's response time.

Rate coding can be further divided into count-, density-, or population rate coding methods as suggested by [41]. For this work, we have focused on count-based coding. More precisely, we resort to Poisson encoding (equation 2.6), where the average firing rate is proportional to the

pixel intensity, and to a probability-based strategy in which the normalized pixel intensity represented the probability of a neuron emitting a spike at each timestep, t. In this coding method, the mean firing rate (ν) is computed over a prespecified time window (T) and can be mathematically described through equation 2.5:

$$\nu = \frac{N_{spike}}{T} \tag{2.5}$$

With N_{spike} being the number of spikes present in the time window T. This is the most widely used coding method as a result of its simplicity and early discovery in biological systems. In fact, Adrian and Zotterman [42], in 1926, performed the first known experience to prove the existence of such encoding method measuring the nerve action potentials' response of a frog's muscle subjected to different weights with a capillary electrometer. In that study, the authors concluded that the muscle's firing rate was proportional to the force exerted by the weights. Roger M. Enoka and Jacques Duchateau[43] also describe the importance of rate coding in human voluntary muscle contraction, especially for intermediate and higher forces and fast contractions. Furthermore, Poisson-like rate coding was also shown to be biologically plausible through medial temporal and primary visual cortex recordings of macaque monkeys [44].

$$spikes = rate^k \cdot \frac{e^{-rate}}{k!} \tag{2.6}$$

with

$$rate = \frac{1}{pixel\ intensity} \cdot \frac{1000}{dt} \tag{2.7}$$

Temporal coding methods have also been proven to exist in the human brain. In fact, although for many years it was thought that the human brain used, essentially, rate coding techniques, some studies found, through human visual system's response time, that the human brain must rely on temporal coding in object recognition tasks. Thorpe et al. [45] prepared an experiment in which subjects were subjected to an image for 20ms and should decide whether there was an animal on the image or not. Through the measurement of event-related potentials, the authors were able to determine that the human brain can perform complex visual tasks in under 150ms, which could only be achievable with temporal coding, since rate coding needs to average over a certain time window. This study was the first step to boost a series of studies of temporal coding methods and, nowadays, several strategies are already used in computer vision tasks for biologically plausible networks. However, as in biological systems, biologically plausible neural networks should also employ different encoding methods, depending on the application. Still, there are some problems with encoding methods' implementation, since in one hand the exact biological methods' functioning is not fully understood while on the other hand it remains to clarify in which situations each strategy is more suitable. Nonetheless, rate-based coding methods remain the most commonly used in SNN algorithms mainly when the information encoding is not the primary focus but rather a different architecture characteristic such as the learning strategy, for example. Temporal coding is also used in several studies when biologically plausibility and fast inference response are desired

even though those studies frequently take advantage of ANN to SNN conversion methods [46, 47], which means that more work to integrate new encoding strategies in SNNs developed from scratch should be developed. Nevertheless, information encoding is crucial to the development of biologically plausible neural networks since information might be lost whilst being propagated through network layers if not properly encoded.

2.5 Learning strategies

Biologically plausible learning strategies, such as STDP, do not perform as well as the widely adopted BP algorithm, which saw its popularity exponentially increase until it became the most widely used method to train DL models [48]. However, BP is not biologically plausible since it consists of a 2-step procedure while learning in the brain occurs in a single step. Furthermore, it requires the use of symmetric forward weights in the FB path, which does not occur in biological systems, the so-called weight transport problem [49]. The non-differentiability of biologically plausible neurons' activation function is another challenge for gradient-based learning. However, several ANN to SNN conversion methods were developed with the goal to take advantage of BP learning algorithm in SNNs. With these conversion approaches, SNN performances are comparable (but still lower) than those presented by SOTA traditional models. This may be due to approximations and assumptions that may deteriorate the converted models' performance [50]. Broadly speaking, learning in SNNs can be categorized into biologically plausible UL and Supervised Learning (SL), where often backpropagation with surrogate gradients is employed.

UL based on STDP is very common due to its biological plausibility. The fact it is not gradient-based is also desirable, considering spiking neuron models. In this learning rule, the synaptic weight change is proportional to the relative timing between pre- and postsynaptic activations [51], as described in equation 2.8.

$$\Delta w = \begin{cases} A e^{\frac{-(|t_{pre}-t_{post}|)}{\tau}}, & \text{if } t_{pre} - t_{post} \leq 0 \\ B e^{\frac{-(|t_{pre}-t_{post}|)}{\tau}}, & \text{if } t_{pre} - t_{post} > 0 \end{cases} \tag{2.8}$$

Where $A > 0$ and $B < 0$ define the learning rates, τ is the timing window constant, t_{pre} and t_{post} are the absolute timings of pre- and postsynaptic spikes, respectively, and Δw is the synaptic weight update. The first branch represents the Long Term Potentiation (LTP), a neural mechanism that describes a persistent increase in synaptic strength if a presynaptic neuron fires briefly before its postsynaptic equivalent, while the second branch refers to Long Term Depression (LTD) that describes the decreasing of synaptic effectiveness of a presynaptic neuron when there is no evident causal relationship between its spiking activity and that of the postsynaptic neuron [52]. Some other STDP variants exist [25, 53, 54], although not fully representing the entire scope of biological learning mechanisms.

One of the major drawbacks with STDP is that it usually limits the models to shallow architectures that would not perform well in real-world tasks [55]. To overcome this challenge, SL

with surrogate gradients has been proposed. Local SL rules are frequently employed due to their biological plausibility, such as in Zhao et al. [56] that proposes a network with a two-step local learning approach. In this paper, global FB connections are used to propagate the prediction error to the hidden layers and the weights are locally updated using STDP. A similar local learning rule is proposed by Kaiser et al. [57], which uses pseudo-targets in each hidden layer, avoiding the weight transport problem, as later discussed. Some papers go further and propose BP approximation algorithms. Zenke and Ganguli [58], for example, proposes a three-factor learning rule that relies on error BP directly from the output to the hidden units, similarly to the methods that use FB alignment. To overcome the problem with non-differentiability that arises in SNN applications, the authors propose a surrogate gradient approach, where they use a fast sigmoid as an approximation to the neurons' activation function for the backward pass, thus allowing to apply BP to multilayered SNNs. In fact, they resort to the partial derivative of the negative half of a fast sigmoid (equation 2.9) to allow the implementation of the BP approach. Furthermore, the authors perform experiments with different methods for the BP of the errors and were able to conclude that the error BP resorting to symmetric weight matrices is beneficial in comparison to the use of random FB weights, even in the case a regularization term was added, arguably showing the advantages of the knowledge of the FF pathway.

$$\sigma^{'} = \frac{1}{(1+|U_i - V_{th}|)^2} \tag{2.9}$$

In turn, Lee et al. [59] proposed an approximate derivative algorithm that accounts for the leaky behavior of LIF neurons. The authors explore a spike-based BP algorithm in which the neurons from the last layer of the network have their threshold set to a high value so that they do not spike. Moreover, the accumulated membrane potential in the last layer is divided by the number of timesteps (T) in order to quantify the output distribution. For the backward pass, a loss function is defined as the sum of squared error over all output neurons (equation 2.10) with the error defined as the difference between the output distribution with the desirable label (equation 2.11).

$$Loss, E = \frac{1}{2}\sum_{j=1}^{n^L} e_j^2 \tag{2.10}$$

$$Output\ error, e_j = output_j - label_j \tag{2.11}$$

To propagate the error to the hidden layers, the activation function for each output neuron is defined as equivalent to the total input current received by the neuron over T timesteps. Regarding the hidden layers, the authors propose a pseudo derivative method that consists in the computation of the estimated derivative of an IF neuron, followed by the addition of an estimate of a leak correctional term. With this strategy they can train deep SNN architectures and achieve 99.59% accuracy in MNIST dataset, 90.09% in the NMNIST, and 90.95% in the CIFAR-10 dataset. Furthermore, authors defend that, resorting to spike-based backpropagation, less computational energy is required than with ANN to SNN conversion methods.

In conclusion, several learning rules have been explored, each of them with its pros and cons and, even though some workarounds were developed, the major problems remain to be the non-differentiability of SNN activation functions, and sparsity of spike trains. Furthermore, from this chapter we were able to understand the gaps that still exist between ANNs and, in particular, SNNs and the human brain and, although several advances have been made in order to close that gap, there are lots of open research questions regarding human brain and the potentialities of SNNs. However, with the ever-increasing search for efficient networks, more and more studies arise in the attempt to reach an optimal solution for each problem. Naturally, neuromorphic hardware is also crucial to take full advantage of such neural networks.

Chapter 3

Neuromorphic data

3.1 Neuromorphic sensors

Neuromorphic computing came to complement the numerous possibilities biologically plausible neural networks offer us. Neuromorphic chips, specifically, have been increasingly studied for their properties that allow computations to be made in an asynchronous and energy-efficient way, which means that, as aforementioned, we can take full advantage of SNNs when applied to this kind of chips. These properties are attractive for every application but particularly desirable for applications where there are constraints in the computation and energy power available such as in ADS. Even though the implementation of SNN-based architectures in neuromorphic chips represents an ultimate goal in SNN development, these networks are still in their infancy, which means that the implementation of highly reliable networks in neuromorphic chips is still far from reality. Nevertheless, in this chapter we present a brief overview of these solutions. Neuromorphic chips can be divided into 3 major design groups: Digital, analog, and mixed-mode (analog/digital) designs. While digital systems are more precise, analog systems present a more biological plausible solution that use native physical properties of electronic devices to perform their computations [60]. Analog hardware solutions, however, suffer from expensive and long design and implementation processes as well as a tendency to be significantly more noisy than digital systems. Nonetheless, neural networks can be robust to noise and faults, which means that they are perfect candidates for analog implementation. Furthermore, unreliability issues brought by analog circuits can, however, be surpassed through the introduction of digital components to analog circuits, originating mixed analog/digital systems. Digital memory is frequently used in analog neuromorphic circuits to store variables' values such as synaptic weights since they are less noisy and more reliable than analog-based memory components. Furthermore, digital communication within the chip and between neuromorphic chips is sometimes employed as well in analog neuromorphic solutions alongside the use of digital components for programmability or learning mechanisms [61].

Even though neuromorphic chips are crucial to take advantage of efficient neural networks architectures, they only represent a way to mimic the mechanisms from which the brain derives and

transmits the information, whilst the codification and the mechanisms through which biological systems obtain the information remains to be discussed. Neuromorphic sensors (i.e., neuromorphic cameras) came to try to mimic the sensing and early visual-processing characteristics of biological systems through biologically inspired analog electronic circuits. Event-based cameras discard redundant visual information as their output is based on visual changes in contrast or motion [62]. With this, neuromorphic sensors present efficient and high-resolution options to display the surrounding environment.

Inivation, in 2008, proposed the DVS128 [63], a high dynamic range event-based camera with 128x128 resolution. Besides its 120 dB of dynamic range, this sensor presents a minimum of 12 μs latency response. The output of the camera is an asynchronous stream of pixel address-events. Each event contains the information of the pixel coordinates where the change was detected, and the respective binary variable known as the polarity of the event. If the polarity is 0, there is an ON event and it was verified an increase of the contrast in the pixel where the event was recorded while there is a decrease in the contrast if the polarity is set to 1, corresponding to an OFF event. Furthermore, each event has a timestamp attached that allows to identify the timing at which the event occurred. In turn, Posch et al. [64] proposed the Asynchronous Time-based Image (ATIS) sensor with 143 dB dynamic range and 304x240 pixel resolution, in many ways similar to the previous mentioned sensor. However, although most neuromorphic sensors are fully asynchronous, in 2014, Inivation proposed another neuromorphic sensor but with the particularity of being frame-based. The DAVIS sensor [65] has higher resolution (240x180), and dynamic range (130 dB) and still manages to need less power than its antecessor, requiring 5-14 mW compared to the 23 mW presented by the DVS128 sensor. The frame-based approach allows simultaneous output of asynchronous events and synchronous frames, making it ideal to employ event-based data into traditional computer-vision algorithms. The creators of this sensor argue that the employment of neuromorphic sensors might be beneficial and highly desirable for applications where power consumption and latency are crucial such as autonomous robots, where the events can be used to track moving features and analyze motion. There are several other approaches regarding the development of neuromorphic sensors and applications of neuromorphic data for visual tasks such as 3D pose estimation through the use of a neuromorphic sensor [66]. They show that, through the combination of both 3D and 2D criteria, the asynchronous high temporal resolution of the event-based camera allows them to achieve real-time accurate pose estimation.

Neuromorphic hardware has been increasingly studied due to its high energy-efficiency and low latency response properties that can leverage the study of biologically plausible neural networks. In fact, although there are some open questions and, currently, the use of neuromorphic computing is still in its infancy, the interface between neuromorphic chips and event-based cameras can leverage the use of SNNs since the information encoding and transmission are addressed by the hardware systems, leaving space to develop in more depth the other aspects of the neural networks. However, as mentioned before, the development of new solutions is constrained by the lack of precise knowledge of the mechanisms that are present in the human brain.

