Environmental Sound Recognition with Spiking Neural Network

Thesis · July 2021

DOI: 10.13140/RG.2.2.19614.46401

5 authors, including:

Taki Hasan Rafi
Hanyang University
15 PUBLICATIONS **21** CITATIONS
SEE PROFILE

Tashfi Nowroz
Ahsanullah University of Science & Tech
1 PUBLICATION **0** CITATIONS
SEE PROFILE

Sayma Saera Rahman
Ahsanullah University of Science & Tech
1 PUBLICATION **0** CITATIONS
SEE PROFILE

Nadim Khandakar
Ahsanullah University of Science & Tech
1 PUBLICATION **0** CITATIONS
SEE PROFILE

ENVIRONMENTAL SOUND RECOGNITION WITH SPIKING NEURAL NETWORK

A thesis
Submitted in partial fulfillment of the requirements for the Degree of
Bachelor of Science in Electrical and Electronic Engineering

Submitted By,
Taki Hasan Rafi
Sayed MD Tashfi Nowroz
Sayma Saera Rahman
Nadim Khandakar

Under the Supervision of,
Mr. Monjur Morshed
Associate Professor
Department of Electrical and Electronic Engineering
Ahsanullah University of Science and Technology

Department of Electrical and Electronic Engineering
Ahsanullah University of Science and Technology
Dhaka, Bangladesh

Spring 2020

CERTIFICATE

This is to certify that the thesis titled ”**Environmental Sound Recognition with Spiking Neural Network** is a bonafide record of the work done by,

Sayed MD Tashfi Nowroz	**160205081**
Sayma Saera Rahman	**160205096**
Taki Hasan Rafi	**160205135**
Nadim Khandakar	**160205161**

under my supervision and guidance in partial fulfillment of the requirements for the award of the degree of **Bachelor of Science** in **Electrical and Electronic Engineering** of the **Ahsanullah University of Science and Technology, Dhaka, Bangladesh**, during the year Fall 2019 - Spring 2020.

Their work is genuine and has not been submitted for the award of any other degree to the best of my knowledge.

DATE:

Mr. Monjur Morshed
Associate Professor
Department of Electrical and Electronic Engineering
Ahsanullah University of Science and Technology

DECLARATION

This is to certify that the work which is being hereby presented by us in this thesis titled "**Environmental Sound Recognition with Spiking Neural Network**" in partial fulfilment of the award of the Bachelor of Science submitted at the Department of Electrical and Electronic Engineering, Ahsanullah University of Science and Technology, Dhaka, Bangladesh is a genuine account of our work carried out during the period from Fall 2019 to Spring 2020 under the guidance of **Mr. Monjur Morshed**, Department of Electrical and Electronic Engineering, Ahsanullah University of Science and Technology, Dhaka, Bangladesh .

The matter embodied in the thesis to the best of our knowledge has not been submitted for the award of any other degree elsewhere.

DATE:

Sayed MD Tashfi Nowroz (160205081)

Sayma Saera Rahman (160205096)

Taki Hasan Rafi (160205135)

Nadim Khandakar (160205161)

ACKNOWLEDGEMENT

We recognize being fortunate enough to have completed this academic thesis whilst in sound health. In the beginning, we express our gratitude to our parents, in whose patronage we could come this far. Most importantly, we thank our supervisor Mr. Monjur Morshed, whose constant guidance proved invaluable. We recognize the constructive pieces of advice from him which shaped our thesis from crude to fine. We further recognize our departmental head Prof. Dr. A K M Baki for inspirations. Lastly, we want to dedicate our thesis to our mothers for their enormous love, support and everything.

DATE:

Taki Hasan Rafi

Sayed MD Tashfi Nowroz

Sayma Saera Rahman

Nadim Khandakar

ABSTRACT

Environmental sound recognition has achieving tremendous attention in past few years. Audio signal classification which includes speech and music signal classification have a traditional impact on signal classification research. Extensive research on environmental sound recognition has been expanded immensely in recent years of signal classification. This thesis focuses on mainly training a spike-based model for environmental sound recognition. At first, we have extracted features using mel spectrogram to feed into our models. We have trained covolutional neural network model on the dataset for initial observation. The trained CNN model has achieved 90.21% accuracy on the test data. We have trained spike-based SNN model via CNN model with the help of nengo dl framework. Our developed spiking neural model has achieved 74.55% accuracy. We also have deployed spike aware, rate regularization and low pass filtering method to improve the performance. But in our experiment, all these methods have decreased the performance by 16% - 51%. Which are not expected. Improving the performance of spike-based models can be a potential research area for further advanced research.

TABLE OF CONTENTS

List of Tables

List of Figures

Chapter 1

Introduction

1.1 What is Environmental Sound

Environmental sounds are getting more and more popular in research both theoretical and methodological which cross the border in various sector. To efficiently develop a productive multifunctional dataset on environmental sounds and apply it in various research field to recognize environmental sound, a deeper understanding of the essence behind environmental sounds, what they mean for humans, factors in environmental sound perception, and how their perception can vary or be identical for different listeners is needed [1].

Environmental sounds are the auditory experience we get from variant sources around us. There are mainly two sources of environmental sound. The first source is nature and the second source is humans. Environmental sound perception is very essential from an ecological perspective. It gives listeners information about objects and events in their immediate environment. It can act as an auditory warning for immediate danger. High pitched sounds like car horns, baby cries, birds chirping causes irritation. This irritation alerts the listeners to immediate danger as well as contributes to one's sense of awareness and wellbeing. It can also act as a soothing mechanism for stressed-out minds. Normal sounds of nature like the crashing of sea waves, the rustling of tree leaves, chirping of melodious birds like a nightingale, light rain sounds can soothe a stressful mind. But excessive sounds can cause the opposite effect. Too much noise can cause stress, irritation, and hamper mental health. The auditory disturbance caused by excess sound is very harmful to mental health. Most stressful noises are man-made. The major culprit among these stressful noises is traffic noise. An analysis shows that general traffic noise is the main reason for auditory disturbance and mental health deterioration. The average city sound level is 60 decibels. Research has shown that these excessive sound or noise as we can call it, increases blood pressure, heart rate, and causes hearing loss [1-2].

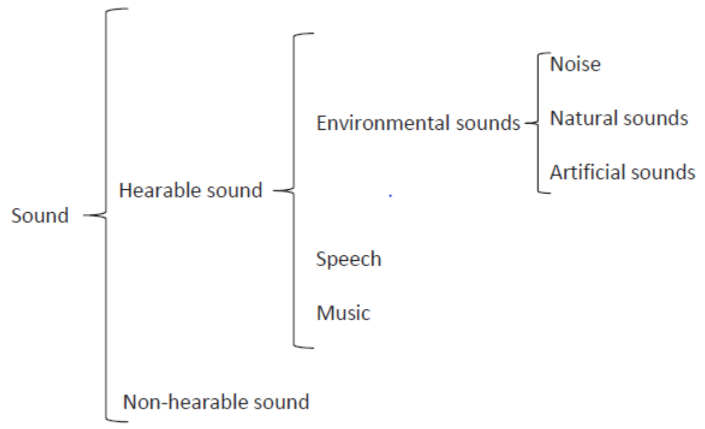


Figure 1.1: Classification of Sound and Environmental Sound.

1.2 Applications of Environmental Sound Recognition

Environmental sounds are worth researching due to their desire to express semantic meaning dependent on dynamic acoustic structure. Environmental sound recognition is a vital capability of a person to quickly gather data from the encompassing environment, the accomplishment of that can lead to inciting developments to be taken before potential possibilities or perils. Such as, if an individual detects a sound of gunfire or bombing, an individual may sense danger without perception and find a safer place. Animals can sense that too. So, Human beings as well as distinctive animals are excellent at recognizing environmental sounds. This remarkable potential has inspired more efforts being dedicated to equipping artificial structures with a comparable potential for environmental sound recognition (ESR) [12]. Environmental sound recognition has gained attention in the last few years, not just because of its research pertinence in solving unpredictable and non-structured data complexities, and also because of its contribution to a variety of innovations, including soundtracks, reconnaissance, and regular hearing aids.

ESR is used in many ways. For example, for home automation, monitoring the elderly people who stay alone at home, danger detection, speech, and music recognition. In any kind of fire outbreak, anyone can detect danger using a fire alarm, thus by using a successful ESR system one can detect fire alarm sound and sends a signal to the fire brigade and police to rescue., ESR is beneficial in day-to-day life, but currently, unstructured sound data are challenging to process in signal for evolving ESR systems [25].

An indispensable dimension of intelligence is hearing. Even though with the absence of associated visual signals, human beings are quite capable of understanding and responding to sounds events. For implementations that include sound monitoring and reconnaissance, natural resource management, Wildlife hazard detection, different devices that detect problems and create awareness, and smart home systems, detection and recognition of sound events can be utilized. The diverse existence of the correlation between the characterizations and detection of auditory stimuli of sound events in both time-frequency domains and these sound tags tends to make it difficult to deal with a technically constructed sound activity recognition solution. Deep learning techniques and algorithms appear to enhance during that phase. Deep Neural Networks (DNN) is used by a dynamic neuronal layer structure to instantly extract a time-varying nonlinear interface of the system [12].

Looking at the overview of environmental hazards, using automatic environmental sound recognition could be a better solution to mitigate wildlife problems. If we look at the recent incident of 2020, that is the fire outbreak at amazon. Between January to august there were a series of fire outbreak occurred on Amazon. Over the same period, 6,315 explosions of fire were reported inside the Amazon. This is not only affecting the wildlife but also creating problems for human life as well. The main purpose of our experiment and research is to find a robust solution for this type of adverse environmental problem by detecting and identifying the sound hazards. Deep Neural Networks are currently used in many sectors, such as the main motivation of our work is to mitigate environmental perils by early detection of sound from the environment. The efficiency of our embedding DNN model is considerably enhanced by integrating the deep neural network and the convolution neural network and respective results, implying the representation of our proposed. We already know deep learning has vast applications of image recognition, video processing and recognition, barcode scanning, fraud detection, sound signal classification, and detection. The main purpose of environmental sound recognition is to detect environmental sound more precisely, it will help us to identify the unwanted jeopardized occurrence in the environment. Most likely, Amazon fire outbreak and mitigating the growing wildlife perils and any kind of environmental endangerment. So, we need a robust deep learning solution to mitigate this problem that has higher accuracy in terms of classification. Our process is resilient to extreme sound stimuli and can analyze complex environmental sound signals that are constant which illustrates our proposed system's feasibility [14-15].

1.3 Computer-Aided Environmental Sound Recognition

Sound signals are surprisingly affordable to encode and store data compared to visionbased signals, which helps to get higher efficiency and low energy consumption. Recently a lot of research work has been done for Environmental sound recognition, research analysts have been extensively analyzed sound data in their initial research works. However, their analogy is a bit outdated, as many new machine learning methods have now developed. It is, therefore, necessary to evaluate sound classification performance with the current machine learning models. Deep Convolutional neural networks with an elevated model capacity are especially dependent on the accessibility of a large volume of sound datasets to learn a non-linear feature from input to output that is well distributed and produces better classification precision on distinctive datasets of sound or audio [15].

The process of deep convolutional neural networks (CNN) to recognize clusters of classification of sound spikes makes it a great choice for the analysis of environmental sound. These are well accustomed to environmental sound classification due to the capacity of deep convolutional neural networks (CNN) to learn different biased spike patterns. The sheer quantity of annotated data, however, has hindered the implementation of this elevated framework [15]. In recognition of environmental sound, computing a Deep Convolutional Neural network model for environmental sound classification can be a better solution. Utilizing environmental sound data augmentation to resolve the data constraint issue and analyze the effect of various increases on the efficiency of the proposed Convolutional Neural Network classification algorithm [12] Different types of successful approaches were made for environmental sound recognition. Deep neural networks with a set of models called deep learning are now flourishing in different classification methods in the past few years with remarkable effects of more precise and efficient output for sound recognition and classification. Billions of neurons in human brains function in the system of a human body. They take signals from any part of the body and send the signal back to the brain to function accordingly. Deep neural network and Convolutional neural network works like the human brain. Signals of human brains are like spikes, which are mostly like electrical signals. These discrete electrical signals can be classified by different computational deep learning methods. In our research, we would like to get the environmental sound signals and convert them as spike signals which act as electrical signals. To deeper comprehend the fundamental computation and modeling principles of deep neural networks, various learning algorithms have been proposed and developed for sound recognition. In our work, we have used deep neural network algorithms and frameworks to detect adverse environmental sound precisely and with better accuracy to resolve environmental hazards [14-15].

1.4 Organization of This Thesis

The chapter 2 discusses about related works. Chapter 3 discusses about theoretical background on different deep learning schemes. Chapter 4 demonstrates the description about Spiking Neural Network. Chapter 5 discusses about the experimental methodology. And chapter 6 shows the results and outcomes of this experiment.

Chapter 2

Related Works

2.1 Literature Review

2.1.1 Conventional Machine Learning Algorithms ESR

To the best of our understanding, Cowling and Sitte [11] made the first attempt to compare different ESR strategies in 2003. Cowling's work conducts further research into ambient non-speech sounds including their path, with the aim of incorporating them in an autonomous mobile surveillance robot [12]. In a recent work, Burak [13] has introduced a 2D functionality package that used SVMs and neural RBF networks and compared the classification accuracies of the features proposed with MFCCs. An extensive survey is provided on latest developments in the ESR area [14].

Another promising work done by Bountourakis et al. [10]. They tried to make a comparison of the most frequently used ESR approaches. They tested six machine learning algorithms (including K-NN, SVM, ANN, Naive Bayes, Logistic Regression, Decision Tree) along with three combined various feature sets and their result was promising.

Salamon et al. [7] proposed an algorithm to forming sound data function vectors in the field of sound recording can be constructed by using Mel-spectrogram. The lengths of each sound must be identical with any sound. The sound gets divided into several units as well as the sound, derived from the Melfrequency cepstral coefficients (MFCCs), is used as a function to produce features of the same scale. These characteristics are used to calculate a melanoma spectrum. In the area of machine learning to train suitable models, these feature vectors may be used. The functions extracted from the librosa library, a Python package for audio and music processing can differ according to the parameter setting in Python. The audio is mounted as a NumPy sequence into the software for processing and manipulation. The models in [7-8] that motivated these studies require use of MFCC data. MFCCs are widely used in ambient sound processing (contains new works [9]) as a competitive standard to benchmark novel methods.

2.1.2 Traditional Deep Learning Algorithms on ESR

Environmental sound detection has been a very difficult task to achieve due to noisy signals that are hardly always identical and scarcity of available labeled resources. Not a lot of research had been done on the topic until deep learning techniques became popular. The best performing technique till then has been non-stationary Continuous Wavelet Transform together with Dynamic Time Warping [32].

Li et al. [33] used the *IEEE challenge on Detection and classification of Acoustic Scenes and Events* dataset to compare all the state-of-the-art deep learning techniques for the purpose. Six sets of feature extraction techniques utilized DNN, RNN CNN architectures. The accuracy of the architectures are 80.2%, 84.2% 82.2% concluding that none of the techniques outperform one another. However, it was observed that the performance can vary significantly with the proper combination of feature extraction.

Mushtaq et al. [34] experimented with all CNN architectures using spectral images of environmental sound samples by a meaningful data augmentation approach. The study was conducted on three well-known datasets - ESC-10, ESC-50, and Us8k. This technique was specifically designed for the augmentation of sound data by using spectrum diagrams. It is named meaningful data augmentation because the added augmentation has physical meaning in terms of sound. The approach seemed to make the trained models a lot more robust and versatile. They achieved 99.49% accuracy with ResNet-152 for the Us8k dataset.

Abdoli et al. [35] used the UrbanSound8k dataset to experiment with audio signals directly in 1D instead of converting them to 2D spectrograms using different architectures of CNN. The technique can process any length of audio as it splits the given audio into frames of specific length and learns directly from the audio signals that utilize the advantage of fine time structure from the raw audio data. Their proposed 1D CNN achieves an average accuracy of 89%, which is higher than any previously mentioned standalone CNNs.

Zhang et al. [36] observed that the built-in small convolutional filters learn very little when applied to ESC data as it needs many layers to achieve capabilities to be able to learn long contextual information which matters. Also, the ReLU function results in 50% information loss despite it making the training process significantly faster by a few times. Based on these observations, they proposed a "Dilated Convolution Neural Network" (D-CNN) with a LeakyReLU function that tackles these problems. This approach achieved 2%-10% higher accuracy than state-of-the-arts on the 3 most popular datasets: UrbanSound8k, ESC50 CICESC.

Wang and Chong [37] attempted to reduce the computational complexity of CNN by reducing the parameters involved. Their proposed neural network achieved 85% accuracy with 183,094 parameters which are significant in terms of efficiency. Their approach offers a reduction of parameters by 24.16% and MAC operations by 20.17%.