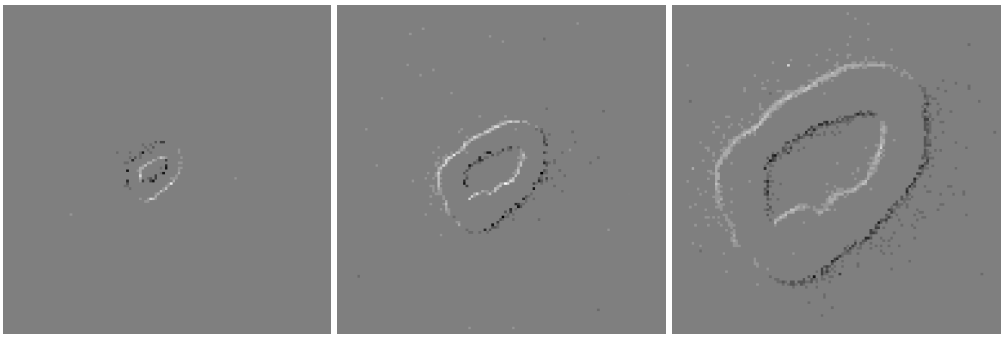

Figure 3.1: Raw data example of a digit from the MNIST-DVS dataset presented in the proposed three different scales

3.2 Relevant datasets

As aforementioned, SNNs expect spike-trains as inputs in order to interpret and propagate information through the network, even though most of the currently used datasets are obtained with frame-based cameras that code information based on pixel intensities. Nonetheless, several studies came to take advantage of neuromorphic sensors to create or adapt common datasets to neuromorphic datasets. One of the most widely used frame-based dataset is MNIST [67] due to its simplicity. This dataset is composed by 60 000 training and 10 000 testing grayscale images of handwritten digits with 28x28 resolution. Orchard et al. [68] proposed a neuromorphic version of the MNIST dataset, which they called NMNIST that was developed in a way to simulate eyes' saccades. This dataset is composed of the same number of training and testing images as the original one and, to acquire the images, the authors used an ATIS sensor and two motors connected to each other through a bracket. They chose to rotate the sensor instead of rotating the image due to its similarity to real-world scenarios. The rotation was done in three different steps, forming and isosceles triangle, each step considered as a saccade. Shortly after, Serrano-Gotarredona and Linares-Barranco [69] also proposed a neuromorphic variant of MNIST, the MNIST-DVS dataset. In this case, however, the authors rotated the images while maintaining a fixed sensor position. To display the moving digits, the authors use an LCD monitor with a frame frequency of 75Hz. Furthermore, to emulate real-world scenarios, where objects may present at different distances, they presented three different scales for the digits (figure 3.1). MNIST-DVS is only composed by 10 000 images from the original dataset, that were upscaled to a total of 30 000 images in the dataset. The same paper presents a simpler neuromorphic dataset composed of about 100 examples of card symbols with 31x31 resolution.

CIFAR-10 [70] is another benchmark dataset composed by 60 000 colored images distributed by 10 different classes (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck) with 50 000 training and 10 000 testing images. This dataset is more complex than MNIST since it has different images with varying backgrounds that may interfere in the algorithms' learning. Li et al. [71] proposed a DVS version of this dataset using a similar approach to the creators of MNIST-DVS, with the images being presented to the neuromorphic sensor through a moving LCD

Table 3.1: Relevant neuromorphic datasets.

Dataset	**Work**	**Total # of images**	**# of Training images**	**# of Testing images**	**# of classes**
NMNIST	[68]	60k	50k	10k	10
MNIST-DVS	[69]	30k	-	-	10
CIFAR10-DVS	[71]	10k	9k	1k	10
NCARS	[73]	24.389k	15.422k	8.607k	2

monitor.

In the context of autonomous driving there are also several developed datasets such as DET [72], that represents the first lane detection neuromorphic dataset. DET is composed of 5 424 event-based images with 1280x800 resolution containing several traffic scenes collected by driving on tunnels, bridges, overpasses, and urban areas. Both per-pixel labels without distinguishing lanes and per-pixel labels with distinction of lanes are provided. Sironi et al. [73], in turn, proposes a binary classification dataset, the N-cars dataset, that consists of 12 336 car samples and 11 693 non-car (background) samples that are split into 7 940 car samples and 7 482 background samples for training with the remaining 4 396 car samples and 4 211 background images for testing. The samples are semi-automatically labeled and manually cleaned to ensure correct labeling. An overview of the relevant classification neuromorphic datasets is present in table 3.1

For 3D perception task, Multi Vehicle Stereo Event Camera (MVSEC) dataset was introduced [74] with data obtained from several different sensors. The ground-truth depth data is generated from a calibrated LiDAR system contributing to stereo depth with the event-based vision sensor [27]. MVSEC dataset presents data acquired in different conditions and speeds, allowing the evaluation of 3D reconstruction in challenging, real-world driving scenarios. For an end-to-end dataset, Binas et al. [75] introduced DDD17, which is composed by recordings of highway and city driving scenes from Switzerland to Germany. The data is acquired with a DAVIS sensor, and it comprises a variety of weather, road, and light conditions.

Chapter 4

Point Clouds

4.1 Point Clouds overview

Accurate perception is crucial in several tasks and, particularly, in autonomous driving, where the system is in contact with a constant highly dynamic, ever-changing environment, as previously mentioned. Hence, RGB cameras are not enough to provide full information of the system's surroundings. Thereby, several sensors are usually integrated into the system to complement each other's acquired information and functionalities. LiDAR has been increasingly studied in the community for autonomous driving applications because of its ability to provide 360-degree weather independent visibility. These sensors act through the emission of pulsed light waves from a laser placed inside the sensor. The light waves will propagate through the air and reflect on the surrounding objects and, consequently, return to the LiDAR. Through the time each wave takes to return, the sensor is able to determine the travelled distance, allowing it to extract the closeness of the objects in the surrounding environment to the system. Additionally, the sensor is also capable of inferring reflective properties of the objects through the measurement of the reflective intensity for each light pulse. In fact, each 3D point is formed by four values, where the first three parameters correspond to the coordinates of the point in the LiDAR referential and the fourth parameter refers to the correspondent point intensity. An example of a LiDAR point cloud is presented in figure 4.1.

There are two major problems in LiDAR data. On one hand, the light beams are emitted by the laser and deviate from the beam propagation axis in which is designated a beam divergent process [77]. In this sense, the point density and intensity will vary with distance, which means that the information collected by the sensor is inversely proportional to the object's distance to it [78]. On the other hand, occlusion also presents a major inconvenience since it leads to incomplete detection of objects, making it difficult to the system to segment and classify such occluded objects. Other minor drawbacks, such as the high cost and the need for higher computing power, are also

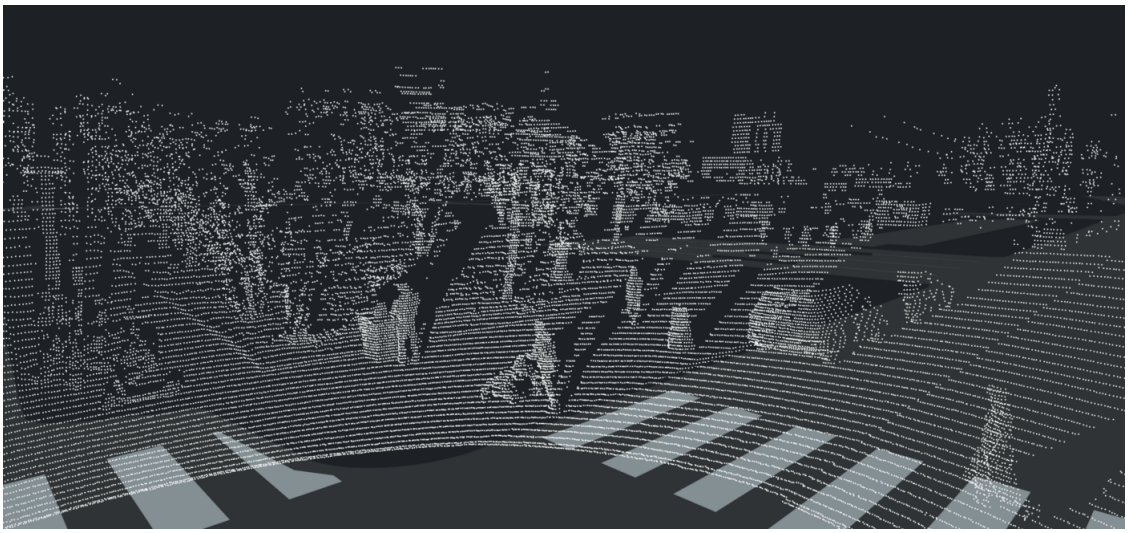

Figure 4.1: Example of a point cloud. Source: [76]

important limitations that lead some ADS companies, such as Tesla, not to use this type of sensors in their solutions [79, 80].

However, point clouds are currently used for a variety of tasks such as segmentation, detection, and classification. While in the segmentation task the models should be able to cluster points with similar properties into homogeneous regions [81], in point cloud detection the purpose is to locate the points of predefined categories and return oriented 3D boxes of the objects from arbitrary 3D point cloud data [82]. Classification, in turn, is about defining the category of a pre-segmented point cloud object.

Even though this work only presents contributions regarding the classification task in point clouds, segmentation and detection methods and works are briefly described since in real world applications classification is highly dependent on these tasks. PointNet [83] appears as one of the most important networks since it is developed in a way that is able to perform object classification, and object or semantic segmentation. For performance evaluation, the authors use overall and average per class accuracy in the classification and intersection over union for the segmentation tasks. Moreover, in semantic segmentation they also used the overall accuracy to assess model's performance. For the three tasks, the model achieved SOTA results or better. Later, the authors introduced PointNet++ [84], a hierarchical network that applies the PointNet recursively, increasing its robustness and ability to learn local features of increasing contextual scales. In turn, Li et al. [85] proposed the PointCNN framework that enables a generalization of the hierarchical representations learned by CNNs in images to point clouds. However, despite the majority of work done with point clouds using traditional ANNs, Wang et al. [86] proposed an SNN-based architecture that uses point cloud data to perform object detection in the context of autonomous driving. The authors use raw LiDAR data without the pulse to point cloud preprocessing step, which arguably improves model's efficiency and reduces the computational power need. Model's performance is assessed in complex and challenging datasets such as the KITTI dataset, and the

achieved performance is in the range of SOTA models, showing lower power consumption.

4.2 Relevant datasets

There are several point cloud datasets for 3D object classification, detection and segmentation. Even though some 3D object detection and segmentation datasets are briefly discussed here, the main focus of this work is related to the classification task, which implies that the most relevant classification-related datasets are discussed with further detail in this section.

The ShapeNet [87] dataset was developed by researchers from Standford and Princeton Universities in conjunction with the Toyota Technological Institute at Chicago and consists of more than 3 million models, from which 220 000 are classified into 3 135 classes under the WordNet taxonomy. The ShapeNet Parts subset, created for part point cloud segmentation, is composed of 31 693 meshes fitted into 16 common object classes, such as table, chair, and plane [88]. PIG-Net [89] represents the SOTA performance for this dataset with an instance average intersection over union of 90.5%. In the context of autonomous driving there are also several segmentation datasets such as the KITTI dataset [90], which constitutes one of the most popular datasets, consisting of several hours of recordings resorting to four high-resolution cameras, one IMU and one Velodyne LiDAR sensor. Other variants of this dataset were created, as the KITTI-CARLA [91], that used the CARLA simulator and identical sensors to the ones used in the KITTI dataset to create a synthetic dataset that, according to the author, allows to improve transfer learning methods from synthetic to a real dataset. Even though the sensors are similar to the ones used in the KITTI dataset, in the synthetic one there are only 2 RGB cameras and 1 LiDAR sensor used. Several other datasets are used for segmentation in the context of autonomous driving, such as Waymo Open dataset [92] and the nuScenes dataset [93], but a detailed description of these fall out of the scope of this work since only the classification task was addressed.

Regarding classification datasets for general purposes, we can highlight the ModelNet40 dataset [94], which is composed by clean synthetic data with the constrain of exhibiting unbalanced and limited training data. It consists of 9 843 training samples with 2 468 samples remaining for the test set. As the name suggests, the samples are dispersed through 40 classes (e.g., airplane, laptop, bookshelf, or lamp) and each sample is uniformly sampled from mesh surfaces, possessing 1024 points. There is the possibility, however, to use a subset of this dataset, which is called ModelNet10 [94], that contains 3 991 and 908 training and testing samples, respectively. Once again, as the name suggests, the samples are distributed through 10 categories. The fact that a fixed point cloud density is used for every samples in these two datasets may imply a decrease in models' robustness since in real-world applications occlusions and under-sampling often occurs, as mentioned before. ModelNet40-C dataset [95] was introduced in order to overcome this problem, providing corrupted images from the original ModelNet40 dataset. This dataset presents images with 15 corruption types and 5 degrees of corruption severity, that allow to emulate real-world scenarios such as occlusion, or noisy signals. ScanObjectNN dataset [96], in turn presents

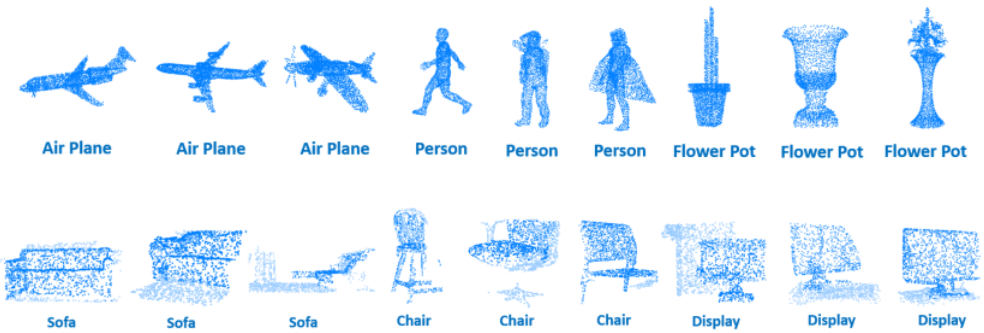

Figure 4.2: Comparison between ModelNet40 (top) and ScanObjectsNN (bottom) datasets. As we can see, the data in the ModelNet40 dataset is cleaner and, ence, more detailed due to its synthetic character, making the ScanObjectNN dataset more challenging. Source: [97]

another challenging classification dataset that can be seen as a more complex alternative to ModelNet40 since it is composed by 15 thousand real-world objects distributed between 15 classes, exhibiting complex backgrounds, missing parts and various deformations that better simulate real data acquisitions. A comparison between ModelNet40 and ScanObjectNN datasets is present in figure 4.2

Considering suitability for the autonomous driving context, the Sydney Urban Objects dataset [98], which is composed by 631 common urban road objects distributed through 14 classes, was proposed. The data can, however, be distributed through 4 major classes: vehicle, pedestrian, sign, and tree. This work aims to provide real-world sensing scenarios, presenting highly challenging data due to highly occluded objects.

As we have seen, LiDAR is mainly used in autonomous driving industry due to their intrinsic properties. However, there are other areas that try to take advantage of point cloud data. In fact, Yang et al. [99] proposed a medical point cloud dataset for aneurysm segmentation and classification, composed by 1694 healthy vessel segments and 215 aneurysm segments.

4.3 Classification on point clouds

While segmentation and recognition tasks are crucial to an accurate perception, objects' misclassification can imply fatal consequences in ADS, since the whole system's pipeline is dependent on

Table 4.1: Relevant point cloud datasets for classification.

Dataset	**Work**	**Total # of images**	**# of Training images**	**# of Testing images**	**# of classes**
ModelNet40	[94]	12.311k	9.843k	2.468k	40
ModelNet10	[94]	4.899k	3.991k	908	10
ModelNet40-C	[95]	12.311k	9.843k	2.468k	40
ScanObjectNN	[96]	2.902k	-	-	15
Sydney Urban Objects	[98]	588	433	155	14

this task to act. Thus, a robust and reliable classification model is needed to improve autonomous systems' robustness and performance.

Classification methods in point clouds can be broadly divided into three groups, according to Guo et al. [100]: multi-view based-, volumetric-based-, and point-based methods, depending on the representation that is fed to the neural network.

In multi-view based methods, point clouds are projected to multiple 2D images and, after several key features are extracted, fused for accurate classification. Hang Su et al.[101] proposed an architecture that is composed by a backbone of standard CNNs trained to independently recognize critical features of the different views. Then, the information from multiple views is fed to a shape descriptor that is able to accurately classify the data. The authors were able to achieve a maximum accuracy of 90.1% in the ModelNet40 classification dataset. Maturana and Scherer [102], in turn, exploited region-to-region and view-to-view relationships through the use of a 2-block relation network, where the first reinforcing block is responsible for the exploration of region-to-region relationships, allowing the reinforcement of the information for each individual view and the second block, named the integrating block, is responsible for the effective integration of each individual view, through the exploration of view-to-view relationships. The information is then aggregated to obtain a discriminative 3D object representation and allow accurate classification. The authors assessed model's performance on the ModelNet40 and ModelNet10 datasets, obtaining 94.3% and 95.3% accuracy, respectively.

Regarding volumetric-based methods, point clouds are usually voxelized into 3D grids and then fed to a 3D CNN. In this sense, VoxNet [103] appeared to improve the classification accuracy of 3D objects. The authors present a system with two main components, with the first being a volumetric occupancy grid and the second a 3D CNN that classifies the data directly from the grid. The authors assess the model's performance in several datasets, such as the Sydney Urban Objects dataset and argue that their best model outperforms SOTA results for the evaluated datasets, while performing real-time classification. PointGrid [104], on the other hand, was proposed as a hybrid model that takes into consideration both grid and point information, with a constant point count for each grid, which allows the network to extract hierarchical and local geometry shape details using 3D convolutions.

On the other hand, point based methods are becoming increasingly popular since they use raw point cloud data, implying that no information is lost and the model has access to the full information, allowing it to extract critical features. Aforementioned PointNet and PointNet++ are the most popular networks that use point-based classification, achieving an accuracy of 89.2% and 91.9%, respectively, on the ModelNet40 dataset.

Even though point cloud classification is being widely studied in conventional ANNs, work combining these data and SNNs is rare. Nonetheless, given the potentially sparse nature of point clouds, classification on these data could benefit from SNNs.

Chapter 5

Methodology

The field of SNNs is evolving and in the last few years outsanding breakthroughs have been made from local plasticity rules to deep SNNs trained with surrogate gradients, and meaningful encoding strategies. Notwithstanding, the performance of some SNNs is highly influenced by the set of model hyperparameters. Yet, not all published research papers present the used set of hyperparameters or provide source code, meaning there are some reproducibility issues, in particular with the highly sensitive SNNs. Therefore, we try to bridge that gap by replicating several SOTA methods. Additionally, we experiment with different concepts to assess their impact on model performance. Finally, given the expected hardware constraints in Autonomous Driving (AD) and subsequent need for efficient algorithms, we try to explore the suitability of current SNN algorithms for ADS, more concretely, to process LiDAR data.

5.1 Spike-timing-dependent plasticity

This work starts by replicating the algorithm of Diehl and Cook [25], due to its simplicity and biological plausibility. The algorithm contains many bio-inspired features, such as local learning (STDP), lateral inhibition, homeostasis, and the widely adopted LIF neuron model.

The proposed architecture is a two-layered network in which the first (input) layer works as a spike generation layer while the second layer consists of a set of excitatory neurons and as many inhibitory neurons. Each input (pre-synaptic) neuron receives a spike, representing the Poisson-encoded pixel intensity, that is then multiplied by the synaptic weights before being integrated at the excitatory (post-synaptic) layer. In turn, each post-synaptic neuron is connected one-to-one to a set of inhibitory neurons that project back to all other excitatory layer neurons, except to the one it receives a spike from. This emulates lateral inhibition. Moreover, to impede a single neuron from dominating over all others and to increase competition, the authors propose an adaptive membrane threshold. A schematic representation of the proposed architecture can be found in figure 5.1.

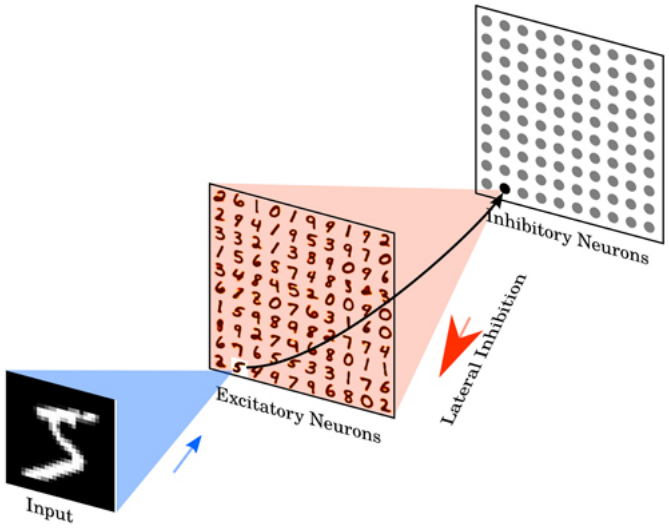

Figure 5.1: Architecture proposed by Diehl and Cook [25]. The highlighted excitatory neuron that fires when the illustrated input is fed to the network is connected to an inhibitory equivalent that will inhibit all other excitatory neurons from firing.

To replicate this work, we chose the BindsNET simulation software [105] and we further investigate the influence of different hyperparameters in models' performance. We opted for BindsNET in detriment of the Brian simulator [106] as it is more ML-oriented, thus easing hyperparameter tuning. Moreover, BindsNET allows to take advantage of GPU-acceleration and rapid prototyping. Besides the MNIST dataset, we investigate if the proposed model could scale to neuromorphic (i.e. event-based) data. More precisely, we test the algorithm on the NMNIST [68] dataset. Here we are not concerned with the biological plausibility (e.g., weights to emulate synaptic conductance), rather than with model performance. For both datasets, we used 60k and 10k images for training and testing sets, respectively (3.1).

A variant of the work of Diehl and Cook [25], based on the Self Organizing Map (SOM) [107], has also been proposed [54]. We experimented with this strategy as well, where inhibition would increase linearly over training within a predefined interval $[c_{min}, c_{max}]$, where c_{min} represents the initial minimum value for inhibition strength and c_{max} the maximum inhibition strength. The range of inhibitory strengths used during training is defined through a hyperparameter defined as the number of splits of the training data. This approach allows the network to freely learn representations in an initial stage, when the inhibition is lower, and then individualize their RFs, to reduce redundancies, later, when the inhibition is higher. Additionally, an inter-neuron distance-dependent inhibition scheme, also proposed by Hazan et al. [54], was investigated. In this approach, the inhibition exerted to each neuron is proportional to the square root of the Euclidean distance to the neuron that fired. This is the so-called SOM approach, that the authors argue allows a better convergence of the performance while making it possible to achieve satisfactory results with only a subset of the input data. Furthermore, in such relaxed inhibition concept, neighboring neurons are encouraged to learn similar filters, thus promoting smooth filter maps between classes. These two approaches were also evaluated on the MNIST dataset to define a

baseline, and then applied to NMNIST to assess scalability and robustness to neuromorphic data.

5.2 Supervised learning

SL supported on BP is not a biologically plausible strategy. Although it presents, by far, the most promising results in both ANNs and SNNs, the algorithm differs from the working principles of the mammalian brain [10]. Nonetheless, it is still the most scalable solution and, for the purpose of this work, we explore some of its principles. Namely, we explore a Deep Continuous Local Learning (DECOLLE) [57] strategy, in which a local gradient algorithm is employed to train a multilayer SNN. The algorithm tries to unify the advantages of the backpropagation algorithm with the benefits and biological plausibility of local learning. The strategy is similar to FB alignment mechanisms. To force locality, each hidden block of a multilayer network is connected to a readout layer of non-trainable fixed random weight matrices, and a pseudo-target is defined to compute the local error. Then, the error is backpropagated, locally, by setting all non-local gradients to zero. For the backward pass, and to overcome the weight transport problem, random weight matrices are used. More specifically, the authors implement a sign-concordant FB alignment mechanism, where the forward random fixed weights are multiplied by fixed multiplicative Gaussian noise.

Then, a three-layered network composed of alternate convolutional, pooling, and spiking blocks (Table 5.1 and figure 5.2) is trained with AdaMax optimizer and smooth L1 loss (see equation 5.1), and assessed on the NMNIST and DVS-Gesture [108] datasets. As previously stated, each hidden block possesses a random readout layer with unique pseudo-targets. For the first layer, a linear ramp function is used, while for the second- and third-layers sinusoidal functions are defined, although of, respectively, high and low frequencies (figure 5.3).

$$Loss = mean(L) \tag{5.1}$$

$$L = \{l_1, l_2, ..., l_N\}^T \tag{5.2}$$

with N being the batch size. For each sample, l_n is defined as

$$l_n = \begin{cases} \frac{0.5 \cdot (x_n - y_n)^2}{\beta}, & \text{if } |x_n - y_n| < \beta \\ |x_n - y_n| - 0.5 * \beta, & \text{otherwise} \end{cases} \tag{5.3}$$

A similar algorithm was evaluated on this thesis. However, in the original implementation, the authors propose a novel neuron dynamics that accounts for two different compartments (somatic and dendritic compartments), creating a biologically plausible model. Furthermore, a regularization step was used to ensure the average firing rate of the somatic membrane potential (U) was kept within a pre-defined interval. Yet, these two mechanisms were not assessed for the purpose of our work. Another key difference is that the authors propose for gradient updates to only be made after the first 50 ms, which we ignore completely in this work. Our primary focus was on simplicity and performance, in detriment of biological plausibility, and with some expected loss

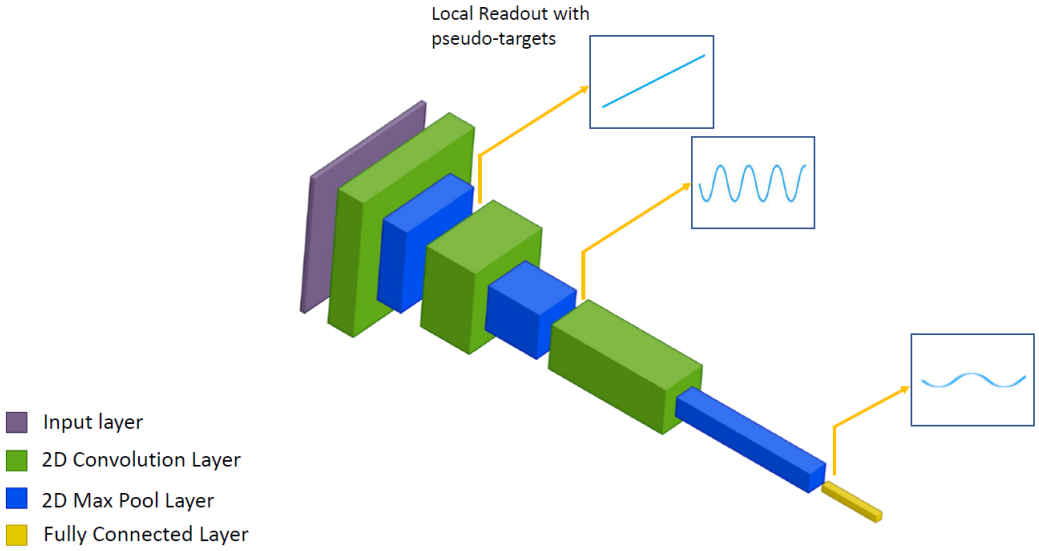

Figure 5.2: Convolutional DECOLLE architecture. Similarly to the MLP DECOLLE approach, each spiking block contains a readout layer with specific pseudo-targets.