2.1.3 Hybrid Deep Learning Approaches on Sound Recognition

While many empirical approaches and deep learning approaches are common in the ESR task, hybrid deep learning approaches for ESR are rare. A method Convolutional Recurrent Neural Networks or CRNN was proposed in [62] that utilizes CNN to grasp localized features and LSTM to utilize temporal summarization over the given audio and found strong performance considering the sum of parameters and time requirement of training. A hybrid model was proposed by [47] which combined utilized Optimum Allocation Sampling (OAS), Short-Time Fourier-Transform (STFT), Convolutional Neural Networks (CNN), and 6 different classification techniques to combat 'no free launch theorem'. Their approach achieved 95.8% accuracy and shows promised of being utilized as a real-time environmental sound detection tool.

Chapter 3

Theoretical Background

3.1 Environmental Sound Classification Problem

The scientific community has increasingly given interest in recent times to the issue of automated environmental sound classification. Its various applications ranging from context-based computation and monitoring to mitigated noise across intelligent acoustic sensor networks [15-17]. Currently there have been various methods of signal processing and machine-learning for the problem, such as matrix factorization, wavelet-filtering, dictionary learning and very recent times, deep neural networks [15, 18, 19]. In fact, CNNs are, in theory, very suitable for the issue of environmental sound classification [31]. First, when added to spectrum-like inputs that have demonstrated to be an essential feature for discriminating amongst sound like engines and jackhammers, they are capable of recording energy modulation trends through time and frequency [19]. Second, the network should learn successfully and then identify Spectro time designs that represent different sound classes by using a convolutional kernel (filters) with a small receiving area. However, some sound is masked (in time/frequency) by other sources (noise), which are where conventional audio features such as Mel-Frequency Cepstral coefficient (MFCC) are failed. However, CNNs are limited to the classification of environmentally sound. In obtaining equivalent outcomes, for example, the CNN suggested the results of dictionary learning (which may be called an instance of "shallow" attribute learning), Yet things didn't get better [15].

Deep learning models with high model potential rely mainly on the availability of large numbers of training data and learning a non-linear function from input to output, which also generalizes and results in high accuracy of classification in unseen data. The relative lack of labeled data for the environmental sound classification is a potential reason for the minimal development of CNNs and the inability to improve basic model designs. During recent years, some new data sets have been published but they are significantly smaller than, for instance, data sets for analysis on the classification of images [15].

The problem of sound detection consists of three separate phases including signal pre-processing, extraction of unique characteristics and classification. The pre-processing signal splits the input signal into several segments used to extract similar characteristics. The extraction method reduces data size and represents dynamic data as vectors. A variety of different classification systems have been used for crossing rates, pitch and frame functionality for speech recognition applications such as decision trees, random woodlands and k-NN, SIF, SAI, LPC have been widely used. In addition, various machine learning and neural networks methods such as the Hidden and Gaussian model, random forestry, a multilayered perceptron and evolving deep learning networks have resulted in improved sound recognition and classification systems efficiency [16].

3.2 Supervised Learning

The supervised algorithms for machine learning are algorithms which require external assistance. A concise model of the class label distribution in terms of predictor characteristics is intended as supervised learning. The dataset is divided into a dataset of trains and tests. The train data set has a projected or graded performance element. Both algorithms learn those patterns and apply them for inference or classification in the test dataset [4-21]. Figure 3.1 demonstrates the process of supervised machine-learning algorithms. Three most popular supervised algorithms are mentioned here.

- **Decision Tree:** Decision trees are certain kinds of trees that identify characteristics by grouping them depending upon their values. The Decision Tree is used primarily for the purpose of classification. The nodes and branches are composed

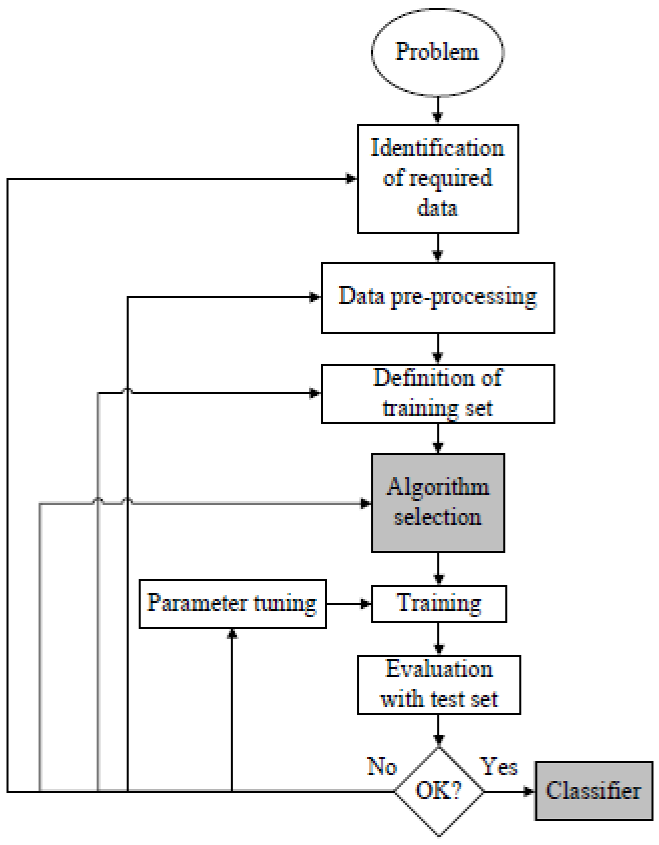


Figure 3.1: The process of supervised ML [21].

of each tree. In a decision tree, each node represents a function in an instance to be categorized, and each branch represents a value that can be assumed by the node [21]. Figure 3.2 is an example of a decision tree.

- **Naïve Bayes:** Naive Bayes is a commonly used system of classification that is based on the principle of Bayes. Due to its simplicity, efficiency, and accuracy, Naive Bayes models have become very common for classification and clustering. The fundamental structure of Naïve Bayes relies on the conditional possibility. such trees are Often known as the Bayesian Network [4].

- **Support Vector Machine:** Support Vector Machine (SMV) is another popularly used machine learning technique. The Support Vector Machines algorithm aims to find the optimum maximizing the margin of data point preparation. If data is not linearly separated, then data instances get to be linearly separable by using kernel techniques into higher n-dimensional characteristics [22]. SVM operates

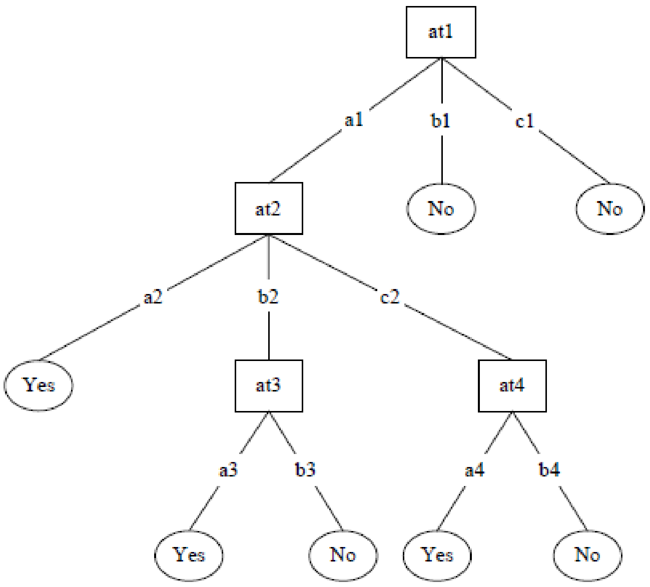


Figure 3.2: A decision tree [21].

on margin estimation theory. The boundaries are drawn under such a manner that the distance in between boundary and the groups is optimum and, thus, the classification error is reduced [4]. Figure 3.3 demonstrates the working of SMV.

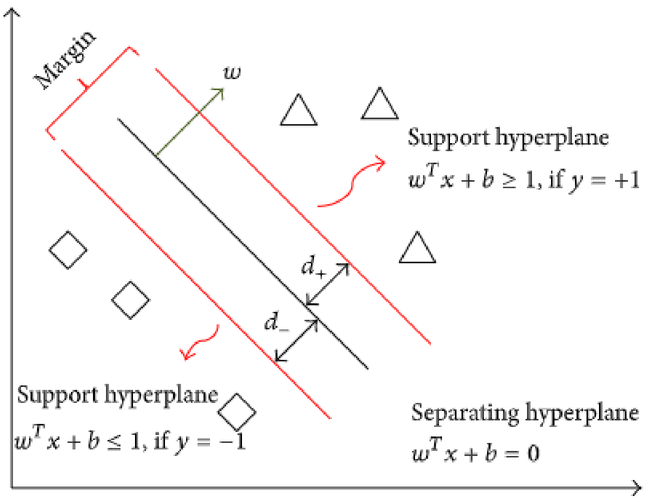


Figure 3.3: Working of Support Vector Machine.

3.3 Unsupervised Learning

This are considered unsupervised lessons since there are no right responses because there is no instructor, compared supervised learning before. It uses the previously learned features to identify the class of the data as new data is assigned. In order to identify and present the fascinating structure in the results, algorithms are leave to their own ideas. The algorithms of unsupervised learning learn few characteristics from the results. It uses the comprehension features to identify the class of the data as new data is added. It is used mostly to cluster and minimize functionality [20]. Figure 3.4 demonstrate the working of unsupervised learning.

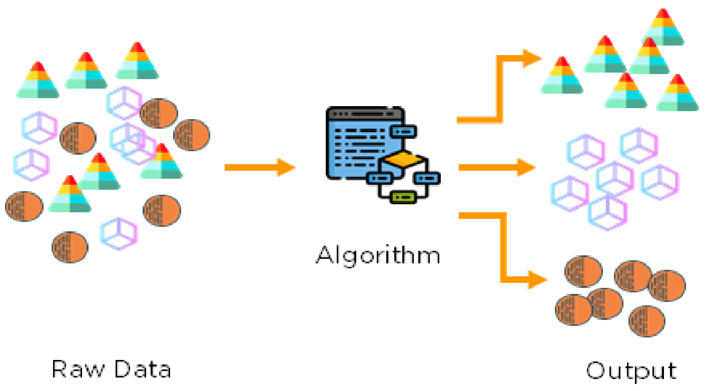


Figure 3.4: Example of Unsupervised Learning.

- **K-Means Clustering:** K-Means is among the most conveniently unregulated algorithms to solve the well-known problem of clustering. The protocol follows a straightforward and simple way of classifying a certain data set by means of a certain number of a priori clusters (assume k clusters). It's an unsupervised type of learning [3-5]. Items with identical features are grouped in the same cluster. The algorithm is k-means since it generates various k-clusters. The middle of this cluster is the mean of the cluster values [4]. In the figure 3.5, a clustered data is demonstrated.

- **Principal Component Analysis:** The data dimension is minimized in the principal part of the analysis or PCA to make it easier and faster to measure. Taking a 2D data illustration to illustrate how PCA functions. When this data is graphically displayed, 2 axes are taken. Apply Principal Component Analysis to data, then the data is 1D [4].

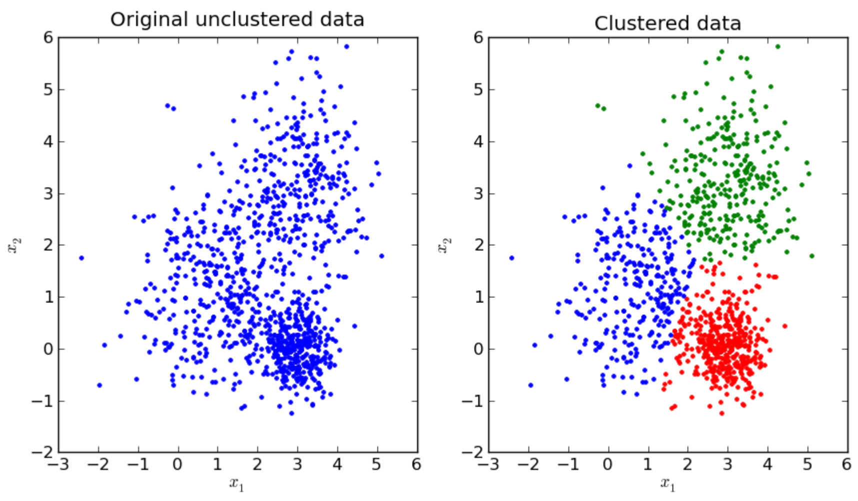


Figure 3.5: K-Means Clustering [4].

3.4 Reinforcement Learning

Reinforcement learning requires learning from conditions and behavior to optimize a scalar benefit or strengthening signal. As in most aspects of machine learning, the learner also isn't told what actions are to take but must instead explore which actions produce the greatest reward by attempting them. In most fascinating and difficult situations, actions can not only affect the immediate reward, but can also affect the next scenario [23]. Figure 3.6 illustrates the general model of reinforcement learning.

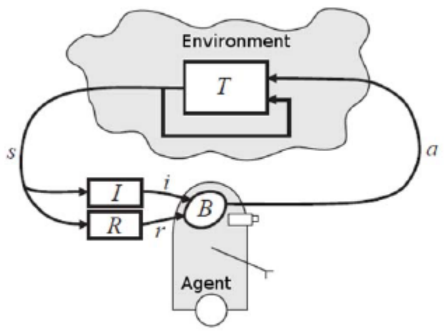


Figure 3.6: Reinforcement Learning Model [4].

The system receives from the environment the input I current states, status change r and input function I. Dependent on the inputs, the agent produces a behavior B and acts to achieve a response.

3.5 Neural Networks

The idea of ANN derives primarily from the biological area in which neural networks play a significant and central important role in the body. The neural network is used to function in the human body. Neural Network is really just a network of millions and millions of connected neurons. You have to understand how a neuron acts in order to understand the neural network. A neuron primarily has four components (see figure.9). They are dendrites, soma, axon and nucleus.

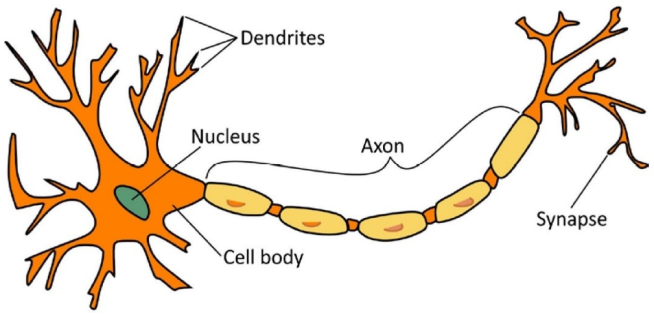


Figure 3.7: Human Neuron.

Electrical impulses are delivered to the dendrites. Soma interprets electrical signal. The axon follows the output to the dendrites' ends where the output is passed to the next neuron. The neuron relation is called the neural network by means of which electrical signals disperse throughout the brain. The same is the case with an artificial neural network. It runs on 3 levels. Join the input layers. The cache continues with the entry. Finally the measured output is sent to the output layer (dendrite terminals) [4-6]. Basically, the artificial neural network comprises of three types: supervised, unsupervised and reinforcement. Figure 3.8 illustrates the structure of ANN.

- **Supervised Neural Network:** There is no previous indication of the output of the data on the neural network. The network's main role is, to identify the data corresponding to such similarities. The neural network monitors the relation between the various classes and inputs [4]. Figure 3.9 demonstrates the process.
- **Unsupervised Neural Network:** There is no previous indication of the output of the data on the neural network. The network's key role is to identify data ac-

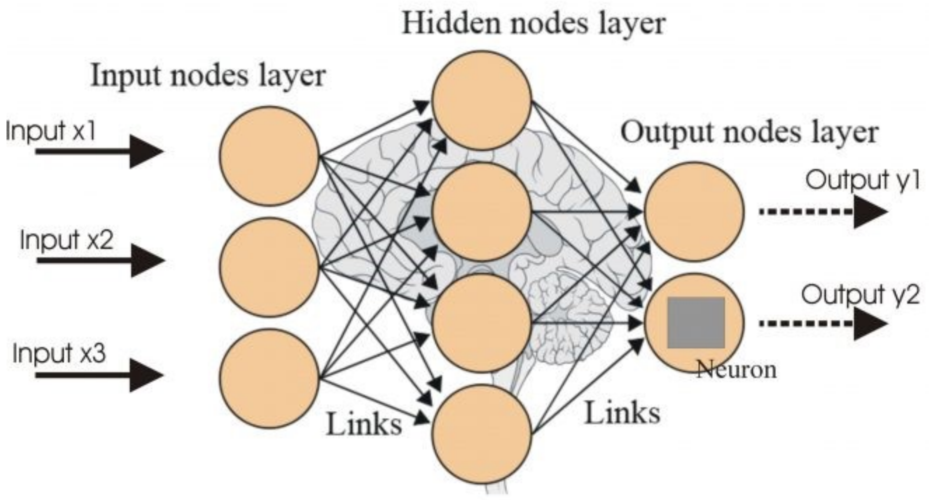


Figure 3.8: Artificial Neural Network Structure.

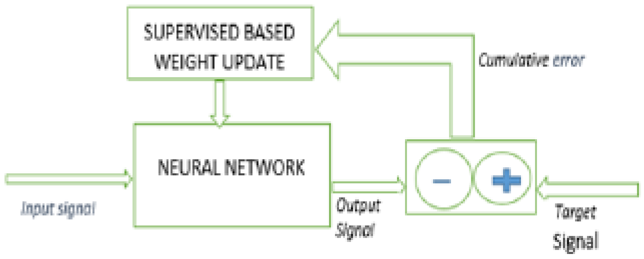


Figure 3.9: Supervised Neural Network.

cording to such similarities. The neural network tracks the association between different inputs and classes [4]. Figure 3.10 demonstrates the schematic diagram.

- **Reinforced Neural Network:** This network functions as if a person communicates with the environment through a stronger neural network. The network collected environmental feedback to detect whether the network decision is right or wrong. If decision is correct, the associations showing this particular production are improved. The relations are otherwise debilitated. The network doesn't have any previous production information [4]. Figure 3.11 demonstrates the schematic diagram.

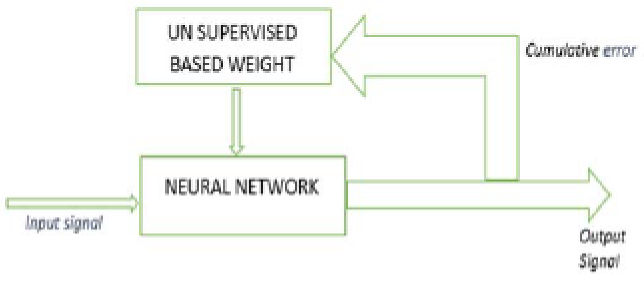


Figure 3.10: Unsupervised Neural Network.

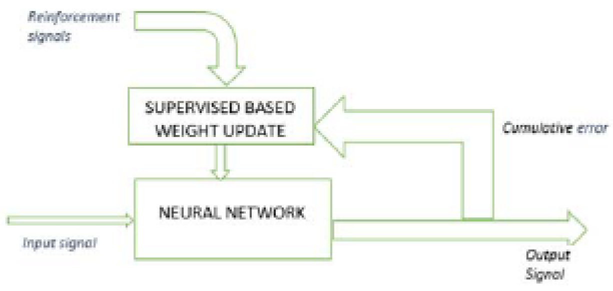


Figure 3.11: Reinforced Neural Network.

3.5.1 Artificial Neural Network

Artificial neural networks are the functional unit of deep learning based on the concept of the human brain. Deep learning uses artificial neural networks which mimic the behavior of the human brain to solve complex data-driven problems. Deep learning itself is a subset of machine learning, which fits under the larger layers of artificial intelligence. So, Artificial intelligence, Machine learning, and Deep learning are interconnected fields where Machine Learning and Deep Learning aids AI by providing a set of algorithms and neural networks to solve data-driven problems.

The artificial neural network is based on an archetype of a human brain-like neural system. It is made up of a large network associated with basic processing components known as neurons and coefficients as weights are also connected to the connections that link them with each other. Now, these neurons get input signals transformed by connecting weights, and through a process of training, the neurons get activated by transfer functions to give the desired output. These neurons now have their input signals converted through connecting weights, and they are stimulated by transfer functions via a training phase to produce the desired output.

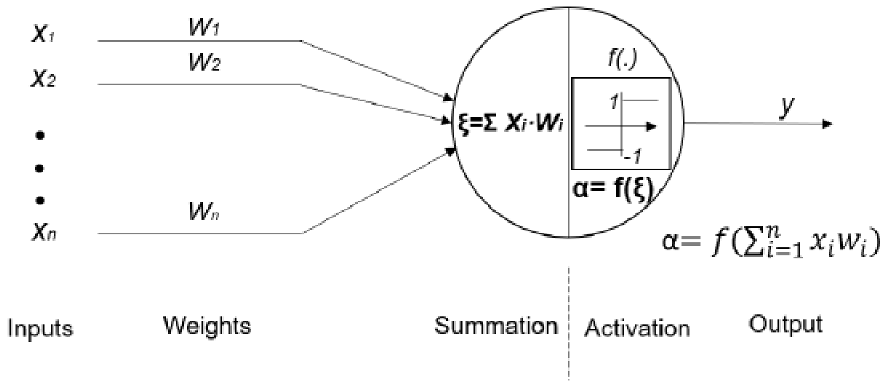


Figure 3.12: Basic Scheme of a single Neural Network [43].

A "neuron" is the fundamental foundation block of neural networks. A neuron can be thought of as a sort of central processor. Neurons in a neural network are connected by "synaptic weights," or "weights". Each neuron in a neural network generates and receives "weighted" signals from the neurons to which it is connected through these synaptic contacts, and generates an output by transferring the weighted sum of those input signals via an "activation function" [38]. The figure below shows a single neuron based neural network structure. The data processing inside a neuron starts with the inputs X_n, which are weighted and connected before proceeding through an activation function to produce the output, which is represented as $Xi.Wi$. Then the activation factor is multiplied by the particular weight Wn for each of the outgoing wires of nodes before moving forward to the next node. If a linear activation function is considered to implement, then the output is $y = (wx + b)$ [42]. Recursive weight modification is used to train a neural network to map a sequence of input data. In order to optimize the weights between neurons, data from inputs is fed forward through the network [41]. Let us consider the output as, h_i, and neurons as i in the hidden layer so the output is,

where (σ) represents the activation function, N specifies the number of input neurons, V_{ij} represents the weights, x_j represents the input neurons' inputs, and T_i^{hid} denotes the hidden neurons' threshold values. The objective of the activation function is to confine the value of the neuron so that the neural network is not paralyzed by divergent neurons, in addition to incorporating nonlinearity into the neural network. The

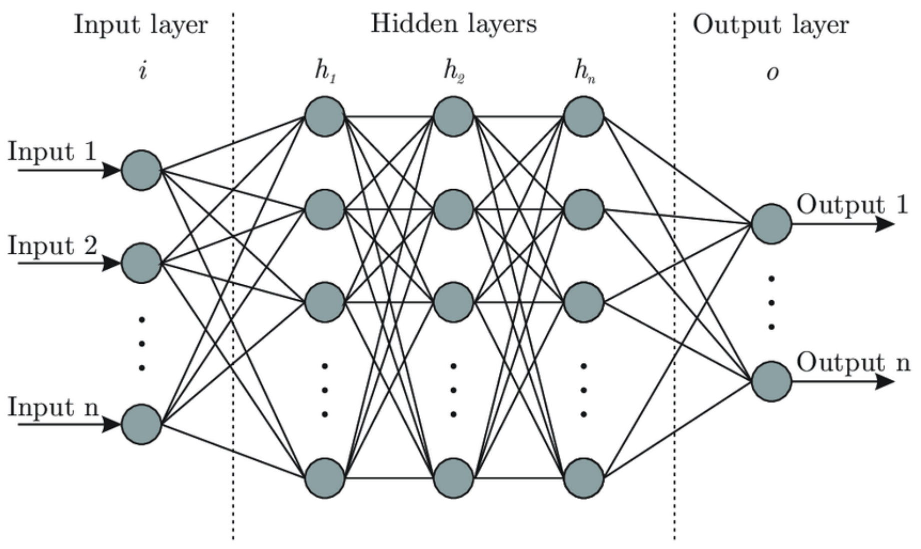


Figure 3.13: Artificial neural network architecture (ANN input $i - h_1 - h_2 - h_n$ - Output o [40].

output amplitude is regulated by an activation function. For instance, an appropriate output range is typically between 0 and 1, but it may also be between -1 and 1. The most commonly used typical activation function is the sigmoid function. Arc-tangent and hyperbolic tangent are two other potential activation functions. They have equal responses to inputs as the sigmoid function, but their output ranges are different [39]

The figure below shows the n number of input layers i_1 to i_n, n number of hidden layers h_1 to h_n, and n number of output layers o_1 to o_n .

Here the input signals are multiplied by the associated weights and then combined before being passed through a transfer function to generate the neuron's output.

Artificial neural networks (ANNs) have been effectively applied to a wide range of areas. Some of the major areas of engineering and science are using this framework. Recently, signal processing, Classification, Device modeling, pattern recognition, detection of different systems, image processing, and stock market forecasts are using ANN for getting better output performance.

3.5.2 Feed-Forward Neural Network

A Feed Forward Neural Network is a type of artificial neural network where its nodes are connected in a one-by-one format. It does not make any circles or loopback patterns. Since data is only transmitted in one direction, the feed forward model is a relatively simple type of neural network. Regardless of how many hidden nodes the data passes through, it only goes in a certain direction and never reverses. A sequence of inputs entering the layer is multiplied by the weights in the FFNN model. Then each data is summed together to get a combined input value. If the sum of the data values is greater than a predetermined threshold, which is normally fixed at zero, the output value will be typically 1, while if the sum is less than the threshold, the output value will show as -1. In Feed Forward Neural Networks, Data is transferred through a process. First

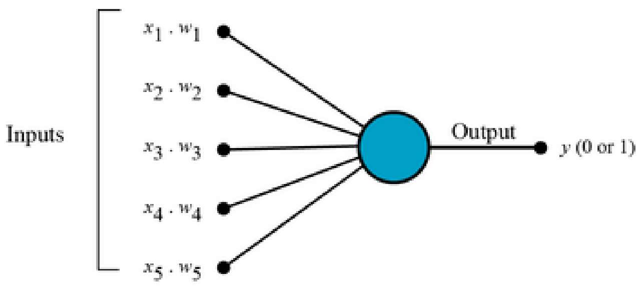


Figure 3.14: Basic Feed-Forward neural network.

input nodes to hidden nodes, then from hidden nodes moving to output nodes. Whenever an input data sequence is inserted further into the network, then the output layer modules compare with one another, with the leading output module having the input data weights that are nearest to the input sequence. The input layers and the output layers each have the identical amount of neurons as the issued inputs and outputs. The learning algorithm can be broken down into two segments: the first is identifying which weight matrix is the first input of the hidden layer and the second one is the training process. The Euclidean length between the input data and the weight matrix of the winning neurons will be determined first. If the length exceeds a specified length threshold, a new hidden layer neuron is generated, with the input serving it as a weight matrix. The input sequence, on the other hand, corresponds to this neuron. Throughout the training process, each sequence represented to the network chooses the nearest neuron on just a measurement of Euclidean distance, adjusting the forward neurons' weight matrix, and spatial nearby neurons. Then it moves them through the path of input neurons, with the gradient descent technique. This gradient descent model is done by adjusting the weights without the neuron that is in forwarding position and its neighbor neurons too [43]. Feed-forward neural networks are categorized into two

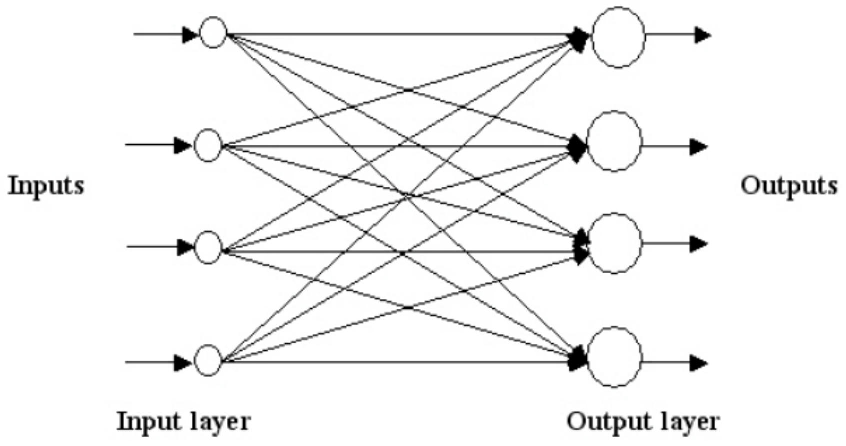


Figure 3.15: A single layer feed-forward neural network [38].

kinds based on the number of layers: “single layer” and 'multi-layer' [38].

In this figure a fully connected single-layer feed forward neural network is represented. In this configuration, there are two layers, namely input, and output layers. Though, the input layer is not considered since there's no calculation or computation is done on that layer. The signals that are generated in the input layer are routed to the output layer by a weights matrix then the neuron situated in the output layer is calculated and determined by the signals generated at the output [38]

In this figure, a multi-layer feed-forward neural network is represented with a “hidden layer”. Opposing the context of a single layer, there is at least one layer of a hidden neural network between the layers of input and output [38]. These multi-layers Feed-forward neural networks are also fully connected. If some of the synaptic connections are not considered or calculated then the network will work as “partially connected” layers.

3.5.3 Convolutional Neural Network

Convolutional Neural Network, more commonly addressed as CNN is an Artificial Neural Network that is widely known for its excellent performance in image classification. Although it is most widely used for image classification, it has also been used

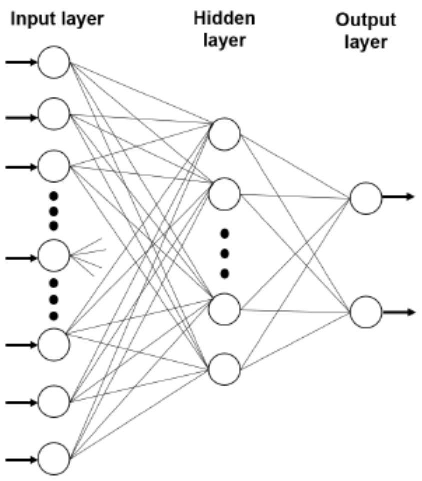


Figure 3.16: A multi-layer feed-forward neural network.

in many other classification or detection applications. The network is specialized for pattern recognition which holds major advantages in image classification problems. It is similar to multi-layer perceptrons or deep neural networks in general, however, it is composed of both convolutional layers and fully connected layers.

The CNN or Convolutional Neural Network is not an entirely new concept. It was first proposed by Yann LeCun in the 1980s. Her work was constructed upon the previous work of Kunihiko Fukushima. During that time, her CNN could detect handwritten digits. Altho the approach had gotten a niche market in the postal banking sectors, it still had many problems. Processing high-resolution images was also very difficult. CNN's are designed by mimicking how the brain processes visual information. Each

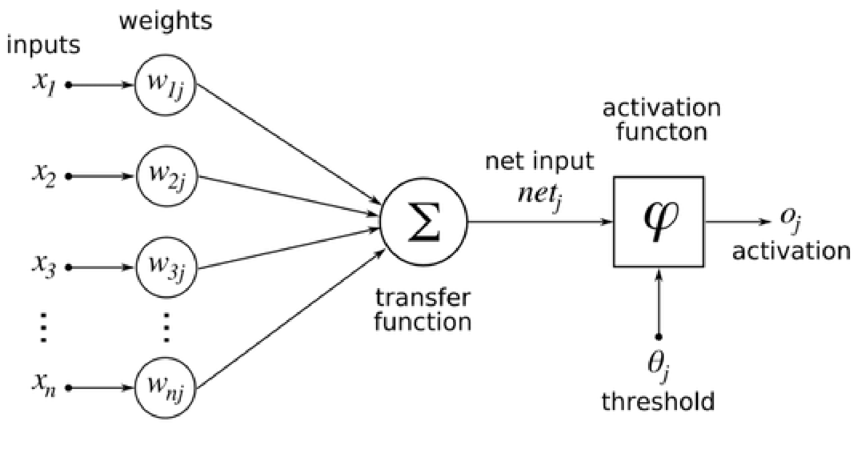


Figure 3.17: Elementary Convolutional Neural Network Model.

neuron in the neural network produces a weighted sum of multiple neurons of the previous layer outputting an activation value. After the activation function, the output matrix goes through Max-polling which is a down-sampling technique where the maximum or largest values are kept and less significant values are eliminated which results in highlighting the more visible features while ignoring the less visible ones.

The behavior of the network is defined by the weights that it fine-tunes itself through an interactive process.

Typically neural networks take all image pixel data that are fed from the input layer. This approach results in losing important image information because the original sequence of the pixels is no longer useful. Convolutional Neural Networks work differently without disrupting the spatial information of the input image. Each convolution layer of the network produces a handful of activation maps. They are possible to be visualized as a set of molds that are moved through the image pixels to check if any particular area of the image fits in or not. This set of templates or molds are known as convolutional kernels.