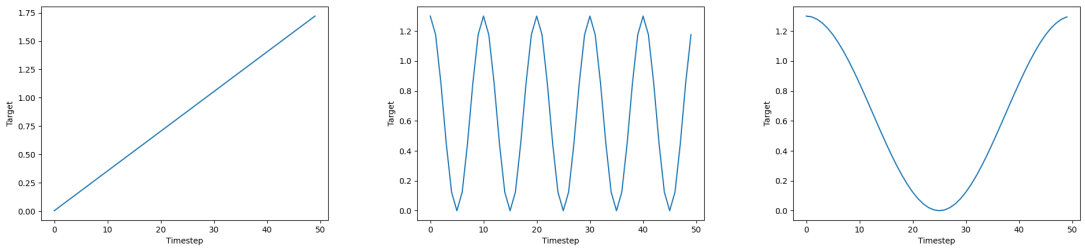

Figure 5.3: Illustration of pseudo-target functions for the first (left), second (middle), and third (right) hidden layers.

Table 5.1: Network implementation characteristics. The numbers inside the parenthesis represent the implementation introduced in this work in relation to the original implementation proposed by [57].

Layer type	**kernel size**	**# of filters**
Input		
Convolutional	7x7	64 (32)
Max Pooling	2x2	64 (32)
Spiking		
Dropout (p = 0.5)		
Dense		11 (10)
Convolutional	7x7	128 (64)
Spiking		
Dropout (p = 0.5)		
Dense		11 (10)
Convolutional	7x7	128 (128)
Max Pooling	2x2	128 (128)
Spiking		
Dropout (p = 0.5)		
Dense		11 (10)

of efficiency. Nonetheless, like the original contribution, we also consider the simple LIF neuron model. Additionally, we opt for a learning rate schedule in which the learning rate decreases by a factor of 0.1, every 10 epochs. A similar strategy was suggested in the original approach [57]. The proposed algorithm was evaluated on the NMNIST dataset. In table 5.1 we can find a summary of the architecture adopted in this work compared with the originally proposed solution [57].

Besides the previously mentioned convolutional network, a simpler version of the DECOLLE algorithm was also considered. In this approach, a MultiLayer Perceptron (MLP), composed of three spiking layers (figure 5.4) was implemented and assessed on the MNIST and NMNIST datasets. To distinguish between approaches, this implementation is named MLP DECOLLE for the rest of this work.

The snnTorch [109] package was chosen to perform these experiments due to its similarities with typical PyTorch [110] workflows. Furthermore, this package is specifically designed for performing gradient-based learning with SNNs. Besides, the GPU-enabled training of snnTorch is also advantageous to increase the computations' speed.

5.3 Point clouds

The NMNIST [68] dataset is event-based, meaning it can be directly used as SNNs' input. On the contrary, common computer vision benchmarks (e.g. MNIST), are frequently encoded resorting to rate coding strategies, like Poisson or Bernoulli encodings [111, 112, 113]. But the study of LiDAR data for SNN-based applications is highly limited, which implies that there is not yet a benchmark spike encoding method for point clouds. In this work, we present a two-step approach

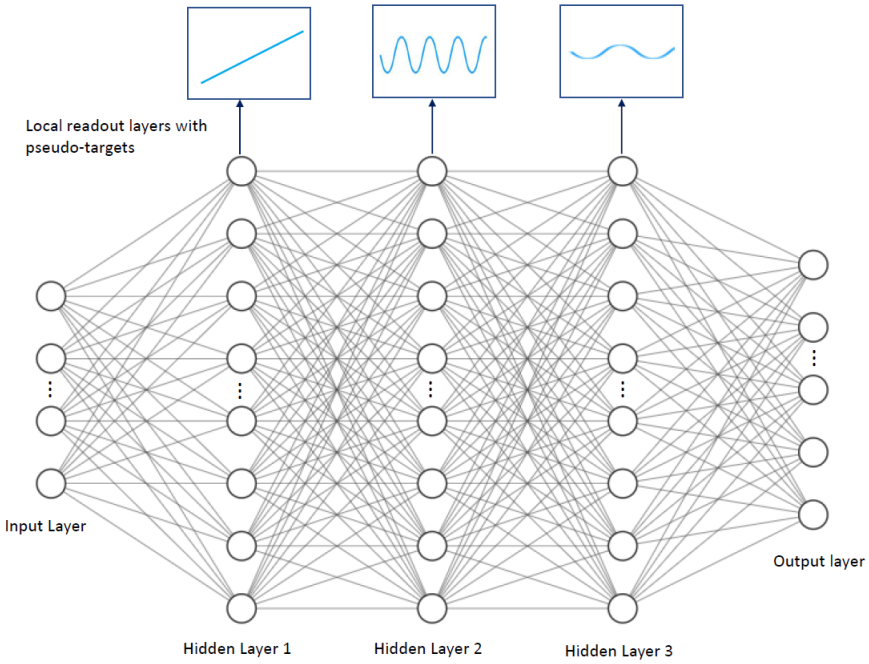

Figure 5.4: MLP DECOLLE architecture, composed of three spiking layers with a local readout layer with a specific pseudo-target each.

to encode LiDAR data, i.e., point clouds, to SNN-suitable spike trains. Considering a LiDAR point cloud dataset where each data point is represented as x, y, and z real-world coordinates (see figure 5.5 (left)), the approach is as follows:

1. **Information discretization:** in this step, the yz plane is discretized to a pre-defined number of pixels. In our case, we opted for an output size of 64 x 64.

2. **Intensity assignment:** An intensity within the range $[0, 255]$ is assigned to each pixel based on the following equation:

$$intensity = \frac{\bar{d}_{x,i}}{x_{max} - x_{min}} \cdot 255 \tag{5.4}$$

Where x_{max} and x_{min} represents the maximum and minimum x coordinates present in the sample, respectively, and $\bar{d}_{x,i}$ represent the mean distance to the origin, along the x axis, in the i^{th} voxel. The value of $\bar{d}_{x,i}$ is computed as in the following equation:

$$\bar{d}_{x,i} = \frac{\sum_P x_{p,i}}{P_i} \tag{5.5}$$

With $x_{p,i}$ representing the x coordinate of point p in the i^{th} pixel and P_i the total number of points lying within that pixel position, after discretization. This way, we obtain a 2D grayscale image to which we can apply rate-based coding.

Three examples of the proposed encoding method can be found in figure 5.5 (right).

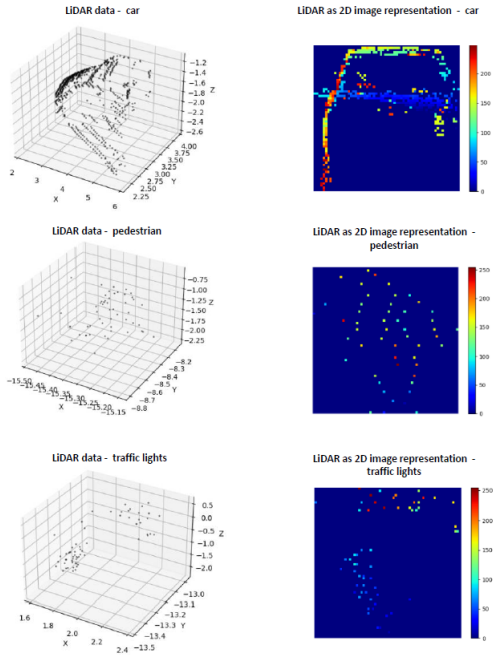

Figure 5.5: Point cloud (left) to 2D image (right) encoding method. The proposed method allows us to transform a 3D point cloud to a grayscale 2D image to which we can then apply rate encoding.

Given such an encoding strategy, we then applied the previously described DECOLLE algorithm to the Sydney Urban Objects point cloud dataset [98], using 533 images for training and 155 for testing (4.1). Our main goals were to assess the performance of SNN-based approaches on LiDAR data and to compare their differences, in terms of performance, with conventional ANNs.

Chapter 6

Results and discussion

6.1 Spike-timing-dependent plasticity

As previously stated, for the replication of the work of Diehl and Cook [25], we started to manually study the hyperparameters' influence in the performance of the model, evaluating it on the MNIST dataset, in a first instance, to establish a baseline and attempt to replicate the reported accuracy. Then, the NMNIST dataset was used to assess the scalability of this model to neuromorphic data.

The excitatory to inhibitory and inhibitory to excitatory synaptic strengths are crucial to the model's performance as the authors argue that it should not be too weak, since it would not produce any effect, nor too strong, in which case once a neuron fires, no other neuron is able to fire too. Furthermore, the adaptive threshold (θ^+) concept is also studied in this work.

As we can see from table 6.1, we began to study the excitatory and inhibitory strengths through the BindsNET default values (17.5 inhibitory and 22.5 excitatory strength). Then, a random search algorithm was developed where an inhibitory strength of 32.0 and an excitatory strength of 64.0 were suggested as possible optimal values. In fact, it improved the accuracy by 26%. However, since the random search algorithm does not cover all possible combinations, it may create a suboptimal solution. This is particularly important in the considered algorithm, as excitatory and inhibitory strengths directly impact learning. To assess if we were on such a suboptimal case, another random search algorithm was implemented but we obtained 35.0 for the inhibitory synaptic strength and 231.0 for the excitatory strength, which allowed a further increase of approximately 1% in the test set. Another study was performed using the values provided by the BindsNET authors in their examples directory (120 inhibitory and 22.5 excitatory strength) that allowed to reach an accuracy of 90.28% with 400 neurons, outperforming the 87% accuracy reported in the original paper for the same number of neurons. Different θ^+ and number of neurons were evaluated and, although we were not able to outperform the accuracy of Diehl and Cook [25] for the network of 100 neurons, the accuracy obtained with 800 neurons is only 0.5% apart from the top performance, reported with 1600 neurons in the original paper. Therefore, we were able to obtain comparable

Table 6.1: Results for the experiments of different hyperparameters for the implementation of the network proposed by [25]. The best results for the different number of neuron architectures is presented in bold.

Nr of Neurons	**Inhibitory strength**	**Excitatory Strength**	θ^+	**Accuracy (%)**
400	17.5	22.5	0.05	46.06
400	32.0	64.0	0.05	72.41
400	35.0	231.0	0.05	73.55
400	**120.0**	**22.5**	**0.05**	**90.28**
400	120.0	22.5	0.01	86.65
400	120.0	22.5	0.1	89.18
800	**120.0**	**22.5**	**0.05**	**91.41**
100	**120.0**	**22.5**	**0.05**	**75.64**

results, with less neurons, highlighting the robustness of the Diehl and Cook algorithm [25]. A comparison of the original results and the best accuracies achieved in this work are summarized in table 6.2.

As we can see in figure 6.1, where it is represented the confusion matrix for the best accuracy obtained, the 4 and 9 are the most common misclassifications, which is consistent with the results reported by the original paper.

Regarding the neuromorphic data, the inhibitory and excitatory strengths were fixed, since their influence has already been analysed on the MNIST study, and we rather focused on the influence of the number of timesteps (T) and adaptive threshold, θ^+. Table 6.3 presents the obtained accuracies for the NMNIST study.

We can see that different values for both T and θ^+ allow an improvement in the model's accuracy until they reach a point for which the performance starts decreasing. However, the best accuracy is still far from the one reported for the MNIST dataset. We argue this derives from the sparsity and higher complexity of the NMNIST dataset. In fact, besides being a bipolar dataset, the neuromorphic data is noisier than MNIST, as we can see in figure 6.2 , where we sum the time dimension of NMNIST to convert the time-varying data to 2D images. As such, it becomes difficult for the network to learn representative prototypes, i.e., RFs.

For the so-called SOM approach, it was tested again on the MNIST and NMNIST datasets. Even though the increasing inhibition strategy, where the inhibition is linearly increased over the training was implemented, there were no relevant differences, so only the increasing interneuron

Table 6.2: Comparison of the results obtained in this work with the ones reported by [25]

Work	**Nr of Neurons**	**Inhibitory strength**	**Excitatory Strength**	**Accuracy (%)**
[25]	100	17.5	22.5	82.90
This work	100	120.0	22.5	75.64
[25]	400	17.5	22.5	87.00
This work	400	120.0	22.5	90.28
[25]	1600	17.5	22.5	91.90
This work	800	120.0	22.5	91.41

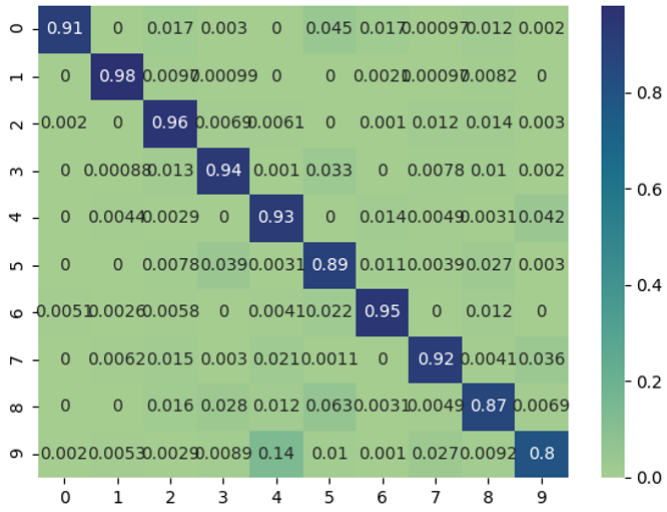

Figure 6.1: Confusion matrix for the best accuracy obtained through the implementation of the network proposed by [25].