If any part of the input is found to be much similar to the kernel, a large positive value is returned. For less or no similarities, zero or very small values are returned.

Let us consider an image that is convoluted with a kernel that checks straight vertical lines.

The first few layers can only identify basic features like lines, simple curves, and so on. The higher layers gradually start to detect more complex features such as circles, contours, etc. At some point, the layers can detect even human faces.

However, despite such powerful performance accuracy on huge popular datasets, CNN still struggles with the detection of the same objects under different lighting angle conditions. They are also extremely resource-heavy, computationally.

3.5.4 Deep Learning

Deep learning is a feature of artificial intelligence (AI) that mimics the functioning of the brain of humans in data analysis, processing, and the creation of structures for decision-making. Deep learning is a subset of artificial intelligence and machine learning that has networks capable of learning unsupervised from unstructured data and supervised learning as well. These are known as the Deep Neural Network that works as a neuron of the brain. Deep learning works as Deep learning gathers data from various sources of data and evaluates these in near real-time, similar to how the human brain receives and analyzes information approaching the body via five senses. They analyze and re-compress data, fine-tuning the processing information and findings over time to reliably identify, define, and explain features in the data. Deep learning algorithms are also addressed as deep neural networks because most of the deep learning approaches utilize the algorithm of neural network architecture. The number of hidden layers in deep neural networks is the basic difference between traditional neural networks and deep learning. Because in traditional neural networks the number of hidden layers is almost 2 to 3 but in deep learning the number of hidden layers is as much as the consumer wants. Without any feature extraction, the training process of deep learning is done by using massive amounts of data and neural network architectures that learn features from the given data. Deep neural networks are made up of

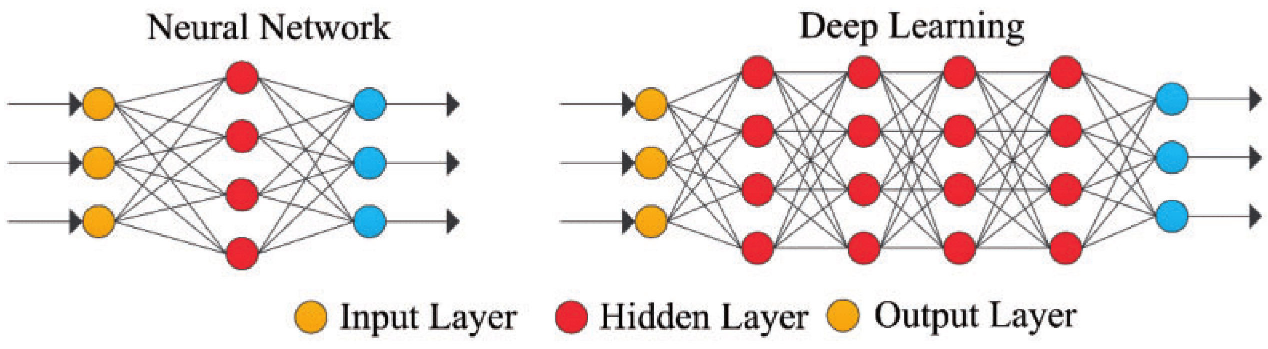


Figure 3.18: Deep Neural Network.

multiple layers of which are interconnected with weighted nodes, and each of these extracts and identifies features and structures within the data using highly complex deep learning techniques. These increasingly complex deep learning architectures are done in the hidden layer. The probability of conviction that the object or data can be labeled or categorized with one or even more forms is then assessed as needed. But the predictions can be more accurate based on the number of data given as input. Also, a higher number of layers gives even better output. So the learning process of algorithms is the most important aspect of Deep Learning. Apart from training these models, a lot of people are familiar with a variety of deep learning frameworks and how to apply them to various aspects. Caffe, TensorFlow, Torch, and Theano are some of the most popular deep learning frameworks [46].

With the rising technological period, which has contributed to volumes of data and information in all formats and from each zone of the world, deep learning has developed with fast-moving technology. This data, generally, alluded to as big data, is derived from, outlets such as social networking sites, online databases, e-commerce websites, and virtual movie shows, advertisements, medical hospitals, and so on. Deep learning, a self-adjustable versatile prediction that gets even more profound classification and correlations with practicing the existing data or with newly upgraded data. These data techniques are one of the foremost prevalent AI techniques utilized for processing big data. In this paper, we have used the deep convolutional neural network to identify different spectral and spectrotemporal patterns extracted from sound to classify environmental sound.

The resiliency nature of deep learning already led to major advances in almost all of the milestones in computer vision, disease detection, traffic control systems, robotics, self-driving vehicles, speech recognition, language comprehension, and other domains. In recent years, progress has been incremental, spreading over several years. Deep learning has been culminating benchmark developments of 20-30% annually. Deep learning is possibly the most important advancement in computer science in recent years. Its influence is already realized in nearly every field of science. Industries and markets are already being challenged and transformed as a result of it. The world's leading industries are in a battle of a race to technological advancement using Deep learning [44].

3.6 Activation Functions

The Deep Neural Networks that have shown so well performance in recognition tasks like Computer vision, Environmental Sound Recognition and so on. Such well accuracy that we currently enjoy were unthinkable back at past not far behind. Since the human brain is so efficient and good at pattern recognition, the deep neural network architectures that are used so heavily today are biologically inspired and tries to mimic the human brain.

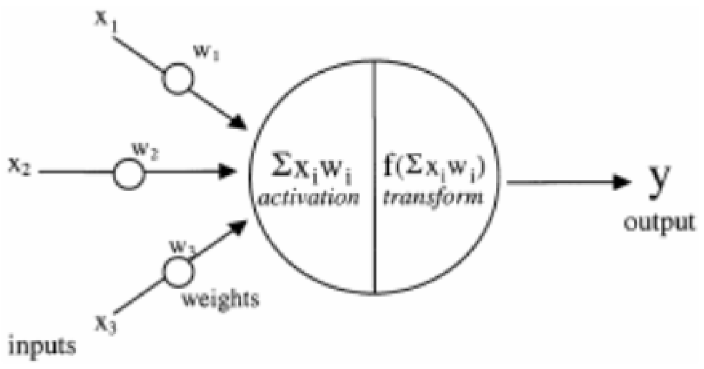


Figure 3.19: Biologically Inspired Neuron [64].

The building blocks of human brain are called neurons that are connected to other neurons. Just like that, in a neural networks, a perception similar to nodes in graph theory. These perceptron are organized in layers and are connected to all the other perceptron at the previous layer.

The perceptron in previous layer acts like a brain neuron and fire values different values which are then weighted summed. After that is done, it is decided if the neuron will itself will fire or not, depending on the input received from the previous layer. If we consider an output of a perceptron as Y, whether it will fire or not depends on the weighted sum of the inputs coming from the previous layer.

$$Y = \text{ Activation } \left(\sum(\text{ Weight } * \text{ input }) + \text{ bias }\right) \tag{3.1}$$

After the weighted sum and bias is added, it is important to decide if this perceptron is supposed to fire or not fire. As this sum can be anything without boundary, there needs to exist a function that bounds this sum and decide the firing. That function is known as activation function. The activation function is an important ingredient in the process. There are a handful of choices and the performance of a model depend on the activation function that we choose. However, it is important to note that, we must choose the activation functions depending on the layers. Input layer, hidden layers and output layers must be assigned with different activation functions because the outputs that we expect from these 3 different layers are very different from one another. Performance of a Network depends strongly upon the activation functions chosen [77]. There are countless activation functions that has been experimented with, some of them are stated below –

In our work, we will only be looking at the details of 3 most popular and well performing Activation functions known to date –

3.6.1 Sigmoid

Sigmoid function is one of the most widely used non-linear activation functions. The value of a sigmoid function always ranges from 0 to 1. This plot of the function looks like following –

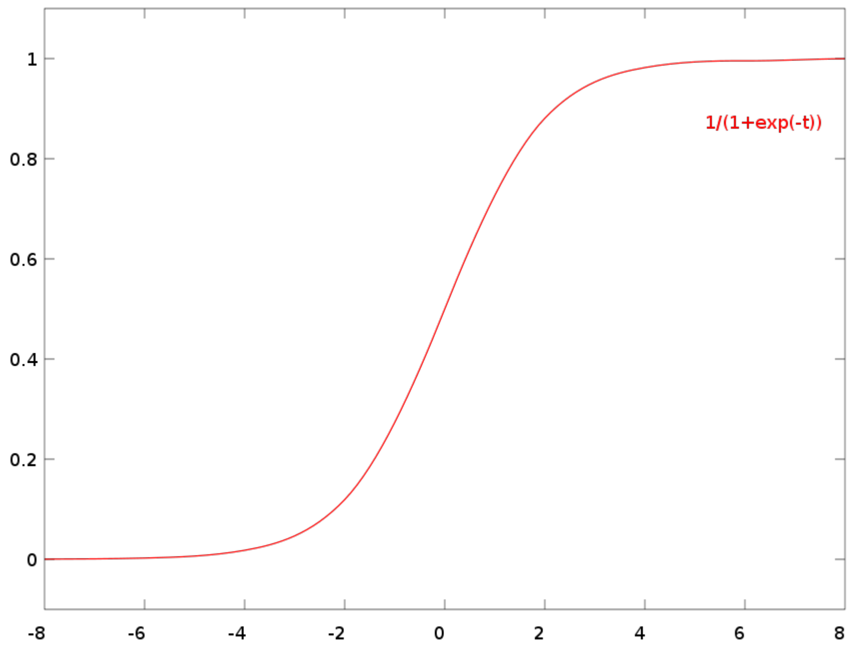


Figure 3.20: Sigmoid Function

Observing the characteristics curve, it is evident that the sigmoid activation function is very appropriate for perceptron at the output layers when used in classification problems.

$$A = \frac{1}{(1 + e^{-x})} \tag{3.2}$$

However, there are challenges with this function as well. As the values of input keeps getting higher, the different in values of you also keeps reducing. After a certain point, the output difference of two high input numbers become very small rising the vanishing gradient problem [65].

3.6.2 tanh

Tanh is another very popular non-linear activation function. The curve of tanh looks very similar to sigmoid function. However, they are not the same. Tanh is a symmetric function, right at the origin.

$$\tanh(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{3.3}$$

Tanh function is symmetric at origin and its value range from -1 to +1. Since it is wider compared to sigmoid function, therefore it is better resistant to gradient decent issues. It provides better performance in deeper Networks than Sigmoid functions [65]. Although it is better but the gradient vanishing problem still does exist.

This particular activation function has been observes to be used in Natural Language Processing and Audio Recognition Tasks [67].

3.6.3 ReLU

ReLU stands for Rectified Linear Unit. Currently ReLU is very popular for its use in the hidden layers [65]. This is not a linear function despite its name. The primary benefit of this is that it cleverly avoids the problems of vanishing gradient and is much better with time memory complexity compared to sigmoid and tanh functions [68].

$$f(x) = max(0, x) \tag{3.4}$$

For simpler computational complexity issues, using ReLU function also speeds up the model training process. The sigmoid and tanh functions are only most sensitive throughout the mid region but less near boundary values. ReLU activation function is

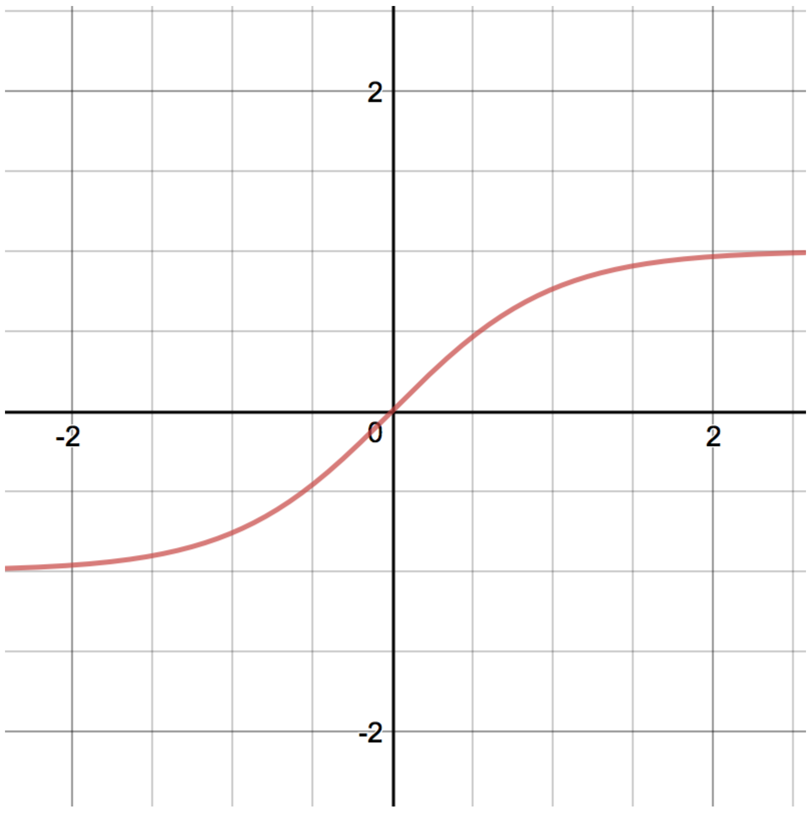


Figure 3.21: tanh Function

the most resistant to that problem because the sensitivity of it is evenly throughout the space.

3.7 Optimizers

As the name suggests, an optimizer optimizes the weight and bias tuning process while training a Network. The primary task of it is to efficiently identify the best combinations and try to find global minima. Identifying the most efficient numerical method for computation is not only an extremely important task in mathematics, but also for Machine learning as well. Optimizers respond to the loss function and fine-tune the relevant parameters to minimize the loss.

Gradient Descent is the most widely known and numerically very intuitive approach to find the global minimal efficiently. It tunes the parameters slightly and extracts insights from the output and decides again how to respond to efficiently identify the global minima. However, there are again several approaches of doing this task, each having their own advantage and disadvantages.

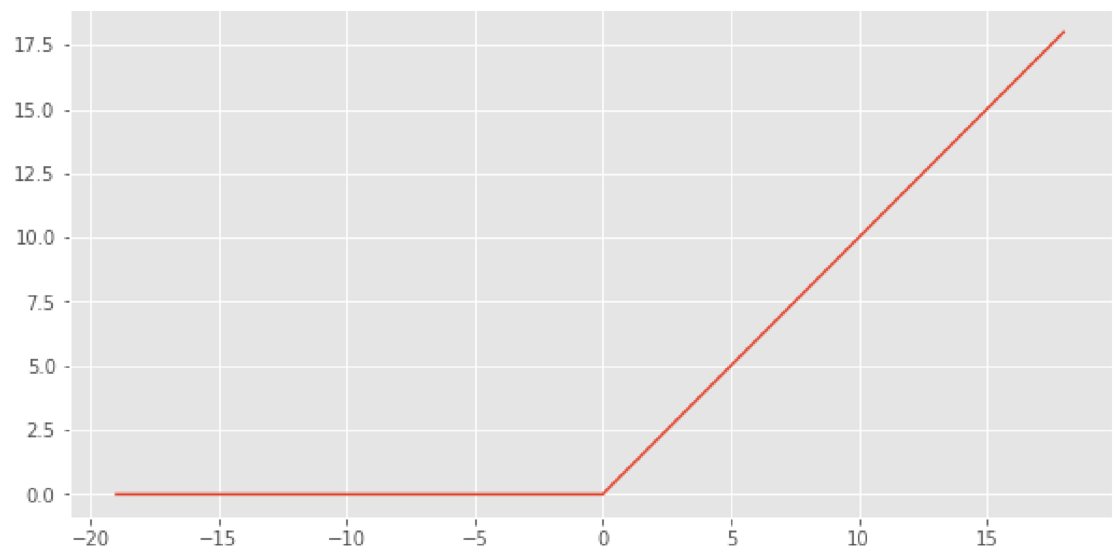


Figure 3.22: ReLU function

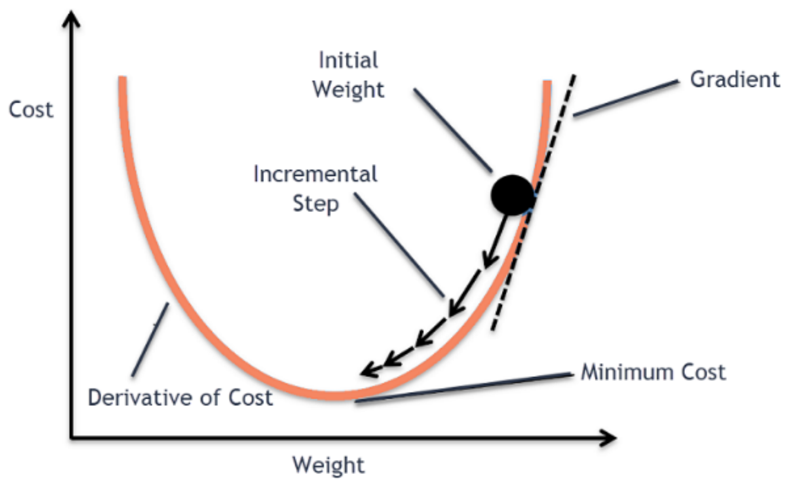


Figure 3.23: Description of Gradient Descent.