Table 6.3: Results for the experiments of different hyperparameters for the implementation of the network proposed by [25] in the NMNIST. The best results for the different number of neuron architectures is presented in bold. All activity accuracy represents the accuracy obtained based on the activity of all output neurons while in proportion accuracy,a weighted class-wise accuracy is computed.

Nr of Neurons	**T**	θ^+	**All activity Accuracy (%)**	**Proportion Accuracy (%)**
400	250	0.05	48.87	47.86
400	250	0.1	65.03	64.20
400	250	0.2	65.62	64.86
400	250	0.3	67.26	67.59
400	250	0.4	72.17	73.67
400	250	0.5	71.66	71.75
400	300	0.4	75.45	76.29
400	350	0.4	75.08	77.08
400	400	0.4	78.69	79.99
400	450	0.4	**79.98**	81.17
400	500	0.4	79.8	**81.46**
400	550	0.4	79.79	80.92
400	750	0.4	78.31	80.00
400	1000	0.4	76.55	78.22

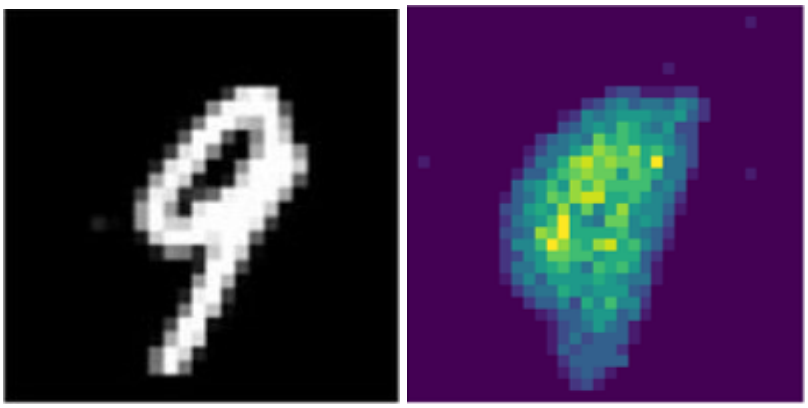

Figure 6.2: Comparison between MNIST (left) and NMNIST (right) examples. As we can see, the neuromorphic image is noisier, making it difficult to the network to learn representative features.

inhibition scheme results will be here discussed. In fact, as expected, we obtained smoother filter maps when compared with the fixed inhibition approach (figure 6.3). However, even though a 400 neuron network is able to achieve 90.71% accuracy on the MNIST dataset (comparing to the 90.28% previously obtained), for the neuromorphic dataset the best accuracy obtained was 53.9%, also with a 400 neuron network. This implies a scalability difficulty of this approach to even the smallest complexity change, such as is the case when we scale up to the NMNIST dataset.

In summary, in this section we have seen that although the proposed shallow network is able to perform relatively well on the simple MNIST dataset, it does not scale well to more complex datasets such as NMNIST that, although still simple, holds noisier and bipolar data. However, a comprehensive study of different hyperparameters allow us to better understand unsupervised SNNs, and since we made our code publicly avilable [1], we also contribute to the SNN community by ensuring reproducibility of the methodology.

6.2 Supervised learning

Biologically plausible STDP-based approaches do not scale to complex data, thus creating a necessity for different learning rules that work well for deeper networks. In that sense, two dominant strategies have been proposed: converted ANN-to-SNN [114, 115, 116] and supervised SNNs trained with backpropagation and surrogate gradients [59, 58]. Nonetheless, we explore DECOLLE, an intermediate strategy that is local, like STDP, but that also takes advantage of the strengths of supervised learning, namely, backpropagation. We explore two different network topologies, namely a convolutional network and an MLP. Then, we evaluate the results with different hyperparameters. The general case of the pseudo-targets for, respectively, the first, second, and third hidden layers are described by equations 6.1, 6.2, and 6.3:

$$y = \beta_0 t + \beta_1 \tag{6.1}$$

[1] Source code available at: https://github.com/MAC2189/Efficient-Neuromorphic-Architectures-for-Visual-Perception

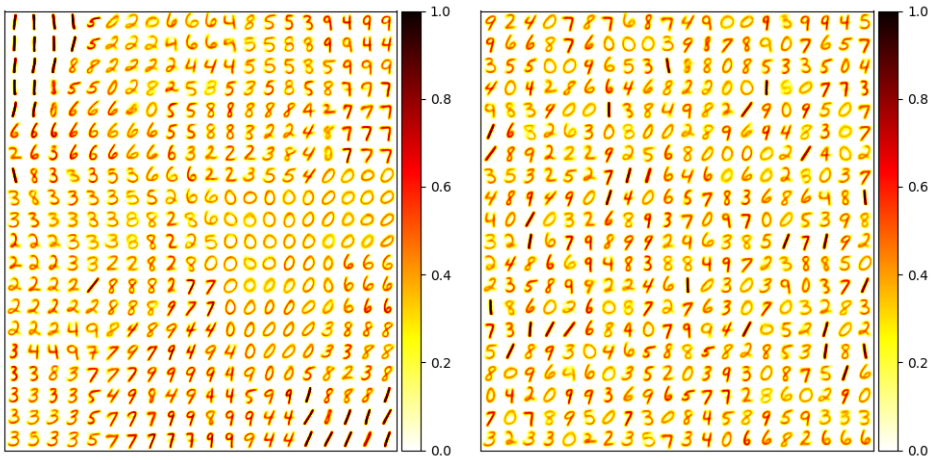

Figure 6.3: Comparison of the filter maps created with the implementation of the network with (left) and without (right) the SOM approach.

$$y = \mathbf{A_1} \cdot (sin(2\pi \cdot \mathbf{f_1} \cdot t + \frac{\pi}{2}) + 1) \tag{6.2}$$

$$y = \mathbf{A_2} \cdot (sin(2\pi \cdot \mathbf{f_2} \cdot t + \frac{\pi}{2}) + 1) \tag{6.3}$$

With β_0 being the slope of the linear ramp, and β_1 the intercept. $\mathbf{A_i}$ and $\mathbf{f_i}$ are, respectively, the amplitude and frequency of the sinusoidal pseudo-target associated with the i-th hidden layer.

For the simpler version of the DECOLLE algorithm, composed of three fully-connected LIF layers, the reported values for the pseudo-targets' parameters were manually determined as well as the number of timesteps and learning rate. Importantly, we experimented with two versions of the algorithm. One version where the forward and backward pass weights were equal and another, where we exploited the weight multiplication by fixed gaussian noise for the BP. Layers composed of 400, 800, and 1600 neurons were evaluated. Since the 800 neuron networks provided the best results, we then fixed the number of neurons to 800 to assess the impact of other features of this architecture on model performance. Furthermore, only the slope and frequencies of the pseudo-targets were assessed, with the intercept and amplitude values held fix and equal to 0.005 and 0.35, respectively. As in the previous experiments, the results were initially tested on the MNIST dataset, to establish a baseline, and then on neuromorphic data, namely NMNIST. As we can see from table 6.4, the results for the equal forward and backward local random readout weights suggest that the used SL method surpasses the UL performance, allowing us to train deeper networks, while maintaining biological plausibility.

Other experiments were conducted, including a model trained with a single channel of the NMNIST dataset as input, or variations of the rate-coding strategy, but not included in the results table since the performances did not improve. Table 6.5, on the other hand, shows the results for the multiplication of the forward random readout weights with fixed gaussian noise for the backward pass. The results suggest that, although there is a gain in biological plausibility, there

Table 6.4: Results for the implementation of the MLP DECOLLE approach in MNIST and NMNIST datasets. For the MNIST applications, the probability of spike was fixed to 0.8. The better performances are highlighted in bold.

Dataset	**T**	β_0	f_1	f_2	**Learning Rate**	**Epochs**	**Accuracy (%)**
MNIST	100	0.0075	0.04	0.02	5e-4	5	90.45
MNIST	50	0.0075	0.04	0.02	5e-4	5	91.19
MNIST	50	0.015	0.08	0.04	5e-4	5	92.29
MNIST	50	0.0225	0.12	0.06	5e-4	5	79.92
MNIST	100	0.015	0.08	0.04	5e-4	5	37.48
MNIST	50	0.015	0.08	0.04	1e-4	5	95.12
MNIST	50	0.015	0.08	0.04	1e-4	10	94.39
MNIST	50	0.015	0.08	0.04	5e-5	5	95.95
MNIST	50	0.015	0.08	0.04	1e-5	5	95.67
MNIST	50	0.015	0.08	0.04	1e-5	10	95.76
MNIST	**50**	**0.015**	**0.08**	**0.04**	**1e-5**	**20**	**96.32**
MNIST	50	0.015	0.08	0.04	1e-5	50	96.24
NMNIST	50	0.015	0.08	0.04	5e-5	5	89.13
NMNIST	400	0.015	0.08	0.04	5e-5	5	89.13
NMNIST	**50**	**0.015**	**0.08**	**0.04**	**1e-5**	**20**	**95.28**

is a slight decrease in performance, which is in accordance with the original work [57]. In our experiments, only the number of neurons and the pseudo-targets' parameters were studied, with the learning rate fixed at 5e-5. Additionally, the network was trained for 20 epochs with 50 timesteps for each example.

Regarding the network with convolutional and LIF-neuron blocks, to which we shall name convolutional DECOLLE for the rest of this analysis, different learning rates, timesteps, pseudo-targets' parameters, and beta values, that rule the membrane potential decay dynamics of the LIF neurons, were tested. In every experiment, the network was trained through 25 epochs using the AdaMax optimizer. For the first and second hidden layers, the pseudo-target's parameters were also fixed on $\beta_0 = 0.035$, $\beta_1 = 0.005$, $A_1 = 0.65$, and $f_1 = 0.10$, respectively. As we can see from table 6.6, it was studied an approach in which the beta value was set to 0.95 in the output layer, while on the other layers it was settled on 0.75. Moreover, it was also implemented a membrane-

Table 6.5: Results of the MLP DECOLLE approach with gaussian white noise being multiplied by the fixed random weights in the backward pass. All the experiments were performed on the NMNIST dataset.

Nr of neurons	β_0	A_1	f_1	A_2	f_2	**Accuracy (%)**
800	0.015	0.35	0.08	0.35	0.04	84.94
1600	0.015	0.35	0.08	0.35	0.04	88.01
800	0.025	0.55	0.10	0.55	0.02	92.75
800	**0.035**	**0.65**	**0.10**	**0.65**	**0.02**	**93.64**
1600	**0.035**	**0.65**	**0.10**	**0.65**	**0.02**	**94.56**

Table 6.6: Results for the convolutional DECOLLE on the NMNIST dataset.

Learning Rate	**T**	**f_2**	**Accuracy (%)**	**Comments**
5e-5	50	0.02	92.52	-
5e-5	75	0.02	89.56	-
5e-5	50	0.04	91.43	-
5e-5	50	0.03	91.13	-
1e-4	50	0.02		-
1e-4	**50**	**0.02**	**96.50**	**output beta = 0.95**
1e-4	50	0.02	95.50	sigmoid approach w/ membrane-based loss
1e-4	50	0.02	95.74	membrane-based loss

based smooth L1 loss for the output layer that, instead of computing the loss by establishing target spikes, it resorts to membrane potential as target. Moreover, a cyclic sigmoid target approach is also assessed. Besides using the membrane-based loss, a sigmoid function is used as a surrogate for the output layer ground truth. Our hypothesis with such an approach was that a time varying target more closely mimics the temporal behavior of a LIF neuron, than forcing the neuron to be close to the threshold at each timestep, thus representing a more accurate estimate of the prediction error. In essence, we intended to penalize less the error for the timesteps in which it was expected for the LIF to be accumulating voltage.

Some additional experiments were conducted where an additional convolutional layer was added to the third hidden block and the BP algorithm was implemented from the output to that layer, remaining the rest of the layers with a local learning rule. For the second convolutional block, a 5 x 5 kernel size was used and the values of the learning rate, timesteps, and epochs were set to 5e-5, 50, and 25, respectively. The pseudo-targets' parameters were fixed to the values that produced best results with the convolutional DECOLLE approach. As we can see from table 6.7, the performance did not improve significantly.

When analysing the results obtained with the different approaches, we observe that with the simpler MLP DECOLLE architecture we were able to achieve performances comparable to the ones we obtained with the MNIST dataset and to the results reported by the authors. However, the use of convolutional blocks in the hidden layers led to significant performance improvements. We suggest that with such an approach, the network is able to learn more representative features. Nonetheless, we were able to show the advantages of biologically plausible SL rules in terms of performance and robustness with the implementation of the proposed networks on neuromorphic

Table 6.7: Results on the convolutional DECOLLE approach, where it was added a second convolutional block to the third hidden layer and the BP algorithm was implemented from the output to the last hidden layer. The experiments were performed on the NMNIST dataset.