3.7.1 Batch Gradient Descent

Batch Gradient Descent is the most computationally expensive approach. It takes the entire dataset into consideration. Changes the weights and biases slightly, observing the loss function. It tried to determine the combinations which keep minimizing the loss function, thus with slow iteration eventually finds the minima. However, in large datasets the number of parameters easily exceed the practical range and it takes very long time to train models which large dataset using this approach. Another major drawback of this approach is that it easily gets drowned in local minimal.

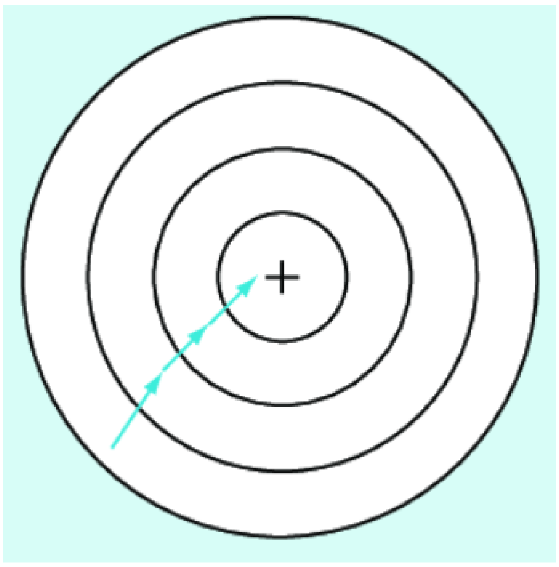


Figure 3.24: Batch Gradient Descent.

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta) \tag{3.5}$$

3.7.2 Mini-Batch Gradient Descent

Mini-Batch Gradient Descent is a mixture of both Batch Gradient Descent and Schocastic Gradient Descent. Instead of chossing parameters totally in random, mini-batch gradient approach makes small sets batches with random data and tries to find its way to minima with every iteration. It is less noisy than SDG and better with large dataset compared to the other approaches of Gradient descent.

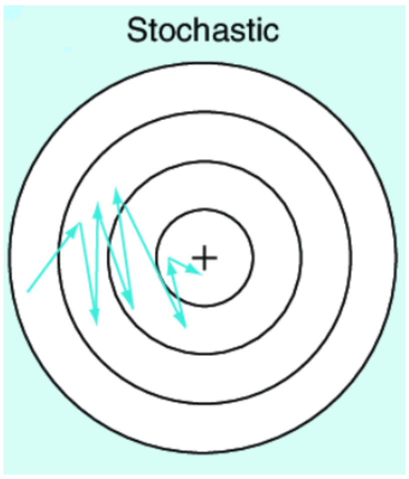


Figure 3.25: Mini-Batch Gradient Descent.

3.7.3 Stochastic Gradient Descent

Stochastic Gradient Descent or SGD in short solves the problems faced by batch gradient descent. Batch gradient descent iterates over the entire dataset to find the better loss from loss function which is computationally very expensive and the learning rate is slow. Stochastic Gradient Descent takes on random values of the set to compare gradient and minimize loss. This variant of gradient descent is better effective on large

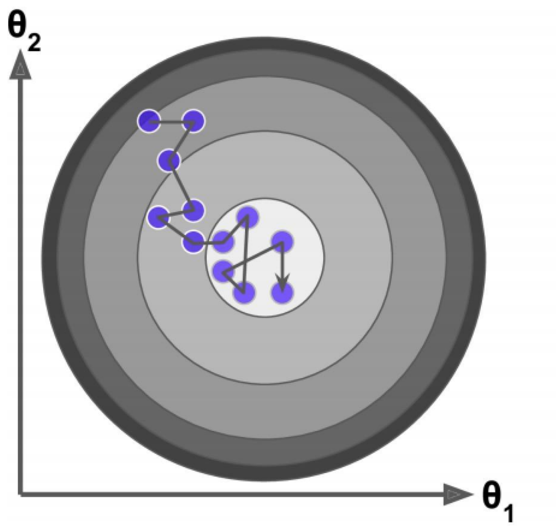


Figure 3.26: Stochastic Gradient Descent

datasets in terms of computational expanse and is also capable of avoiding local minima compared to batch gradient descent.

3.7.4 Adam

Adam stands for Adaptive moment estimation and was proposed in the 2017 by Kingma & Ba [88]. Adam optimizer was particularly designed to be more computationally efficient when worked up larger datasets with large number of parameters.

Although Adam optimized is much more efficient than the above mentioned approaches and it cleverly updates step sizes based on momentum and other changes derived from parameters, it still does not perform as good as SGD and do not perform as good as the state-of-art approaches known today [88-90].

Chapter 4

Spiking Neural Network (SNN)

4.1 What is Spiking Neural Network?

Spiking Neural Networks (SNN), which have grown in recent years, are the third generation of neural networks. Even long before the arrival of sigmoid or perceptron neurons, SNNs were popular. SNN has been shown to be suitable in both digital hardware and analog hardware for parallel execution. SNNs is found to be suitable for digital hardware and also in analog hardware for sequential execution [27]. Analog signals used to transmit information through one neuron to the another has been used by past generations of neural networks. SNN communications use spikes similar to those used by natural biological neurons. Only the moment they appear is these spikes acknowledged [28]. The preceding neuron measures a value using the weighted number of analog inputs using a non-linear sum-specific. This value determines the delay of the spike production for the next neuron. The spiking neuron will generally be considered mostly as leaky integrator. The aim of this is to incorporate spikes over time as well as to recognize the resulting integral value as the membrane potential. When a specific threshold value exceeds the membrane potency, the neuron passes a spike then resets its own membrane potential [30].

A number of external criteria (like gene and protein expression) that must be addressed for neurons to spike was explained by emerging aspect of information processing in biological neurons. These extracted features may also include physical contact properties. Some of these characteristics have indeed been mathematically modeled so applied in the context of biological neurons. SNNs are composed with artificial neurons that interact using trains, that could be thought of it as pulse coded info. SNNs are bio-plausible and have certain way to reflect info processed time, frequency, phase and other functions. SNN is able to teach neurons to transform spatial-temporal information into spikes. When the neuron structure for large SNN is selected, there is a compromise among biological and computational effectiveness. Where the computational performance is much more significant than bio-plausibility, its reduced computational cost means that the model Leaky Integrate and Fire (LIF) can be implemented [30].

4.2 Biological Inspiration Behind Spiking Neural Network

The progression of thought on the fundamental concepts of brain processing motivates a new research into normal neuronal processing. The prevalent view was that wisdom was built upon rationality when the first neural networks became developed, and that reason is the foundations of reasoning. Actual neuron spike, as an integral part of data transfer with one neuron to the another, is at least the majority of biological neural networks rely on pulse. The neuron will produce a potential for action – the spike – in the soma cell body of the neuron in a rugged, non-exhaustive contour. This rapid electric pulse (1 or 2ms duration) is then connected to the recipient end of several other neurons, dendrites, along its neuron's axon (see Figure 4.1, left view). At the axon's edge, synapses associate one neuron to the other, and at the appearance of a single spike, neurotransmitters may be released along the synaptic spike. These neurons are picked up by the neuron at the receiving end and alter the condition and, in particular, the membrane potential of this neuron, which usually makes the neuron almost susceptible to fires for a while. The temporary effect of a Spike on the membrane permeability of a neuron is normally called the PSP, and the PSP may suppress subsequent firing – inhibitory postsynaptic potentials, IPSP, or excite neurons to fire – an excitatory EPSP. The PSP can directly affect membrane potential for somewhere between 10 microseconds and 100 milliseconds, based on the neuron as well as the particular type of connection [25-27].

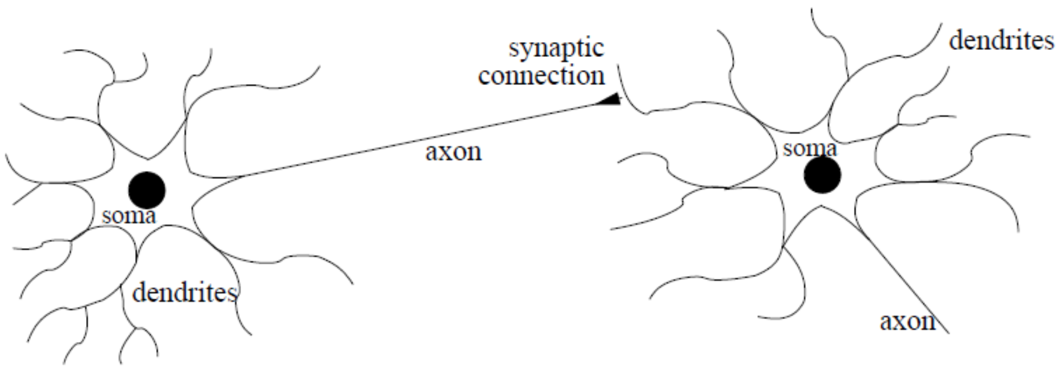


Figure 4.1: Elementary scheme of biological neurons.

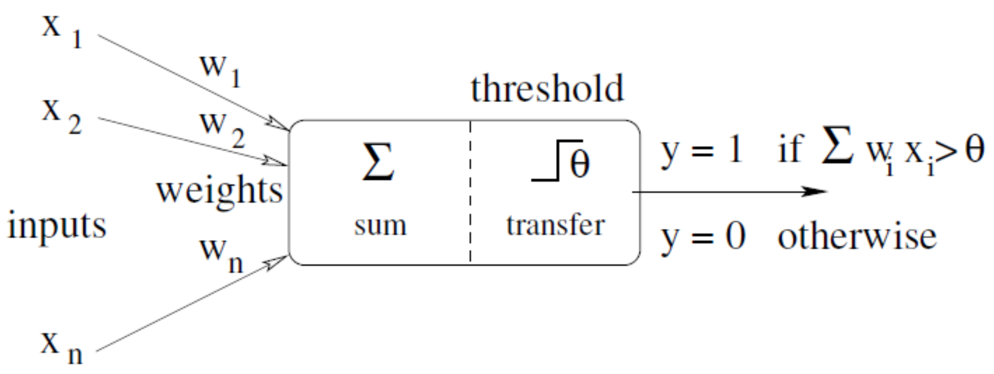


Figure 4.2: First mathematical model of artificial neuron.

4.3 Artificial Spiking Neurons

A neuron, as in an artificial neural system, is the central processing unit of an ANN. The neuron output is the linear weighted number of its inputs that have been activated. Equation and define output of an artificial neuron.

$$v_j = \sum_{i=1}^{n} w_{ji} x_i + b_j \tag{4.1}$$

$$y_j = \varphi\left(v_j\right) \tag{4.2}$$

Multiple types of activation functions allow an appropriate one to be chosen depending on the specific application. Common activation mechanism is threshold, partially linear, and sigmoidal. The main advantage of these basic processing units was their ability in the artificial spiking neural networks to acquire and preserve data from their surroundings. The ANNs will retrieve and archive knowledge from the data provided to the network via a learning process. The information derived is saved via the connection weights in the network and can be collected for future use [26].

4.4 Spike Response Model

The Spike Response Model (SRM) is a generalized leaky integration and fire model. The leaky integrate and fire model explains the neuron's biophysical pathways primarily through its membrane function. Moreover, the time lap obtained from last fire case is of great concern to this model [26]. We are going to define the SRM in detail in this section.

The neuron condition is characterized by the membrane voltage of the cell a single variable u. Suppose the neuron shot its final peak at a time $\ddot{t}$. At the time of $t > \ddot{t}$, the cell state can be written as,

$$u(t) = \dot{\eta}(t - \ddot{\mathrm{t}}) + \int_{-\infty}^{+\infty} k(t - \ddot{\mathrm{t}}, \mathrm{s}) I(t - \ddot{\mathrm{t}}) ds \tag{4.3}$$

The last expression is the product of an external current $I(t)$. The method of integration is defined by the k kernel. The kernel $\dot{\eta}$ contains the spike shape and the potential for post-hyperpolarization. We do have minimum requirement for spike development, as always:

$$\text{If } u(t) \geq \theta u'(t) > 0, \text{ then } \mathrm{t} = \ddot{\mathrm{t}} \tag{4.4}$$

Mention that spiking only takes place as the membrane tension reaches from below the limit. Either as a continuous or as a time-dependent threshold itself. The SRM is more commonly known as leaky integrate fire model. The option of the kernel k is not limited in particular. However, k is depend on time since produced final spike $t-\ddot{t}$. Lastly, the AHP is more realistically defined than the leaky-integrate-fire model, in which the time constant of the AHP is equivalent towards the membrane time constant

[26-29].

4.5 Neuron Models

4.5.1 The leaky integrate-and-fire (LIF) model

The leaky integrate-and-fire (LIF) model of spiking neural networks establishes an analytically manageable structure of the firing rate of neurons in terms of the membrane time constant, threshold, and refractory duration of a neuron [53]. In LIF models consistently depolarization occurs due to a constant current source , I. This steady current source increases the membrane potential V exponentially. When the membrane potential V exceeds the threshold level V_{Th} then a spike is generated. The membrane capacitance starts to discharge spontaneously at the initiation of the spike, then the membrane potential V is again reset to the resting potential. This firing mechanism is repeated endlessly as long as the input current is present in figure. The LIF model's RC integrated system uses the standard exponential charging framework to compare membrane potential with input current. The equation is [53]:

$$V = IR(1 - e^{\frac{t}{\tau}}) \tag{4.5}$$

Where, τ is a membrane time constant.

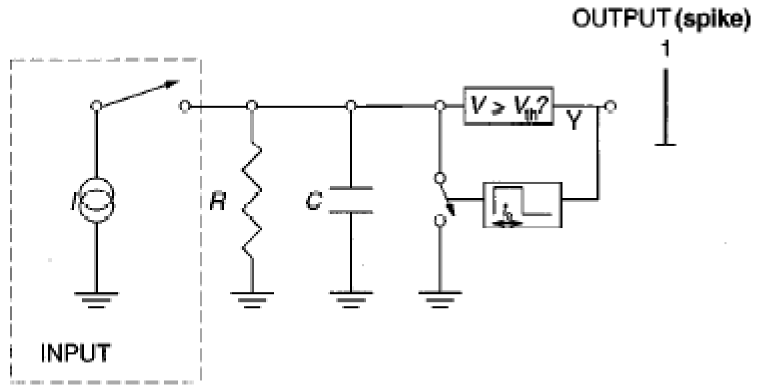


Figure 4.3: RC circuit model charging of a neuron's membrane from its resting potential to V_{Th}.

According to the Leaky Integrate and Fire (LIF) Model, the neuronal membrane is configured as a capacitor C in parallel with a resistor R and also a synaptic charging current $I(t)$ [48].

$$\tau_m \frac{du}{dt} = -u(t) + RI(t) \tag{4.6}$$

In this equation, $m = RC$ is the neuronal membrane time constant and the membrane potential is $u(t)$. As mentioned earlier, if the membrane potential $u(t)$ exceeds the triggering threshold level u_{Th} during depolarization, at that instant $t(f) : u(t(f)) = u_{Th}$, the neuron will instantaneously produce a spike, and the membrane potential is reset to the resting potential value $u_r < u_{Th}$.

$$\lim_{t \to t^{(t)}, t > t^{(t)}} u(t) = u_r \tag{4.7}$$

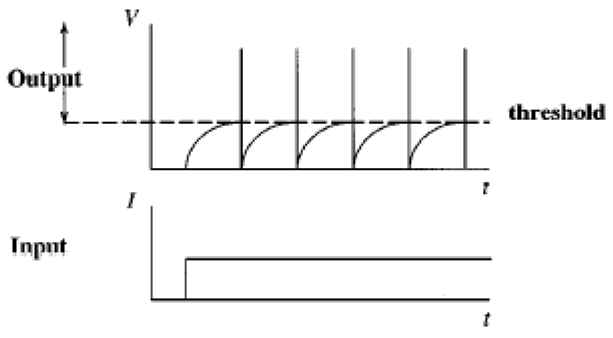


Figure 4.4: Membrane voltage versus time in an RC circuit LIF model, in response to a step current.

Specifically, the leaky integrate and fire model may include an actual refractory cycle trto reduce its firing frequency by prohibiting it from firing throughout that period: if $u(t)$ exceeds the threshold at time $t = t(f)$. During an arbitrary refractory time period the dynamics are kept freeze and adjusted the membrane potential to be ur. Then using the initial condition of membrane potential at time $t = t(f) + tr$, the dynamics can be restarted again [48].

4.5.2 The integrate-and-fire neuron model

The mechanics of the LIF neuron are described by the equation below:

$$\tau \frac{dv}{dt} = -\left(V - V_r\right) + R\left(I_o + I_{in} + I_n\right) \tag{4.8}$$

where $\tau = RC$ signifies the time constant of the membrane time constant, C is the membranes' capacitance, and R represents the membrane resistance, V specifies the membrane potential, thus V_r denotes the resting potential. As V reaches a constant level threshold V_{Thr}, the neuron bursts, and V is reset as V_r for something like a refractory time $t_r = 0.01s$. Io represents the continuous insertion current, In represents the ambient noise current, which will be set to zero, and the input current will be I_in [54].

The figure below shows the simulation of Leaky Integrate and Fire Neuron Model.

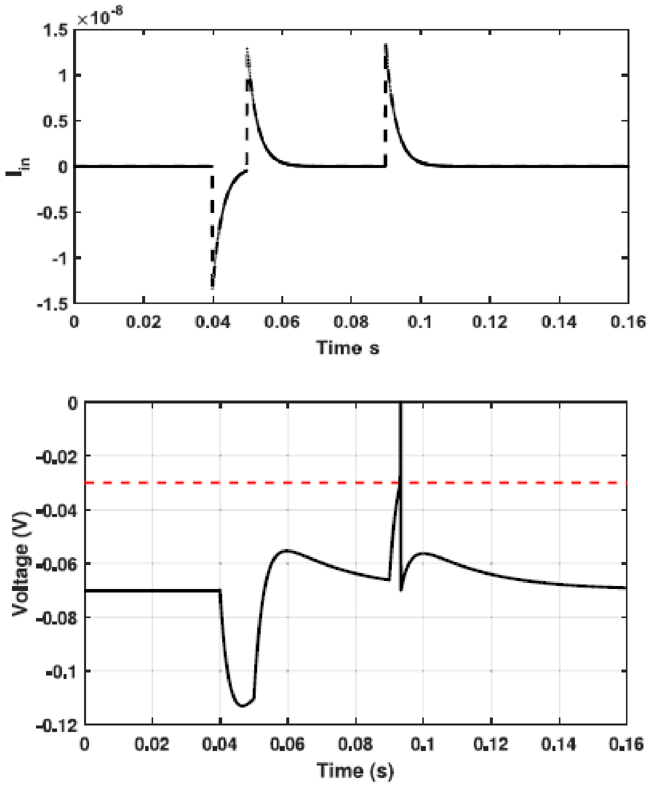


Figure 4.5: The Simulation of Leaky Integrate and Fire Neuron Model.

4.5.3 Different Neuron Models

Spiking neural networks (SNN) have achieved popularity in recent years, based on their genetic plausibility like human brain spikes. SNN can encode spatial-temporal information through both spiking rates and spike time duration using different spike-based neuron models. These neuron models have benefits over conventional spiking-rate dependent artificial networks which only use spiking rates [48]. Figure below shows action potentials pass through axons and trigger synapses process, illustrating the basic process of spiking signal propagation. The neurotransmitter released by these synapses dissipates rapidly to the postsynaptic neuron. These neurotransmitters influence the membrane potential of the postsynaptic neuron. Excitatory Postsynaptic Potentials does depolarization, and this increase in membrane potential creates excitation, therefore, leaks along with a standard time constant in the absence of new synaptic inputs [47].

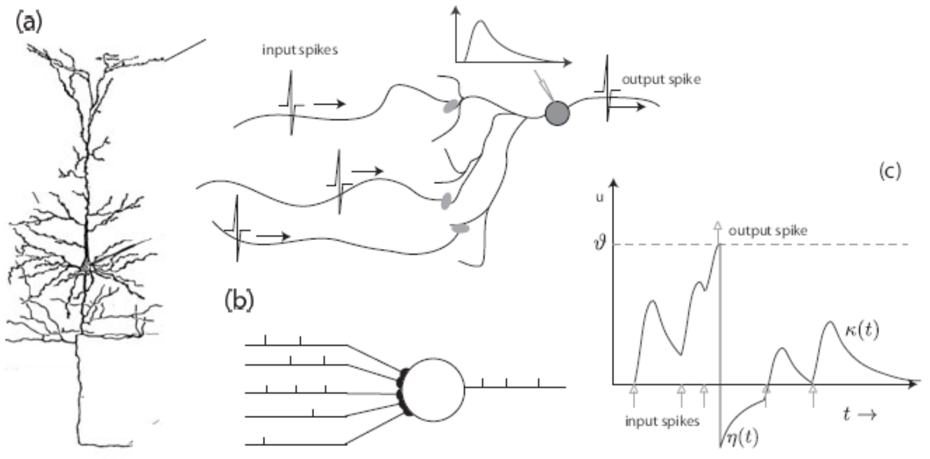


Figure 4.6: Different Neuron Model Structure.

After depolarization Inhibitory Postsynaptic Potentials does hyperpolarization. When a certain amount of EPSPs reaches a neuron the membrane potential starts to depolarize. After a certain depolarization, the membrane potential reaches a certain threshold level, and then it creates a spike by itself by regenerating the membrane potential. Then this spike propagates to other neurons to generate more spikes [47].

Actual neurons have a variety of spiking patterns: a few responses to feedback with only a delay, while some give response bursts of a lot of spikes. Several remarkable spiking behaviors displayed by neuron models are already analyzed utilizing mathematical tools based on computational efficiency and biological plausibility [50]. To illustrate patterns and spiking behavior, comprehensive models were established.

The most significant model is the Hodgkin-Huxley model.Hodgkin and Huxley developed a mathematical model in 1952 to compensate for the diffusion of electric current through all the surface membranes of a squid giant neural fiber. The Hodgkin-Huxley model is employed to describe the multiple spiking behaviors of a neuron after it has been stimulated with varying currents [49]. In cognitive neuroscience, the

Hodgkin-Huxley model is amongst the most biologically feasible models. Their model is indeed a complex nonlinear ODE structure made up of four equations that describe the membrane potential, activation, and inhibition of various ionic gating parameters [38]. The equations of HH models are:

$$\frac{du}{dt} = -g_{Na} m^3 h\,(u - E_{Na}) - g_K n^4\,(u - E_K) - g_L\,(u - E_L) + I(t) \tag{4.9}$$

$$\frac{dm}{dt} = \alpha_m(u)(1-m) - \beta_m(u)m \tag{4.10}$$

$$\frac{dn}{dt} = \alpha_n(u)(1-n) - \beta_m(u)m \tag{4.11}$$

$$\frac{dh}{dt} = \alpha_h(u)(1-h) - \beta_h(u)h \tag{4.12}$$

In this equation, u(t) represents the neuron membrane voltage, variables g_{Na}, g_k, g_L are utilized for model network channel conductances. For example, the maximum conductance for $Na+$ channel is g_{Na}, and when all networks are active for K+ channel is g_K . And other mentioned parameters m,n regulate the opening of Na+ channels, while parameter h regulates the opening of the $K+$ channel. The reversing potentials are represented by the parameters E_Na, E_K, E_L [48].

Utilizing bifurcation theory, The Izhikevich model is an ordinary differential equation having 2 dimension and 3 equations,

$$\frac{dv}{dt}(t) = 0.04v^2 + 5v + 140 - u + I(t) \tag{4.13}$$

$$\frac{du}{dt}(t) = a(bv - u) \tag{4.14}$$

If v $>$ 30 mV, then $v - c$ and $u - u + d$.

Here, the membrane potential of neuron is represented by parameter v, while parameter u is used for slow membrane recovery variable for the activation of $K+$ ion currents and inactivation of Na+ ion currents [51] $I(t)$ represents the excitatory input current while a, b, c, d are variables that can be modified and they are dimensionless gain. Based on the rule, whenever the membrane voltage $v(t)$ approaches its peak (30 mV), a spike is generated, and also the membrane voltage $v(t)$ and recovery variable u are restored and reset to their default values. And the IZ model's resting potential ranges between -70 and -60 mV based on the value of b. To imitate typical spiking neurons we can give some fixed time scale values of recovery variable a, b, c, d [48].

Hugh Wilson proposed an alternate solution to the Hodgkin-Huxley formulations depending upon the basic HH model's polynomial curve fitting [52]. The equation of WIL model:

$$c\frac{dv}{dt} = -\left(17.81 + 47.71v + 32.63v^2\right)(v - 0.55) - 26.0R(v + 0.92) + I(t) \tag{4.15}$$

$$\frac{dR(t)}{dt} = \frac{1}{\tau}(-R + 1.35v + 1.03) \tag{4.16}$$

In WIL model, v represents the membrane potential, R represents the recovery variable and the values of $c = 0.8/cm^2$ and $\tau = 1.9ms$. In this model, the membrane potential v is calibrated to be restricted in a small area, as [-2,2] mV, and also the current is calibrated to be approximately 1% of the HH model's input current [53].

The FitzHugh-Nagumo model is based on phase plane analysis and is an equivalent to the Hodgkin-Huxley model [48]. A polynomial regression model of the preceding format below was achieved by imitating the nullclines of the Hodgkin-Huxley model using a cubic function and a straight line [54]:

$$\frac{dv}{dt} = v - \frac{v^3}{3} - w + I(t) \tag{4.17}$$

$$\frac{dw}{dt} = \varepsilon(v + a - bw) \tag{4.18}$$

The variables have been measured in this context, with the rapid parameter v representing membrane potential and the slow parameter w operating as a recovery variable.

The variables a, b, ϵ are employed to compensate for the recovery variable's time scale and activation energy [55].

4.6 Learning in Spiking Neural Network

The correlation between data and spike patterns is defined by synaptic neuron codes. The goal will be to use spiking neural networks to accomplish effective computations [56]. Learning is performed in almost all Artificial Neural Networks, spiking or non-spiking, by varying scalar-valued neural weights. Spiking makes it possible for the replication of a kind of bio-plausible learning rule which is not possible in non-spiking networks. Several variations of such learning rules have been established and defined by neuroscientists which falls under spike-timing-dependent plasticity (STDP). Its main aspect is that the weight connection between a pre-synaptic and post-synaptic neuron is modified based on its respective spike time over a tens of millisecond time period [54]. The preceding subdivisions cover all supervised and unsupervised learning rules in SNNs:

Unsupervised learning data is implemented without a label, and gets no feedback performance. Detecting and adjusting the synaptic patterns in data is the basic exercise. Hebbian learning and also its spiking presumptions, like STDP, is a typical example of this [47]. Here STPD works as a learning mechanism. The most basic structure of bio-plausible STDP has a concise concept. The most basic structure of bio-plausible STDP has a concise concept. The synaptic weight connection is strengthened if a pre-synaptic neuron triggers for 10 milliseconds triggering the post-synaptic neuron. If the pre-synaptic neuron activates just after the post-synaptic neuron for a short period of time, the correlation between the spatial events is distorted, and the synaptic weight becomes weak. Here, Long-term potentiation (LTP) is the process of strengthening and long-term depression is the process of weakening (LTD) [61]. For a single pair of spikes, the equation below recognizes the most commonly found STDP rule acquired by adjusting with experimental data [55].

$$\Delta w = \begin{cases} Ae^{\frac{-(|t_{pre}-t_{post}|)}{\tau}} & t_{pre} - t_{post} \leq 0, \quad A > 0 \\ Be^{\frac{-(|t_{pre}-t_{post}|)}{\tau}} & t_{pre} - t_{post} > 0, \quad B < 0 \end{cases} \tag{4.19}$$

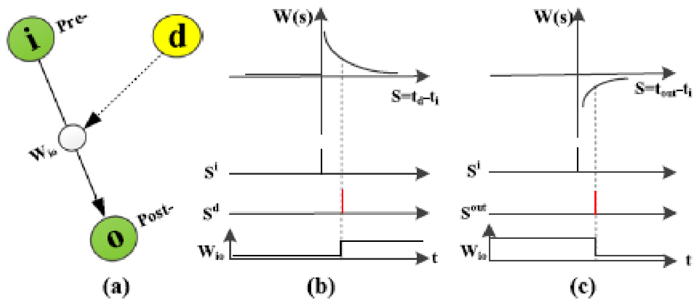


Figure 4.7: (a) shows the connection between pre-synaptic and post-synaptic triggering times, as well as the connection between pre-synaptic and target triggering times, which influences synaptic plasticity. (b) shows whenever a required spike is produced, the synaptic weight is enhanced. (c) demonstrates that whenever a spike is triggered, the synaptic weight decreases. When the real output spike is induced at a specific time, no further modifications are implemented [59].

Here, the synaptic weight is represented by w. In most cases, A $>$ 0 and B $<$ 0 are constant variables that indicate learning rates. is the temporal τ learning time constant which is 15 ms. LTP is mentioned in the first instance, while LTD is defined in the second. The amplitude of the impact is controlled by the constant-scaled time difference between the pre-synaptic and postsynaptic spikes, which is modulated via a decreasing exponential [59].

In supervised learning, input data is paired with the expected target labels, and the task is to correlate the input classes with the target outputs, resulting in a projection or regression between inputs and outputs. Between both the expected target and the real outcome, an error signal is measured and utilized to change the synaptic network's weights. SpikeProp tends to be the very first method to back-propagate errors to train Spiking Neural Networks. SpikeProp could do classification of non-linearly differentiable data for a spatio - temporal encoded XOR error utilizing a 3-layer structure because their cost function kept spike duration into consideration. Through implementation of Gerstner's spike-response model (SRM) for the spiking neurons was among their core design ideas [57]. Since the outputs of the hidden modules can be precisely modeled by constant values of PSPs implemented to the output neurons which they predicted to. So the problem of using derivatives from their output spikes was eliminated using the SRM model. One drawback of this study is that every output device was limited to emitting only one spike [61]. Later Multi-SpikeProp, a more versatile edition of SpikeProp, are used with multiple spike coding.

Another learning rule, ReSuMe stated that it can execute accurate spike-timing-based learning, combining a Hebbian and an anti-Hebbian mechanism within a learning span [56]. After training, this could emit several targeted spikes responding to the provided input pattern [58]. The intended output $(S^d(t))$ or the authentic output $(S^{out}(t))$ triggers the adjustment of weights by:

$$\frac{dw_i(t)}{dt} = \lambda \left[S^d(t) - S^{\text{out}}(t)\right] \left[a_d + \int_0^{\infty} W(s) S_i(t-s) ds\right] \tag{4.20}$$

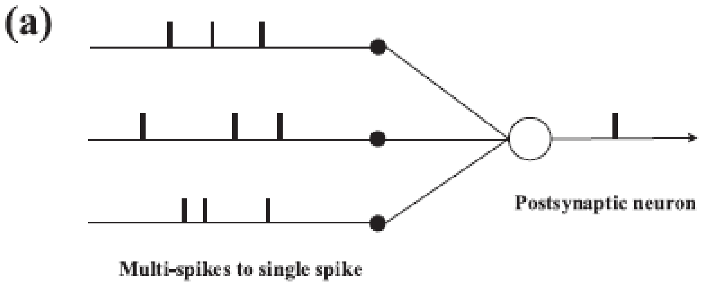


Figure 4.8: Spike rhythms for the Tempotron code that only emits a spike when multiple input spikes are present.

Here, learning rate is represented by λ, W is the learning window and and a_d is constant. The convolution variable is the last part of the equation which determines the magnitude of the transformation.

The above figure depicts the ReSuMe learning rule. (a) shows the connection between pre-synaptic and post-synaptic triggering times, as well as the connection between pre-synaptic and target triggering times, which influences synaptic plasticity. (b) shows whenever a required spike is produced, the synaptic weight is enhanced. (c) demonstrates that whenever a spike is triggered, the synaptic weight decreases. When the real output spike is induced at a specific time, no further modifications are implemented [59].