Learning Rate	**Kernel size**	**Accuracy (%)**
5e-5	**7 + 5**	**92.92**
5e-5	7 + 5 (w/ AvgPool)	92.10

Table 6.8: Results for the implementation of the 2-layered fully-connected ANN to allow comparison with SNN-based implementations.

Nr of neurons	Optimizer	Batch Size	Accuracy (%)
100	AdaMax	4	42.11
100	Adam	4	42.76
75	AdaMax	4	38.82
150	AdaMax	4	41.45
200	**AdaMax**	**4**	**48.03**
200	AdaMax	8	40.79
100	AdaMax	8	43.42

data. Overall, the DECOLLE algorithm enables the development of multilayer models with the advantages of local learning. More precisely, one could consider this approach as an efficient and scalabe transitional algorithm from conventional ANN models to SNNs, while no scalable, biologically plausible, local learning alternative is proposed. Importantly, overall, we observe a performance improvement when comparing the results of the STDP-based approaches with the supervised learning algorithms, suggesting that, although there is some loss in efficiency and biological plausibility, a more adequate accuracy/efficiency trade-off is achieved. In fact, supervised SNNs could be suitable for ADS as with this use case, a high performance is highly desirable due to road safety and regulatory reasons, but where hardware constraints are also imposed.

6.3 Point clouds

Regarding the suitabilty of SNNs to point cloud data, the previously described convolutional DECOLLE approaches were assessed. Since our conversion method from LiDAR point clouds to 2D images presents some limitations, namely, leads to some loss of spatial resolution, we hypothesised both shallow conventional ANNs and SNNs would not be robust with these data. Another limitation is in regard to the sparseness of this dataset, that could dampen the performance of conventional CNNs. Moreover, we aimed for a direct comparison between SNNs and CNNs. Therefore, we started by implementing several fully-connected artificial neural networks with 2 layers of varying hidden neurons' number, batch size, and optimizer (table 6.8). A learning rate schedule was also adopted where an initial value of 1e-3 was decreased by a factor of 0.1 every 10 epochs.

Considering the DECOLLE MLP, we were able to obtain a best accuracy of 45.86%, which is slightly inferior to the performance of the traditional ANN. Next, the convolutional DECOLLE was applied on the LiDAR data with the AdaMax optimizer, 7 x 7 kernel size, and pseudo-targets parameters fixed. The pseudo-targets parameters used were $\beta_0 = 0.035$, $\beta_1 = 0.005$, $A_1 = A_2 = 0.65$, $f_1 = 0.10$, and $f_2 = 0.02$. The rest of the parameters were studied, and the results are described in table 6.9.

For the comparison with SNN-based performances, two strategies based on traditional neural networks were addressed. On the one hand, a convolutional DECOLLE version was implemented

Table 6.9: Results on the convolutional DECOLLE strategy. The parameter *Scheduler* is labeled **Y** if the learning rate was multiplied by 0.1 every 10 epochs and **N** if it was used a fixed learning rate for the totality of the training epochs.

Learning Rate	**Scheduler**	**Batch Size**	**Epochs**	**Beta**	**T**	**Max prob. spike**	**Dropout prob**	**Accuracy (%)**
5e-5	Y	4	25	0.85	75	1	0.5	44.74
5e-5	Y	4	50	0.85	75	1	0.5	44.74
1e-4	Y	4	25	0.85	75	1	0.5	46.05
1e-4	Y	4	25	0.85	50	1	0.5	45.39
1e-4	Y	4	25	0.85	50	0.8	0.5	44.08
1e-4	Y	8	25	0.85	75	0.8	0.5	50.00
1e-4	Y	8	25	0.85	75	0.8	0.2	52.63
5e-4	Y	8	25	0.85	50	0.8	0.5	56.58
5e-4	Y	8	25	0.85	25	0.8	0.5	55.26
5e-4	**N**	**8**	**25**	**0.85**	**25**	**0.8**	**0.2**	**57.24**
5e-4	Y	6	25	0.85	75	0.8	0.2	56.94
5e-4	Y	12	25	0.85	75	0.8	0.2	56.25
5e-4	Y	12	25	0.75	75	0.8	0.2	56.94
5e-4	Y	8	25	0.75	75	0.8	0.2	52.63
5e-4	Y	8	25	0.75	25	0.8	0.2	55.92

in which the LIF neurons are replaced with conventional activation functions, namely ReLU and Sigmoid activations (table 6.10). Since the input to conventional ANNs, in the 2D computer vision domain, is not time varying, the pseudo-targets are set to be equal to the actual target, for all layers. We argue this also allows the different layers of the network to learn distinct and discriminant features, for two reasons mainly. Not only because the local random readout weights are unique for each layer, but also because of the increasing number of filters with network depth. On the other hand, an end-to-end traditional ANN was defined (table 6.11) as well as an end-to-end convolutional SNN (table 6.12) to assess the influence of the learning method. Furthermore, the additional convolutional block approach was also implemented with the convolutional DECOLLE approach but, as we can see from table 6.13, did not provide satisfactory results comparing to the ones obtained without the additional convolutional block. Finally, an inference study was performed in the SNN- and ANN-based convolutional DECOLLE approaches, with the parameters that provided the best results for each strategy. We can see from table 6.14 that the inference in the SNN-based convolutional DECOLLE is higher than in the ANN-based strategy even if we consider the mean inference time for each timestep. We argue that this can be caused by the usage of the snnTorch package, that works with recursion rather than with the concept of time. Furthermore, the use of neuromorphic hardware could improve the results for the SNN-based architecture and should be considered as a study case for future implementations.

Table 6.10: Results of the convolutional DECOLLE strategy applied to traditional ANN, with different activation functions. For all experiments, the AdaMax optimizer was used.

Learning Rate	**Kernel Size**	**Batch Size**	**Epochs**	**Dropout prob**	**Accuracy (%)**
1e-4	7 x 7	4	50	0.5	28.95
1e-4	7 x 7	4	25	0.5	29.61
5e-5	7 x 7	4	25	0.5	29.61
1e-6	7 x 7	4	25	0.5	29.61
5e-4	7 x 7	4	25	0.5	29.61
5e-4	**7 x 7**	**8**	**25**	**0.5**	**59.21**
1e-4	**7 x 7**	**8**	**25**	**0.5**	**59.21**
1e-4	**7 x 7**	**8**	**25**	**0.2**	**59.21**
1e-4	5 x 5	8	25	0.2	57.89

Table 6.11: Results for an end-to-end traditional ANN. For these experiments the AdaMax optimizer was used. The values of the kernel size, batch size, and dropout probability were set to 7 x 7, 8, and 0.2 respectively. Each network was trained during 25 epochs.

Learning Rate	**Accuracy (%)**	**Comments**
1e-5	46.71	
5e-4	57.89	
1e-4	55.92	
1e-3	54.61	
1e-3	**53.95**	**w/ batch normalization**
5e-4	**53.95**	**w/ batch normalization**
1e-4	51.97	w/ batch normalization

Table 6.12: Results for an end-to-end SNN. For these experiments the AdaMax optimizer was used. The values of the kernel size, and batch size were set to 7 x 7, 8, and 0.2 respectively.

Learning Rate	**Epochs**	**beta**	**T**	**Max prob. spike**	**Accuracy (%)**
1e-6	25	0.75	50	0.8	44.74
1e-6	25	0.85	50	0.8	44.08
1e-6	25	0.75	75	0.8	44.74
1e-6	25	0.75	75	1	44.74
5e-6	**9**	**0.75**	**50**	**1**	**46.05**

Table 6.14: Inference time comparison between a SNN- and ANN-based convolutional DECOLLE. For the SNN architecture we present the overall result and the mean inference value for each timestep. The inference time is higher in the SNN network, arguably due to the nature of the snnTorch framework.

Network	**Inference Time (ms)**
ANN conv. DECOLLE	**1.231**
SNN conv. DECOLLE	32.732
SNN conv. DECOLLE (timestep mean)	1.347

Table 6.13: Results for the implementation of the convolutional DECOLLE strategy with an additional convolutional block in the last hidden layer, similar to the strategy adopted in the neuromorphic dataset.

Kernel Size	**T**	**Accuracy (%)**
7 + 7	**50**	**48.68**
7 + 3	50	38.16
7 + 5	50	44.08
7 + 7	75	47.37

As we can see, using SNN-based convolutional DECOLLE we are able to achieve similar performances to the ones obtained through traditional ANNs and, in particular, DECOLLE-based traditional ANNs. Nonetheless, from figure 6.4 we can see that the SNN-based convolutional DECOLLE presents a less entropic weight distribution in all hidden layers, which we argue that may contribute to the extraction of less discriminant features (figure 6.5) and, consequently lower performance comparing to the ANN-based approach. Furthermore, the use of different pseudo-targets for the two approaches as well as the different codification may also contribute for the gap in performance, with the poisson encoding possibly leading to information loss in the implementation of the SNN-based convolutional DECOLLE. However, these are promising results that suggest we could leverage the properties of LiDAR data through SNN-based efficient architectures. Nonetheless, the results presented in this work should be seen as preliminary since other approaches could be more precise and there are still many unanswered questions in this regard. For example, a more class-balanced dataset and with more samples should be discussed as well as different encoding methods for the point cloud data could be explored. In fact, since the assessed dataset (and real-world data) present different object scales, the encoding method may not be ideal once it creates very sparse images when discretizing lower scale objects, which may imply loss of information and detail. Furthermore, the optimum SNN model architecture for point cloud data is yet to be proposed.

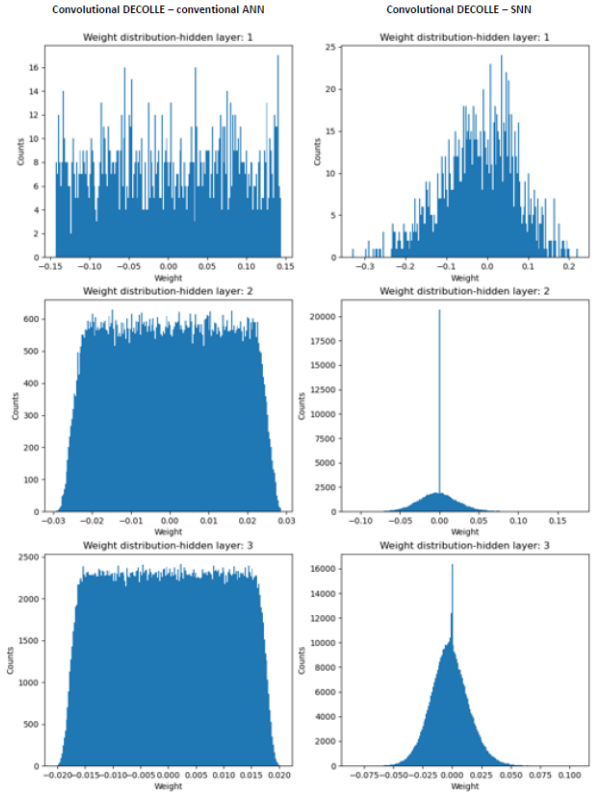

Figure 6.4: Hidden layer weight distributions for the SNN- and ANN-based convolutional DECOLLE approaches. As we can see, the SNN-based approach presents less entropic distribution, arguably decreasing the model's performance.

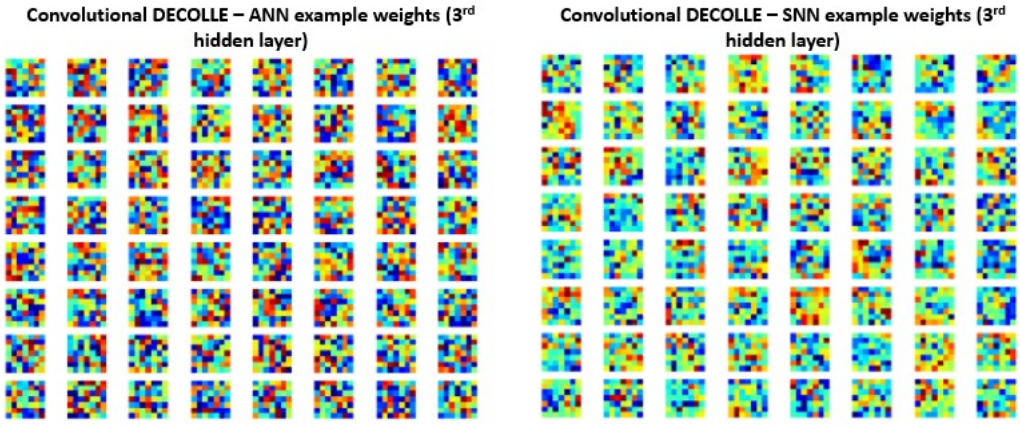

Figure 6.5: Examples of the weights of the third hidden layer for both SNN- and ANN-based convolutional DECOLLE strategies, where we can see the less discriminant features for the SNN-based implementation that may be caused by the lower variance in the weight distribution.