Another learning rule that is recently used is The Tempotron rule. This learning rule will maximize synaptic weights by minimizing the possible difference between the triggering thresholds and the actual membrane potential [60]. But it only contains two output solutions: triggering or not triggering. The triggering phase is intended to refer to a class (P+), whereas the absence of a triggering state is represented with a sequence corresponding to another class (P-). Based on the LIF model, post-synaptic neurons attain spikes from multiple pre-synaptic neurons, resulting in varying levels of stimulation or activation by distinct synaptic weights. From all input spikes, the weighted sum of all input spikes' post-synaptic potentials (PSPs) are the membrane potential of a neuron:

$$V(t)=\sum_i w_i \sum_{t_i} K\left(t-t_i\right)+V_{\text{rest}} \tag{4.21}$$

Where, the synaptic efficacy is denoted by w_i and t_i is the i_{th} afferent triggering time. And the resting potential is denoted by V_{rest}. The Tempotron rule will alter the synaptic weights (w_i) whenever an error occurs [58]. And for the reduction of costs, the gradient descent learning rule is implemented:

$$\Delta w_i=\begin{cases}\lambda \sum_{t_i<\text{sex}} K\left(t_{\max}-t_i\right) & \text{if } P^{+}\text{error;} \\ -\lambda \sum_{t_i<\text{lmax}} K\left(t_{\max}-t_i\right), & \text{if } P^{-}\text{error;} \\ 0, & \text{otherwise.}\end{cases} \tag{4.22}$$

Here, t_{max} is the time that the neuron approaches its peak potential value in the required time space. $\lambda > 0$ is a constant that represents the learning rate and demonstrates the maximum improvement in synaptic efficacy.

Multiple temporal learning rules produce spike rhythms and learning windows.

4.7 Implementation of Spiking Neural Network

Spiking Neural Networks are more practical biologically inspired model approach in the recent years that promises more efficiency in terms of computation and energy [73, 75, 82, 83]. Our brains are amazing at recognizing patterns and are also very efficient with energy utilization. Brains tend to utilized spikes and produce spike trains that are spare over time by nature [70]. In ANNs, non-linear activation functions model the behavior of each neurons whereas real neurons produce action potentials or spikes. The spike rate duration contain a lot of information. The nature of SNNs show promise of more efficient hardware that will be brain like, however challenges lay in implementing then in currently popular computer architectures. Simulation of SNNs in software is also possible and has been done by many, finding very promising performances. [72, 73, 76, 82, 83, 91].

4.7.1 Existing Software Implementations

Many software based simulation of spiking neural network implementation is available, however none offer any clear documentation which makes them very difficult to utilize. Some of the publicly available SNN implementations available today are –

- BindsNet
- Brain
- SpykeTorch
- Norse
- SpikingJelly
- Nengo
- PySNN
- SNN toolbox

There are several approaches for training Spiking Neural Networks. Some approaches include converting existing ANNs to SNNs. Other approaches adopt different learning rules or utilize Back-propagation to extract results.

4.7.2 Implementation Approaches

Many implementation approaches have been taken to implement the spiking neural network algorithms. These approaches include converting existing trained architectures to Spiking Networks, utilizing back propagation algorithms to make the network learn and etc. Some of the approaches are explained below.

Conversion-Based Methods

Despite being a very promising technology, Spiking Neural Networks have been applied on relatively non-complex datasets like MNIST or CIFER datasets. A big reason behind that is its scalability issues. Complex dynamics of the training algorithms and non-differentiable operations of spiking neurons [82]. Spation-temporal nature of the spikes in a spiking neural network deliver crucial information. The continuously differentiable properties no longer apply to the internal state and loss function of SNN. Thus, conventional methods no longer remain feasible for SNNs.

One way to mitigate this problem is by mapping the parameters cultivated from a trained CNN model and convert them to equivalent SNN parameters. [70, 82, 83] attempted this approach first according to our best knowledge and found directly converting a CNN architecture into SNN still results in unacceptable performance drops. They implied that the reasons for such performance drop are –

- **Negative Values:** The use of sigmoid activation function which output [-1, 1]. Weights and biases calculated from inputs coming through these activations can be negative. Output values from preprocessed dataset can also be negative however, negatives values impact the accuracy of being represented in SSN which causes the performance drop.
- **Biases Representation:** The biases that come with CNN are difficult to represent in SNN.
- **Max Pooling:** Max pooling is a good way to harvest spatial insights from images. However, SNN require 2 layer networks to implement this, resulting in added neurons and complexity.

[70] suggest to solve these mentioned problems by –

- Keeping all values positive by adding an abs() function layer after preprocessing phase and using ReLU function to rectify the issues.
- Setting all biases to zero within the entire network.
- Use Spatial Linear Subsampling instead of max-pooling

Everything from the CNN remains same. The neurons remain the same but the weights of each neuron of previous CNN is converted to synaptic strengths in the converted SNN architecture. Connections in SNN are now axioms and spikes propagate through the network.

CNN Architecture

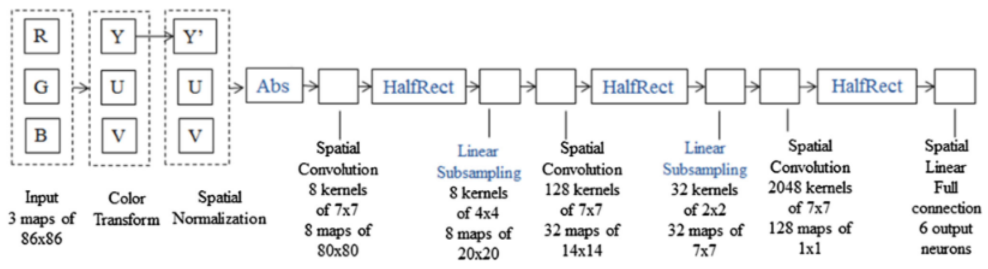

R
G
B
Y
U
V
Y'
U
V
Abs
HalfRect
HalfRect
HalfRect
Input 3 maps of 86x86
Color Transform
Spatial Normalization
Spatial Convolution 8 kernels of 7x7 8 maps of 80x80
Linear Subsampling 8 kernels of 4x4 8 maps of 20x20
Spatial Convolution 128 kernels of 7x7 32 maps of 14x14
Linear Subsampling 32 kernels of 2x2 32 maps of 7x7
Spatial Convolution 2048 kernels of 7x7 128 maps of 1x1
Spatial Linear Full connection 6 output neurons

Figure 4.9: Tailored CNN Architecture.

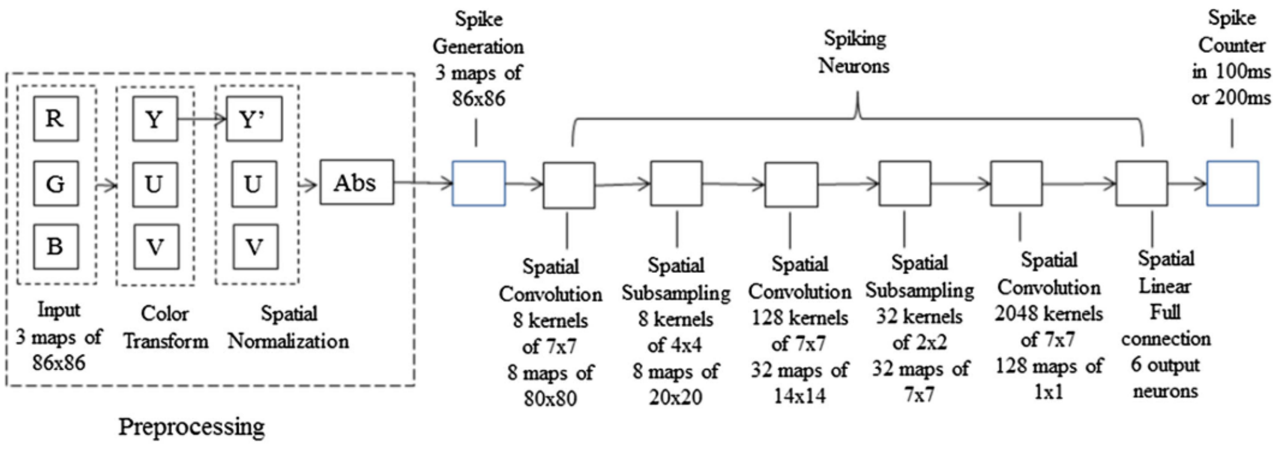

Spike Generation 3 maps of 86x86
Spiking Neurons
Spike Counter in 100ms or 200ms
R
G
B
Y
U
V
Y'
U
V
Abs
Input 3 maps of 86x86
Color Transform
Spatial Normalization
Preprocessing
Spatial Convolution 8 kernels of 7x7 8 maps of 80x80
Spatial Subsampling 8 kernels of 4x4 8 maps of 20x20
Spatial Convolution 128 kernels of 7x7 32 maps of 14x14
Spatial Subsampling 32 kernels of 2x2 32 maps of 7x7
Spatial Convolution 2048 kernels of 7x7 128 maps of 1x1
Spatial Linear Full connection 6 output neurons

Figure 4.10: Block Diagram of Spiking CNN (SNN) Architecture.

SNN trained with Back-propagation

Spikes are impulse and they are not continuous and non-differentiable. If differentiation is not possible, it is also not possible to calculate slops and make sense of how to tune so that the network performs better through an iterative process. This is the biggest problem with SNN. However, many approaches have already been proposed against this problem [86]. In general approach, the spikes are sharp impulses voltage rises instantaneously when the membrane potential reaches threshold. The modelling looks like this –

$$\tau \dot{s} = -s + \sum_{k} \delta \left(t - t_k\right) \tag{4.23}$$

Here, δ is Dirac-delta function tk denotes the time of threshold-crossing. In this approach, the spikes are cleverly modelled to be differentiable by allowing the synaptic current to increase in gradual manner.

$$\tau \dot{s} = -s + g\dot{v} \tag{4.24}$$

Here the impulse function has been replace by a gate function g. Time integral of the synaptic current is a dimensionless quantity, thus the integral is always 1 indicating the fact that the total amount of current is always same regardless the time needed for depolarization.

$$\int sdt = \int g\dot{v}dt = \int gdv = 1 \tag{4.25}$$

Now the network can be modelled as –

$$\vec{I} = W\vec{s} + U\vec{i} + \vec{I_0} \tag{4.26}$$

Where the input current vector depends upon: recurrent connectivity weight matrix (W), input weight matrix (U) and ionic current (I_o). Now as all the involved parameters are differentiable, the above model can be optimized utilizing gradient descent [80].

4.7.3 SNN trained with Biological Plasticity Rules (STDP, Hebbian,etc)

This is an unsupervised learning method that is more brain like. This approach utilizes the plastic nature of the neurons in brain. The brains is composed of neurons which are connected to one another via synapses. Information in one neuron is processed

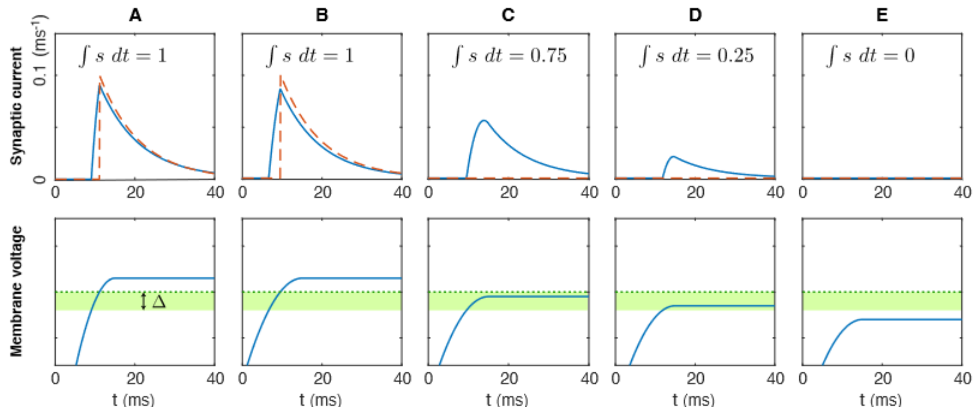


Figure 4.11: Synaptic Current in response to membrane voltage.

and also transmitted to the next neuron through chemical changes which influence the corresponding neurons to also trigger. It has been observed that, for regular tasks neurons connected together often show a pattern of firing. Some neurons are more likely to fire within 20ms of the previously connected neuron and some others might not fire as frequently [69, 87].

Overtime, the neurons that frequently fire in corresponding order seems to develop a permanent behavior that is called, “LTP” or Long-Term Potentiation. The Neurons that don’t seem to maintain strong connections develop “LTD” or “Long-Term Depression”. Spiking Neural Networks with Leaky-Integrate and Fire Neuron Models which utilize this learning technique of unsupervised biologically inspired learning model have shown very robust performance in environmental sound recognition tasks [76]. Spiking Neural Networks have been implemented in hardware. It is the promise of highly efficient low power hardware implementation that makes this technology so appealing. Attempts have already been taken to implement SNN on VLSI based Spiking Hardware [88]. The existing architectures are very energy demanding and CNNs often require GPU for training. GPUs are extremely power consuming. The way these are trained make them hard to implement over the von Neumann architecture of computers that we all use today [88]. However, the spikes are easy to model in digital systems and are also easy to interconnect and scale. The parallelism of the way these networks are hoped to function make them very suitable to be implemented in hardware which would consume ultra-low-power which is revolutionary.

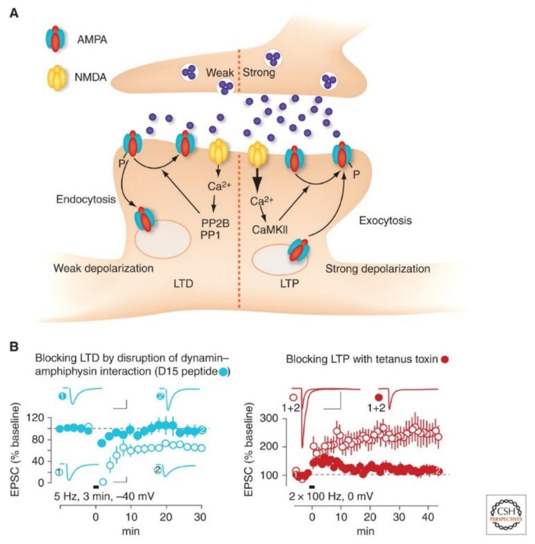


Figure 4.12: Spike Timing Dependent Plasticity [71].

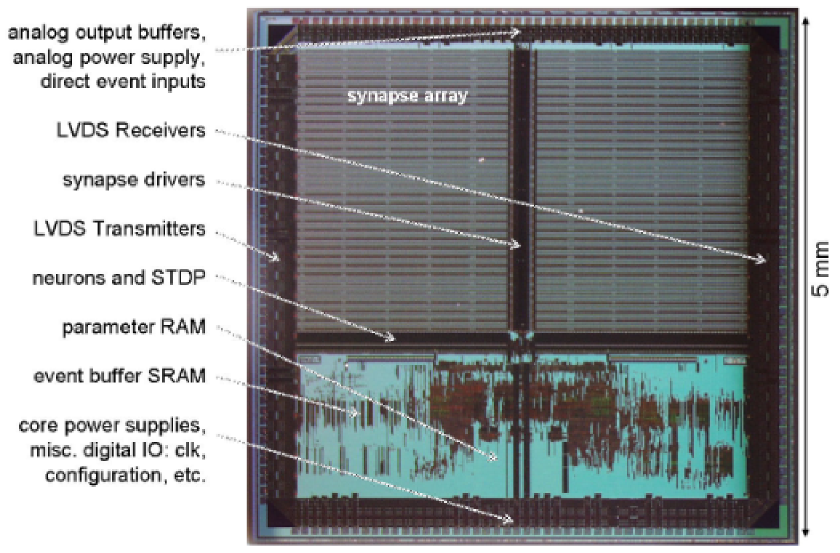

analog output buffers, analog power supply, direct event inputs
LVDS Receivers
synapse drivers
LVDS Transmitters
neurons and STDP
parameter RAM
event buffer SRAM
core power supplies, misc. digital IO: clk, configuration, etc.
synapse array
5 mm

Figure 4.13: Dia-photograph of a presented neural network chip [91].

4.8 Applications of Spiking Neural Network

The limits of SNN is yet to be discovered and there is still puzzles to solve before that happens. However, it is expected to bring revolutionary changes to our views regarding the limits and capabilities of computers till date. The energy intensive computational tasks like image, speech detection, and other tasks that only human brains perform the best will be challenged. The power consumptions will be significantly lower than what require today on von Neumann architecture.