Chapter 7

Conclusions and Future Work

Autonomous driving vehicles' development is becoming a major focus of several companies, as it provides a safer and more comfortable alternative to traditional driving systems. Nonetheless, the interconnection and smooth communication between the different modules must be ensured, to avoid possible fatal accidents. The perception module constitutes the first and most important module that allows the system to understand and interpret the surrounding environment and pass the information for the subsequent modules. An error or a simple delay in this module is propagated through the whole pipeline and can result in severe accidents, with a fast and accurate response, thus being mandatory for the perception task of the system. Nowadays, the SOTA results for object classification in the autonomous driving context (and in other contexts as well), are presented through the use of CNN models. However, SNN came as an efficient alternative to those models, through biologically plausible dynamics. Their efficiency and low computational power need are suitable features for the autonomous driving task. Additionally, neuromorphic sensors came to increase even more the efficiency of these networks, removing the need to encode information into spike-trains and removing redundancy through the creation of event-based data. However, some of the features that give SNNs advantages over CNNs, are also the source of some major drawbacks, such as the non-differentiable nature of the neurons' activation functions, that don't allow the direct use of the BP algorithm. Although several workarounds have been proposed in the literature, papers frequently fail to report the full range of hyperparameters used, which, given SNNs' sensitivity to hyperparameters, specially considering surrogate gradients, leads to difficulties in the reproducibility of the approaches.

To overcome the aforementioned reproducibility issue, a two-layered network with the use of the STDP learning rule and implementation of lateral inhibition was implemented as proposed in the work of Diehl and Cook [25]. This work was studied on the simple MNIST and N-MNIST datasets to study its scalability to neuromorphic data. A SOM approach was also studied in both datasets in order to assess the influence of guided and flexible inhibition mechanisms in the learning of representative filters. Furthermore, a supervised learning approach based on FB

alignment was also studied as proposed by Kaiser et al. [57]. In this approach, for each layer there is a pseudo-target that allows to compute the local errors resorting to fixed random FB weights, without the need to use the forward weight matrices to propagate the error through the network. Although this network is not as biologically plausible as the previous approaches, it showed better scalability and robustness to the more complex neuromorphic data.

In conclusion, the supervised learning approaches were applied to the more challenging LiDAR data, in order to study the robustness and applicability in real-world scenarios. Even though the results were far from the current SOTA results using CNN-based architectures, this work presents relevant results in point cloud data, showing the promising nature of SNNs in the autonomous driving context.

We have seen that a lot of work has been developed towards achieving ever-more biologically plausible neural networks, even though there are several mechanisms in biological networks that are not fully understood, which leads SNNs to achieve lower performances in comparison to traditional ANNs. As for future directions in line with this work, we can point the possibility to analyze different and deeper networks, the use of different pseudo-targets in the DECOLLE implementation and different, more biologically plausible neuron dynamics. Plenty of work has to be done regarding LiDAR data, starting with different encoding techniques and possibly more robust algorithms, with the ultimate goal to implement such models in neuromorphic hardware and evaluate them in real-world scenarios in an ADS. Furthermore, as 3D point clouds can not be fully described through 2D images, the development of an ensemble model that classifies the data based on multiple views is also seen as a promising future improvement. Another promising direction is related to the adaptation methods that are currently applied for traditional neural networks for SNNs, such as data fusion implementations, taking advantage of both RGB and LiDAR data. Some preliminary work and developments should be done in the classification task in a first instance. Nonetheless, detection and segmentation tasks can be further analysed once SNN-based approaches present nearly SOTA performances, with the ultimate goal being the development of an end-to-end algorithm, able to detect, segment and classify real-world objects.

References

[1] World Health Organization. *Global status report on road safety 2018*. World Health Organization, Geneva, 2018.

[2] Laura Eboli, Gabriella Mazzulla, and Giuseppe Pungillo. The influence of physical and emotional factors on driving style of car drivers: A survey design. *Travel Behaviour and Society*, 7:43–51, April 2017.

[3] Katia M'bailara, Thierry Atzeni, Benjamin Contrand, Cyrielle Derguy, Manuel-Pierre Bouvard, Emmanuel Lagarde, and Cédric Galéra. Emotional reactivity: Beware its involvement in traffic accidents. *Psychiatry Research*, 262:290–294, April 2018.

[4] Martina Smorti and Silvia Guarnieri. Do aggressive driving and negative emotional driving mediate the link between impulsiveness and risky driving among young Italian drivers? *The Journal of Social Psychology*, 156(6):669–673, November 2016.

[5] The 6 Levels of Vehicle Autonomy Explained | Synopsys Automotive.

[6] Tesla. tesla autopilot. https://www.tesla.com/autopilot, 2022. [online; accessed 25-may-2022].

[7] Waymo. Safety report waymo. technical report, 2020.

[8] Johannes C. Thiele, Olivier Bichler, and Antoine Dupret. Event-Based, Timescale Invariant Unsupervised Online Deep Learning With STDP. *Frontiers in Computational Neuroscience*, 12:46, June 2018.

[9] Mingyuan Meng, Xingyu Yang, Shanlin Xiao, and Zhiyi Yu. Spiking Inception Module for Multi-layer Unsupervised Spiking Neural Networks. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, July 2020. arXiv:2001.10696 [cs, q-bio].

[10] Joao D. Nunes, Marcelo Carvalho, Diogo Carneiro, and Jaime S. Cardoso. Spiking Neural Networks: A Survey. *IEEE Access*, 10:60738–60764, 2022.

[11] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133, December 1943.

[12] Henry J. Kelley. Gradient Theory of Optimal Flight Paths. *ARS Journal*, 30(10):947–954, October 1960.

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, page 1097–1105, Red Hook, NY, USA, 2012. Curran Associates Inc.

[14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, Miami, FL, June 2009. IEEE.

[15] Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, Ce Liu, Mengchen Liu, Zicheng Liu, Yumao Lu, Yu Shi, Lijuan Wang, Jianfeng Wang, Bin Xiao, Zhen Xiao, Jianwei Yang, Michael Zeng, Luowei Zhou, and Pengchuan Zhang. Florence: A New Foundation Model for Computer Vision. Technical Report arXiv:2111.11432, arXiv, November 2021. arXiv:2111.11432 [cs] type: article.

[16] Neil C. Thompson, Kristjan Greenewald, Keeheon Lee, and Gabriel F. Manso. Deep Learning's Diminishing Returns: The Cost of Improvement is Becoming Unsustainable. *IEEE Spectrum*, 58(10):50–55, October 2021.

[17] 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*, 117(4):500–544, August 1952.

[18] Lyle Long and Guoliang Fang. A Review of Biologically Plausible Neuron Models for Spiking Neural Networks. In *AIAA Infotech@Aerospace 2010*, Atlanta, Georgia, April 2010. American Institute of Aeronautics and Astronautics.

[19] E.M. Izhikevich. Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, 14(6):1569–1572, November 2003.

[20] Wulfram Gerstner and Werner M. Kistler. *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge University Press, 1 edition, August 2002.

[21] Sangya Dutta, Vinay Kumar, Aditya Shukla, Nihar R. Mohapatra, and Udayan Ganguly. Leaky Integrate and Fire Neuron by Charge-Discharge Dynamics in Floating-Body MOSFET. *Scientific Reports*, 7(1):8257, December 2017.

[22] Wikipedia contributors. Hodgkin–huxley model — Wikipedia, the free encyclopedia, 2022.

[23] Arnaud Delorme, Laurent Perrinet, and Simon J. Thorpe. Networks of integrate-and-fire neurons using Rank Order Coding B: Spike timing dependent plasticity and emergence of orientation selectivity. *Neurocomputing*, 38-40:539–545, June 2001.

[24] Wei Zhang and David J. Linden. The other side of the engram: experience-driven changes in neuronal intrinsic excitability. *Nature Reviews Neuroscience*, 4(11):885–900, November 2003.

[25] Peter U. Diehl and Matthew Cook. Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Frontiers in Computational Neuroscience*, 9, August 2015.

[26] David Holmes. Reconstructing the retina. *Nature*, 561(7721):S2–S3, September 2018.

[27] Guang Chen, Hu Cao, Jorg Conradt, Huajin Tang, Florian Rohrbein, and Alois Knoll. Event-Based Neuromorphic Vision for Autonomous Driving: A Paradigm Shift for Bio-Inspired Visual Sensing and Perception. *IEEE Signal Processing Magazine*, 37(4):34–49, July 2020.

[28] Ralph Nelson and Victoria Connaughton. Bipolar cell pathways in the vertebrate retina. *Webvision: The Organization of the Retina and Visual System [Internet]*, 2012.

[29] Liqun Luo. Architectures of neuronal circuits. *Science*, 373(6559):eabg7285, September 2021.

[30] Holger Awater, Jess R. Kerlin, Karla K. Evans, and Frank Tong. Cortical Representation of Space Around the Blind Spot. *Journal of Neurophysiology*, 94(5):3314–3324, November 2005.

[31] Lisa Roux and György Buzsáki. Tasks for inhibitory interneurons in intact brain circuits. *Neuropharmacology*, 88:10–23, January 2015.

[32] Thomas Serre. Hierarchical models of the visual system. *Encyclopedia of computational neuroscience*, 6:1–12, 2014.

[33] Pierre Falez, Pierre Tirilly, Ioan Marius Bilasco, Philippe Devienne, and Pierre Boulet. Multi-layered Spiking Neural Network with Target Timestamp Threshold Adaptation and STDP. *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2019. Publisher: arXiv Version Number: 1.

[34] Ruthvik Vaila, John Chiasson, and Vishal Saxena. Deep Convolutional Spiking Neural Networks for Image Classification. 2019. Publisher: arXiv Version Number: 2.

[35] Dermot Kerr, T Martin McGinnity, Sonya A Coleman, Qingxiang Wu, and Marine Clogenson. Spiking hierarchical neural network for corner detection. In *International Conference on Neural Computation Theory and Applications*, pages 230–235. SciTePress, 2011.

[36] Frank Michler, Thomas Wachtler, and Reinhard Eckhorn. Adaptive Feedback Inhibition Improves Pattern Discrimination Learning. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Dough Tygar, Moshe Y. Vardi, Gerhard Weikum, Friedhelm Schwenker, and Simone Marinai, editors, *Artificial Neural Networks in Pattern Recognition*, volume 4087, pages 21–32. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. Series Title: Lecture Notes in Computer Science.

[37] Qiang Fu and Hongbin Dong. An ensemble unsupervised spiking neural network for objective recognition. *Neurocomputing*, 419:47–58, January 2021.

[38] Timor Kadir and Michael Brady. Saliency, scale and image description. *International Journal of Computer Vision*, 45(2):83–105, 2001.

[39] Plinio Moreno López. *Object component models using Gabor filters for visual recognition*. PhD thesis, Citeseer, 2008.

[40] Y Morales, R Nuñez, J Suarez, and C Torres. Digital tool for detecting diabetic retinopathy in retinography image using gabor transform. *Journal of Physics: Conference Series*, 792:012083, January 2017.

[41] Daniel Auge, Julian Hille, Etienne Mueller, and Alois Knoll. A Survey of Encoding Techniques for Signal Processing in Spiking Neural Networks. *Neural Processing Letters*, 53(6):4693–4710, December 2021.

[42] E. D. Adrian and Yngve Zotterman. The impulses produced by sensory nerve endings: Part 3. Impulses set up by Touch and Pressure. *The Journal of Physiology*, 61(4):465–483, August 1926.

[43] Roger M. Enoka and Jacques Duchateau. Rate Coding and the Control of Muscle Force. *Cold Spring Harbor Perspectives in Medicine*, 7(10):a029702, October 2017.

[44] Yuwei Wang, Yi Zeng, Jianbo Tang, and Bo Xu. Biological Neuron Coding Inspired Binary Word Embeddings. *Cognitive Computation*, 11(5):676–684, October 2019.

[45] Simon Thorpe, Denis Fize, and Catherine Marlot. Speed of processing in the human visual system. *Nature*, 381(6582):520–522, June 1996.

[46] Seongsik Park, Seijoon Kim, Byunggook Na, and Sungroh Yoon. T2FSNN: Deep Spiking Neural Networks with Time-to-first-spike Coding. 2020. Publisher: arXiv Version Number: 1.

[47] Seongsik Park, Seijoon Kim, Hyeokjun Choe, and Sungroh Yoon. Fast and Efficient Information Transmission with Burst Spikes in Deep Spiking Neural Networks. 2018. Publisher: arXiv Version Number: 2.

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

[49] Timothy P. Lillicrap, Daniel Cownden, Douglas B. Tweed, and Colin J. Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, 7(1):13276, December 2016.

[50] Bodo Rueckauer, Iulia-Alexandra Lungu, Yuhuang Hu, Michael Pfeiffer, and Shih-Chii Liu. Conversion of Continuous-Valued Deep Networks to Efficient Event-Driven Networks for Image Classification. *Frontiers in Neuroscience*, 11:682, December 2017.

[51] Guo-qiang Bi and Mu-ming Poo. Synaptic Modifications in Cultured Hippocampal Neurons: Dependence on Spike Timing, Synaptic Strength, and Postsynaptic Cell Type. *The Journal of Neuroscience*, 18(24):10464–10472, December 1998.

[52] Timothée Masquelier, Rudy Guyonneau, and Simon J. Thorpe. Spike Timing Dependent Plasticity Finds the Start of Repeating Patterns in Continuous Spike Trains. *PLoS ONE*, 3(1):e1377, January 2008.

[53] J.-P. Pfister. Triplets of Spikes in a Model of Spike Timing-Dependent Plasticity. *Journal of Neuroscience*, 26(38):9673–9682, September 2006.

[54] Hananel Hazan, Daniel Saunders, Darpan T. Sanghavi, Hava Siegelmann, and Robert Kozma. Unsupervised Learning with Self-Organizing Spiking Neural Networks. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6, Rio de Janeiro, July 2018. IEEE.

[55] Joseph M. Brader, Walter Senn, and Stefano Fusi. Learning Real-World Stimuli in a Neural Network with Spike-Driven Synaptic Dynamics. *Neural Computation*, 19(11):2881–2912, November 2007.

[56] Dongcheng Zhao, Yi Zeng, Tielin Zhang, Mengting Shi, and Feifei Zhao. GLSNN: A Multi-Layer Spiking Neural Network Based on Global Feedback Alignment and Local STDP Plasticity. *Frontiers in Computational Neuroscience*, 14:576841, November 2020.

[57] Jacques Kaiser, Hesham Mostafa, and Emre Neftci. Synaptic Plasticity Dynamics for Deep Continuous Local Learning (DECOLLE). *Frontiers in Neuroscience*, 14:424, May 2020.

[58] Friedemann Zenke and Surya Ganguli. SuperSpike: Supervised Learning in Multilayer Spiking Neural Networks. *Neural Computation*, 30(6):1514–1541, June 2018.

[59] Chankyu Lee, Syed Shakib Sarwar, Priyadarshini Panda, Gopalakrishnan Srinivasan, and Kaushik Roy. Enabling Spike-Based Backpropagation for Training Deep Neural Network Architectures. *Frontiers in Neuroscience*, 14:119, February 2020.

[60] N. Izeboudjen, C. Larbes, and A. Farah. A new classification approach for neural networks hardware: from standards chips to embedded systems on chip. *Artificial Intelligence Review*, 41(4):491–534, April 2014.

[61] Catherine D. Schuman, Thomas E. Potok, Robert M. Patton, J. Douglas Birdwell, Mark E. Dean, Garrett S. Rose, and James S. Plank. A Survey of Neuromorphic Computing and Neural Networks in Hardware. 2017. Publisher: arXiv Version Number: 1.

[62] Nico Messikommer, Daniel Gehrig, Antonio Loquercio, and Davide Scaramuzza. Event-based Asynchronous Sparse Convolutional Networks. 2020. Publisher: arXiv Version Number: 2.

[63] Patrick Lichtsteiner, Christoph Posch, and Tobi Delbruck. A 128 x 128 120 dB 15 µs Latency Asynchronous Temporal Contrast Vision Sensor. *IEEE Journal of Solid-State Circuits*, 43(2):566–576, 2008.

[64] Christoph Posch, Daniel Matolin, and Rainer Wohlgenannt. A QVGA 143dB dynamic range asynchronous address-event PWM dynamic image sensor with lossless pixel-level video compression. In *2010 IEEE International Solid-State Circuits Conference - (ISSCC)*, pages 400–401, San Francisco, CA, USA, February 2010. IEEE.

[65] Christian Brandli, Raphael Berner, Minhao Yang, Shih-Chii Liu, and Tobi Delbruck. A 240 × 180 130 dB 3 µs Latency Global Shutter Spatiotemporal Vision Sensor. *IEEE Journal of Solid-State Circuits*, 49(10):2333–2341, October 2014.

[66] David Reverter Valeiras, Garrick Orchard, Sio-Hoi Ieng, and Ryad B. Benosman. Neuromorphic Event-Based 3D Pose Estimation. *Frontiers in Neuroscience*, 9, January 2016.

[67] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.

[68] Garrick Orchard, Ajinkya Jayawant, Gregory K. Cohen, and Nitish Thakor. Converting Static Image Datasets to Spiking Neuromorphic Datasets Using Saccades. *Frontiers in Neuroscience*, 9, November 2015.

[69] Teresa Serrano-Gotarredona and Bernabé Linares-Barranco. Poker-DVS and MNIST-DVS. Their History, How They Were Made, and Other Details. *Frontiers in Neuroscience*, 9, December 2015.

[70] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, Canadian Institute for Advanced Research, 2009.

[71] Hongmin Li, Hanchao Liu, Xiangyang Ji, Guoqi Li, and Luping Shi. CIFAR10-DVS: An Event-Stream Dataset for Object Classification. *Frontiers in Neuroscience*, 11:309, May 2017.

[72] Wensheng Cheng, Hao Luo, Wen Yang, Lei Yu, Shoushun Chen, and Wei Li. DET: A High-Resolution DVS Dataset for Lane Extraction. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1666–1675, Long Beach, CA, USA, June 2019. IEEE.

[73] Amos Sironi, Manuele Brambilla, Nicolas Bourdis, Xavier Lagorce, and Ryad Benosman. HATS: Histograms of Averaged Time Surfaces for Robust Event-based Object Classification. 2018. Publisher: arXiv Version Number: 1.

[74] Alex Zihao Zhu, Dinesh Thakur, Tolga Ozaslan, Bernd Pfrommer, Vijay Kumar, and Kostas Daniilidis. The Multi Vehicle Stereo Event Camera Dataset: An Event Camera Dataset for 3D Perception. 2018. Publisher: arXiv Version Number: 2.

[75] Jonathan Binas, Daniel Neil, Shih-Chii Liu, and Tobi Delbruck. DDD17: End-To-End DAVIS Driving Dataset. 2017. Publisher: arXiv Version Number: 1.

[76] LiDAR 3D Perception and Object Detection, March 2022.

[77] Demetrios Gatziolis and Hans-Erik. Andersen. A guide to LIDAR data acquisition and processing for the forests of the Pacific Northwest. Technical Report PNW-GTR-768, U.S. Department of Agriculture, Forest Service, Pacific Northwest Research Station, Portland, OR, 2008.

[78] Zhen Wang, Liqiang Zhang, Tian Fang, P. Takis Mathiopoulos, Xiaohua Tong, Huamin Qu, Zhiqiang Xiao, Fang Li, and Dong Chen. A Multiscale and Hierarchical Feature Extraction Method for Terrestrial Laser Scanning Point Cloud Classification. *IEEE Transactions on Geoscience and Remote Sensing*, 53(5):2409–2425, May 2015.

[79] Jeff Hecht. Lidar for self-driving cars. *Optics and Photonics News*, 29(1):26–33, 2018.

[80] Jessica Rowbury. Driving innovation. *Electro Optics*, (296):3–4, 2019.

[81] Anh Nguyen and Bac Le. 3D point cloud segmentation: A survey. In *2013 6th IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, pages 225–230, Manila, Philippines, November 2013. IEEE.

[82] Charles R. Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. Frustum PointNets for 3D Object Detection from RGB-D Data. 2017. Publisher: arXiv Version Number: 2.

[83] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. 2016. Publisher: arXiv Version Number: 2.

[84] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. 2017. Publisher: arXiv Version Number: 1.

[85] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. PointCNN: Convolution On X-Transformed Points. 2018. Publisher: arXiv Version Number: 5.

[86] Wei Wang, Shibo Zhou, Jingxi Li, Xiaohua Li, Junsong Yuan, and Zhanpeng Jin. Temporal Pulses Driven Spiking Neural Network for Time and Power Efficient Object Recognition in Autonomous Driving. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 6359–6366, Milan, Italy, January 2021. IEEE.

[87] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. 2015. Publisher: arXiv Version Number: 1.

[88] David Griffiths and Jan Boehm. A review on deep learning techniques for 3D sensed data classification. 2019. Publisher: arXiv Version Number: 1.

[89] Sindhu Hegde and Shankar Gangisetty. PIG-Net: Inception based Deep Learning Architecture for 3D Point Cloud Segmentation. 2021. Publisher: arXiv Version Number: 1.

[90] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361, Providence, RI, June 2012. IEEE.

[91] Jean-Emmanuel Deschaud. KITTI-CARLA: a KITTI-like dataset generated by CARLA Simulator. 2021. Publisher: arXiv Version Number: 1.

[92] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Sheng Zhao, Shuyang Cheng, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in Perception for Autonomous Driving: Waymo Open Dataset. 2019. Publisher: arXiv Version Number: 7.

[93] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving. 2019. Publisher: arXiv Version Number: 5.

[94] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3D ShapeNets: A Deep Representation for Volumetric Shapes. 2014. Publisher: arXiv Version Number: 3.

[95] Jiachen Sun, Qingzhao Zhang, Bhavya Kailkhura, Zhiding Yu, Chaowei Xiao, and Z. Morley Mao. Benchmarking Robustness of 3D Point Cloud Recognition Against Common Corruptions. 2022. Publisher: arXiv Version Number: 1.

[96] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Duc Thanh Nguyen, and Sai-Kit Yeung. Revisiting Point Cloud Classification: A New Benchmark Dataset and Classification Model on Real-World Data. 2019. Publisher: arXiv Version Number: 2.

[97] Shi Qiu, Saeed Anwar, and Nick Barnes. Geometric Back-Projection Network for Point Cloud Classification. *IEEE Transactions on Multimedia*, 24:1943–1955, 2022.

[98] Mark De Deuge, Alastair Quadros, Calvin Hung, and Bertrand Douillard. Unsupervised feature learning for classification of outdoor 3d scans. In *Australasian conference on robitics and automation*, volume 2, page 1. University of New South Wales Kensington, Australia, 2013.

[99] Xi Yang, Ding Xia, Taichi Kin, and Takeo Igarashi. IntrA: 3D Intracranial Aneurysm Dataset for Deep Learning. 2020. Publisher: arXiv Version Number: 2.

[100] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep Learning for 3D Point Clouds: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(12):4338–4364, December 2021.

[101] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view Convolutional Neural Networks for 3D Shape Recognition. 2015. Publisher: arXiv Version Number: 3.

[102] Ze Yang and Liwei Wang. Learning Relationships for Multi-View 3D Object Recognition. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7504–7513, Seoul, Korea (South), October 2019. IEEE.

[103] Daniel Maturana and Sebastian Scherer. VoxNet: A 3D Convolutional Neural Network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928, Hamburg, Germany, September 2015. IEEE.

[104] Truc Le and Ye Duan. PointGrid: A Deep Network for 3D Shape Understanding. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9204–9214, Salt Lake City, UT, June 2018. IEEE.

[105] Hananel Hazan, Daniel J. Saunders, Hassaan Khan, Devdhar Patel, Darpan T. Sanghavi, Hava T. Siegelmann, and Robert Kozma. BindsNET: A Machine Learning-Oriented Spiking Neural Networks Library in Python. *Frontiers in Neuroinformatics*, 12:89, December 2018.

[106] Dan Goodman. Brian: a simulator for spiking neural networks in Python. *Frontiers in Neuroinformatics*, 2, 2008.

[107] Teuvo Kohonen. *Self-organizing maps*, volume 30. Springer Science & Business Media, 2012.

[108] Arnon Amir, Brian Taba, David Berg, Timothy Melano, Jeffrey McKinstry, Carmelo Di Nolfo, Tapan Nayak, Alexander Andreopoulos, Guillaume Garreau, Marcela Mendoza, Jeff Kusnitz, Michael Debole, Steve Esser, Tobi Delbruck, Myron Flickner, and Dharmendra Modha. A Low Power, Fully Event-Based Gesture Recognition System. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7388–7397, Honolulu, HI, July 2017. IEEE.

[109] Jason K. Eshraghian, Max Ward, Emre Neftci, Xinxin Wang, Gregor Lenz, Girish Dwivedi, Mohammed Bennamoun, Doo Seok Jeong, and Wei D. Lu. Training Spiking Neural Networks Using Lessons From Deep Learning. 2021. Publisher: arXiv Version Number: 4.

[110] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[111] Bing Han and Kaushik Roy. Deep Spiking Neural Network: Energy Efficiency Through Time Based Coding. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, volume 12355, pages 388–404. Springer International Publishing, Cham, 2020. Series Title: Lecture Notes in Computer Science.

[112] Piotr Opiełka, Janusz T. Starczewski, Michał Wróbel, Katarzyna Nieszporek, and Alina Marchlewska. Application of Spiking Neural Networks to Fashion Classification. In Leszek Rutkowski, Rafał Scherer, Marcin Korytkowski, Witold Pedrycz, Ryszard Tadeusiewicz, and Jacek M. Zurada, editors, *Artificial Intelligence and Soft Computing*, volume 11508, pages 172–180. Springer International Publishing, Cham, 2019. Series Title: Lecture Notes in Computer Science.

[113] Shihui Yin, Shreyas K. Venkataramanaiah, Gregory K. Chen, Ram Krishnamurthy, Yu Cao, Chaitali Chakrabarti, and Jae-sun Seo. Algorithm and hardware design of discrete-time spiking neural networks based on back propagation with binary activations. In *2017 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pages 1–5, Torino, October 2017. IEEE.

[114] Jianhao Ding, Zhaofei Yu, Yonghong Tian, and Tiejun Huang. Optimal ANN-SNN Conversion for Fast and Accurate Inference in Deep Spiking Neural Networks. 2021. Publisher: arXiv Version Number: 1.

[115] Yang Li and Yi Zeng. Efficient and Accurate Conversion of Spiking Neural Network with Burst Spikes. 2022. Publisher: arXiv Version Number: 2.

[116] Nguyen-Dong Ho and Ik-Joon Chang. TCL: an ANN-to-SNN Conversion with Trainable Clipping Layers. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pages 793–798, San Francisco, CA, USA, December 2021. IEEE.