Vision, auditory and other sensory based system will be challenged easily and get

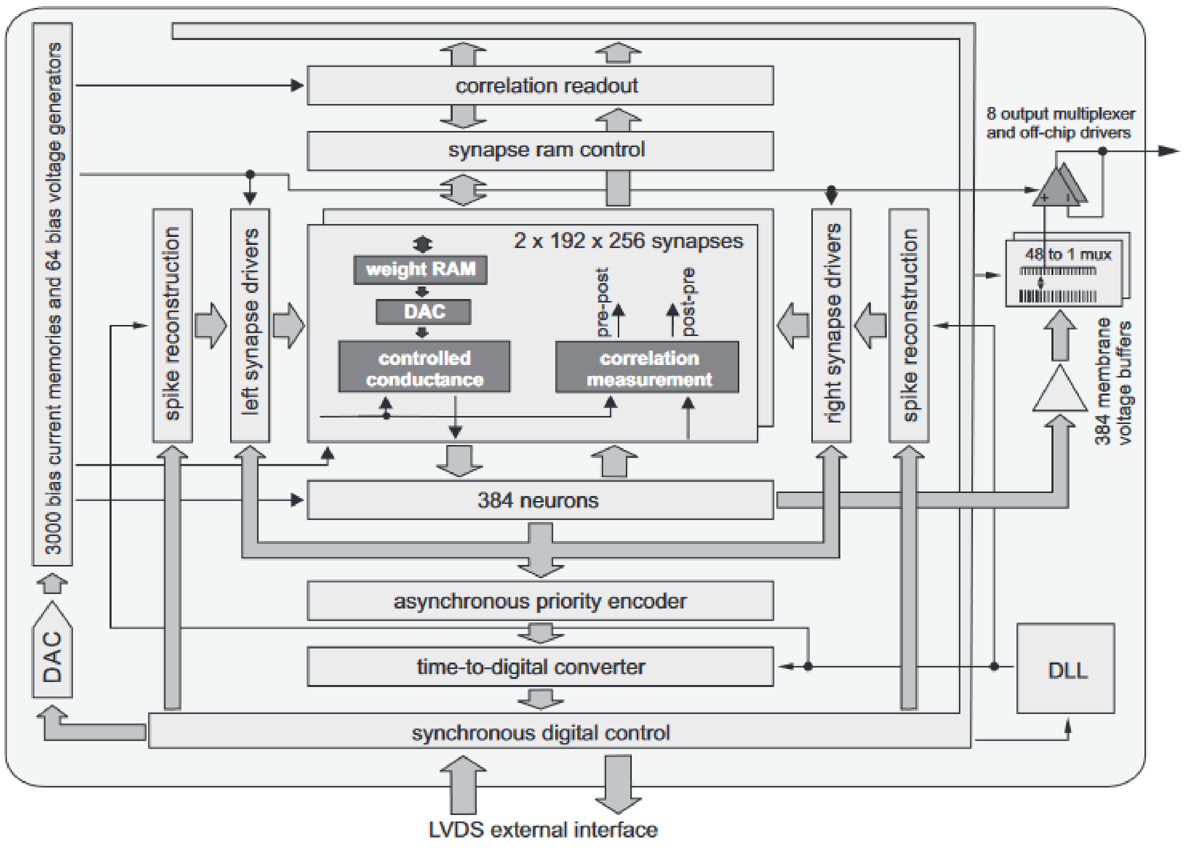
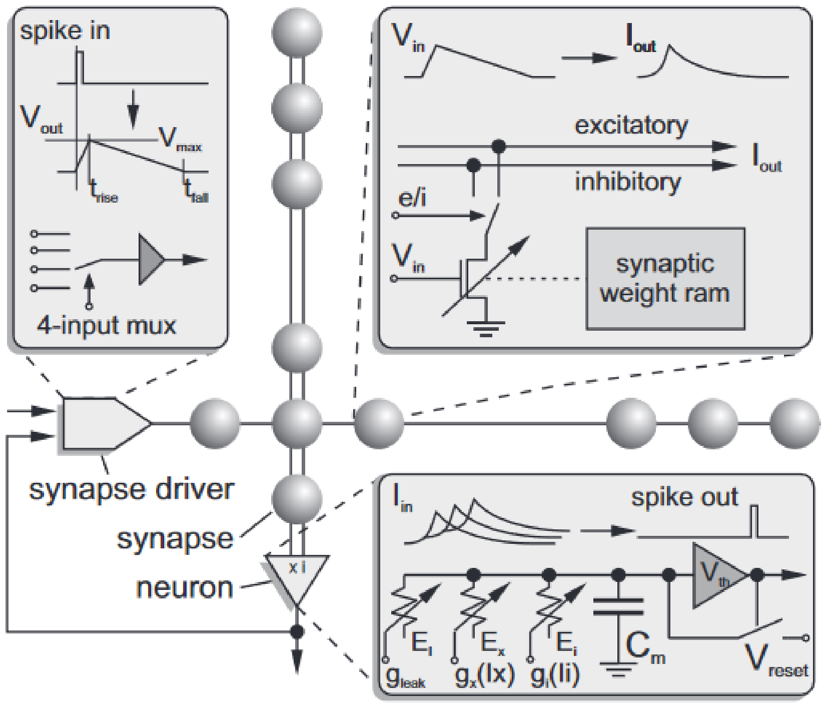


Figure 4.14: Block diagram of a spiking neural network chip. .

new benchmarks. Biomedical instrumentations systems will also experience changes. Spike based pacemakers, retina implants and many wearable prosthetics technologies are expected to assist many lives off misery and misfortune. The robots will get a lot better, thus space, robotics and automation industry are to see big changes too. The potential of such systems is immeasurable and the development of neuromorphic systems will change the way we have known life.

4.9 Challenges on Spiking Neural Network

Spiking Neural Networks and its promise neuromorphic computing still has a lot of challenges to overcome and questions to answer. Figuring out precisely how the brain functions and modelling them for efficient software and efficient dedicated hardware is still a long road. Another big challenge lies in the broadness of this field as a whole. It require experts from many different fields like neuroscience, software engineering, computer engineering, material science, electrical electronics engineering and many



.

Figure 4.15: Operating principle of a spiking neural network. .

more. Despite all these, 3 of the most major challenges of Spiking Neural Networks along are – Comparability, Generalization and lack of formulation [74, 89].

One of the most major problem with this architecture is the challenge of learning which has no general algorithm yet. Unlike in ANN/CNN architectures where back-propagation and its variants offer a generalized solution for all problems, SNN still lacks generalized algorithms. The idea nature of the spikes are sparse, which are mathematically discontinuous and non-differentiable. This becomes a very major issue for training the model. If the parameters are not differentiable, there is no way to calculate gradient and understand if the small step changes in the model are positively enhancing the system performance and tune itself to perform better. The implementations are also limited and lack proper documentation for applications.

Chapter 5

Experimental Methodology

This experimental thesis works with Environmental Sound Classification (ESC) - 50 dataset. The methodology includes developing a dedicated spiking neural network for environmental sound classification and compare the results with other convolutional neural networks. In past decades, utilizing CNN in image classification as well as audio classification has attained a huge popularity. In the following sections of this chapter discuss about various experimental phases and visualization of this experiment.

5.1 Block Diagram of Proposed Method

The block diagram is illustrated in Figure 5.1.

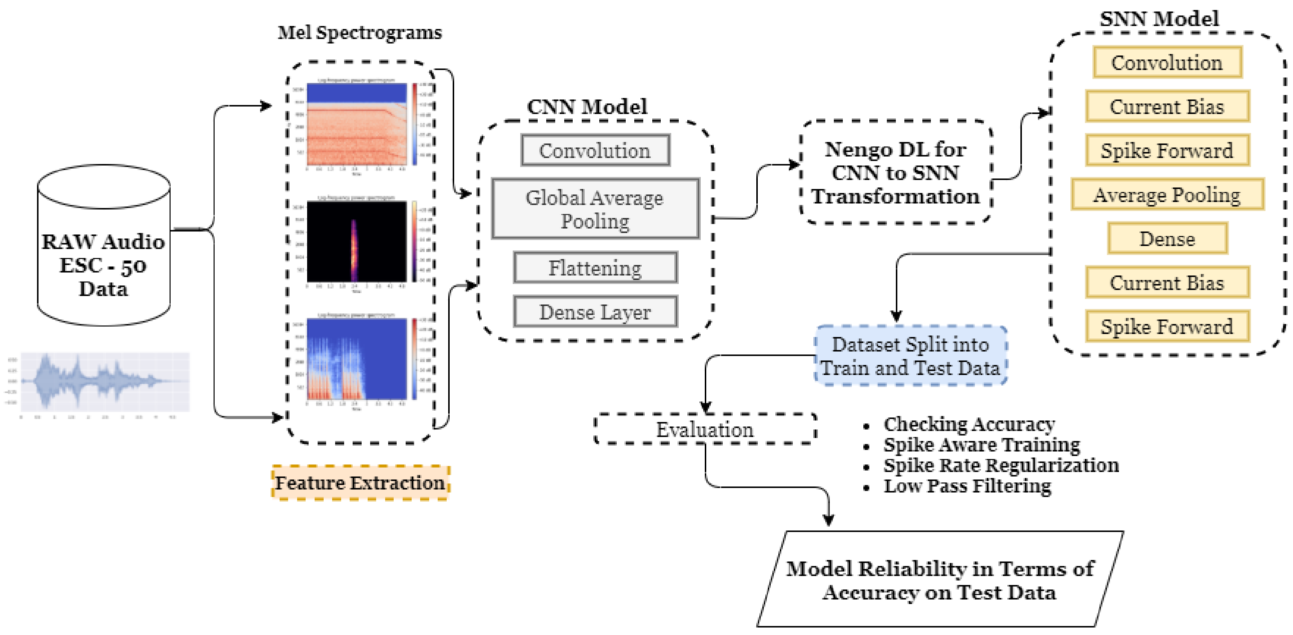


Figure 5.1: Schematic Block Diagram of Our Proposed Methodology.

5.2 Dataset Description

The dataset is an open-sourced publicly available dataset. It is retrieved from Kaggle ESR dataset. Originally it has been developed for environmental sound classification (ESC) and named as ESC-50 dataset. The dataset has been developed by Karol [92]. This dataset is an extensive combination of 2000 individual clips with 50 individual classes. It is mainly created for "Freesound Project".

The dataset is constructed into three major parts. It contains a labeled 50 class compressed sound records. The dataset reconstructed from open sourced 'Freesound Project'. There are three types of ESC datasets such as ESC-50, ESC-10 and ESC-US. In this study, we have utilized the ESC-50 dataset which has more classes and sound clips. This will help us to build a more robust classification system rather than generic ones. ESC-50 dataset has 50 classes from 10 classes each from 5 individual category. However, it is constructed with 2000 clips. Every class has equally distributed clips of 40 clips per class. So we can understand it's a balanced classified dataset. The classed are grouped into 5 major category. Such as

- Animal Sounds
- Water Sounds and Natural Soundscapes
- Human (Non-vocal) Sounds
- Interior Sounds
- Exterior unwanted Sounds/Noises

The main challenge of the extraction part is to minimize the inherent sound noises as much as possible. Some of the field recordings are far away from the desolate. However, some of the clips have overlapping noise for this reason. Some frequent sounds are laughter, cat meowing, dog barking etc. The main drawback of this dataset might be lack of sound classes and limited portion of the clips in per class. A brief summary of the dataset is presented in Table 1.

Table 5.1: A Broad Description of the ESC-50 Dataset

Animals	**Natural soundscapes & water sounds**	**Human, non-speech sounds**	**Interior/domestic sounds**	**Exterior/urban noises**
Dog	Rain	Crying baby	Door knock	Helicopter
Rooster	Sea waves	Sneezing	Mouse click	Chainsaw
Pig	Crackling fire	Clapping	Keyboard typing	Siren
Cow	Crickets	Breathing	Door, wood creaks	Car horn
Frog	Chirping birds	Coughing	Can opening	Engine
Cat	Water drops	Footsteps	Washing machine	Train
Hen	Wind	Laughing	Vacuum cleaner	Church bells
Insects (flying)	Pouring water	Brushing teeth	Clock alarm	Airplane
Sheep	Toilet flush	Snoring	Clock tick	Fireworks
Crow	Thunderstorm	Drinking, sipping	Glass breaking	Hand saw

5.3 Audio Visualization

For better understanding of the dataset, some random samples of the dataset have the visualized. The visualization can be obtained in Figure 5.2. Code snippet is given following for the visualization part.

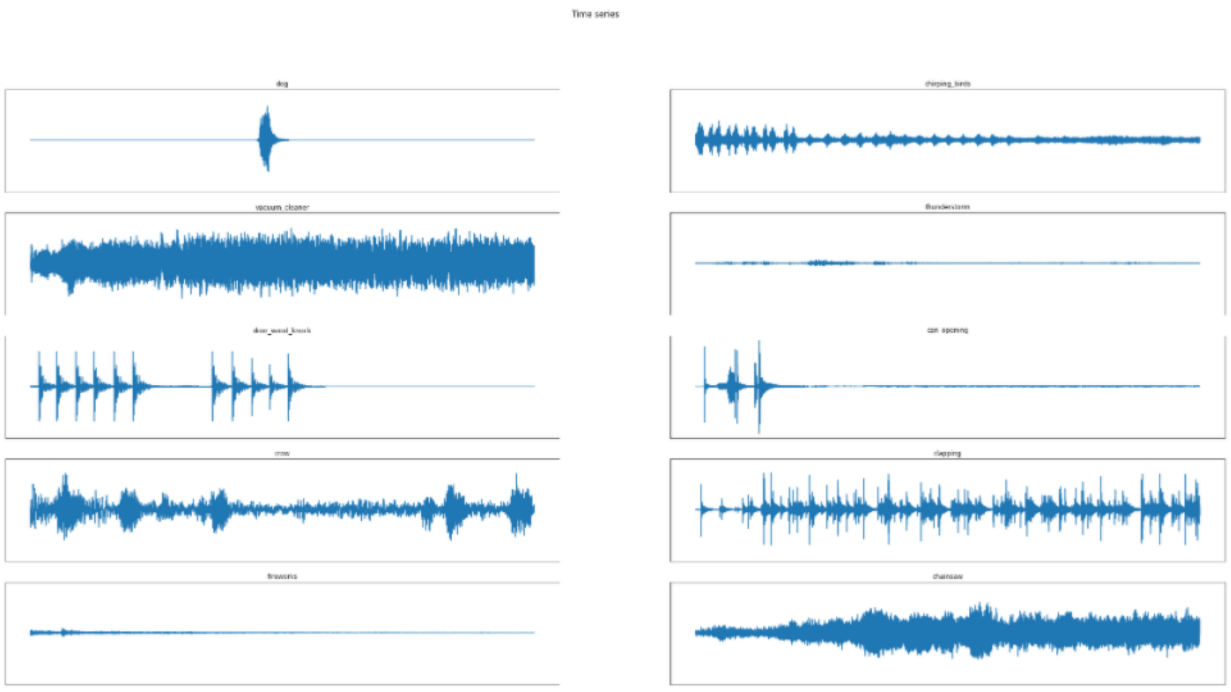


Figure 5.2: Random Class Audio Signal Visualization.

5.4 Feature Extraction

Feature extraction is defined as one of the important stage of developing machine learning and deep learning models. It mainly used for reducing the number of quantity from an image and positioning it to a defined label. In most cases, machine learning models are not well prepared for analyzing complex data. So feature extraction can play a major role to enhance the performance of the machine learning models by extracting features from an image and convert it to numerical properties. It also reduces the computational power for analyzing large dataset. In this experiment, we have used Log-frequency power spectrogram to input into our trained models.

5.4.1 Mel-Spectrogram

Mal-spectogram is a kind of spectrogram which is converted in mal-scale. Mal-scale is a non-linear transformation of frequencies which attempt to process them in more human ear-like manner. Human ear does not process all frequencies in same manner. Our ears can differ changes in lower frequencies better than higher frequencies. So, it is much easier to tell the difference between 100hz and 200hz rather than the difference between 10,100hz and 10,200hz, although the difference between both is same.

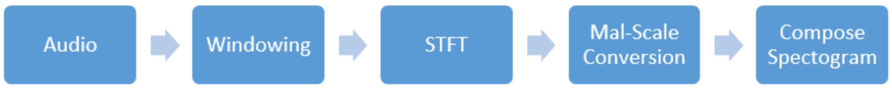


Figure 5.3: Audio to mel spectrogram conversion process.

At first we take the audio samples and perform windowing. Windowing is a method of taking a specific number of samples and then multiplying then with a specific function that allows the middle values to be maximum while the values of the starting ending points are made zero in slow and smooth manner. This is beneficial for converting the function to frequency domain as it help to get sooth output free of irrelevant frequency spikes. Blackman windowing is a very well-known method. The equation of blackman windowing function looks like this –

$$w[n] = a_0 - a_1 cos\frac{2\pi n}{N} + a_2 cos\frac{4\pi n}{N} \tag{5.1}$$

Where,

$$a_0 = \frac{1-\alpha}{2}, a_1 = \frac{1}{2}, a_2 = \frac{\alpha}{2} \tag{5.2}$$

20% overlapping must be kept among the windows as windowing causes loss in data on each window at the edges. Each window, is then converted to STFT. STFT converts the time domain data to frequency domain. After that, the frequency scale is converted to mal-scale which represent more human like. The mal-scale stated below is represents corresponding Mel for frequencies. A well-known formula to converted Mel from frequency is -

$$m = 2595 log_{10}(1 + \frac{f}{700}) \tag{5.3}$$

Then a spectrogram is constructed using the Mel-scale.

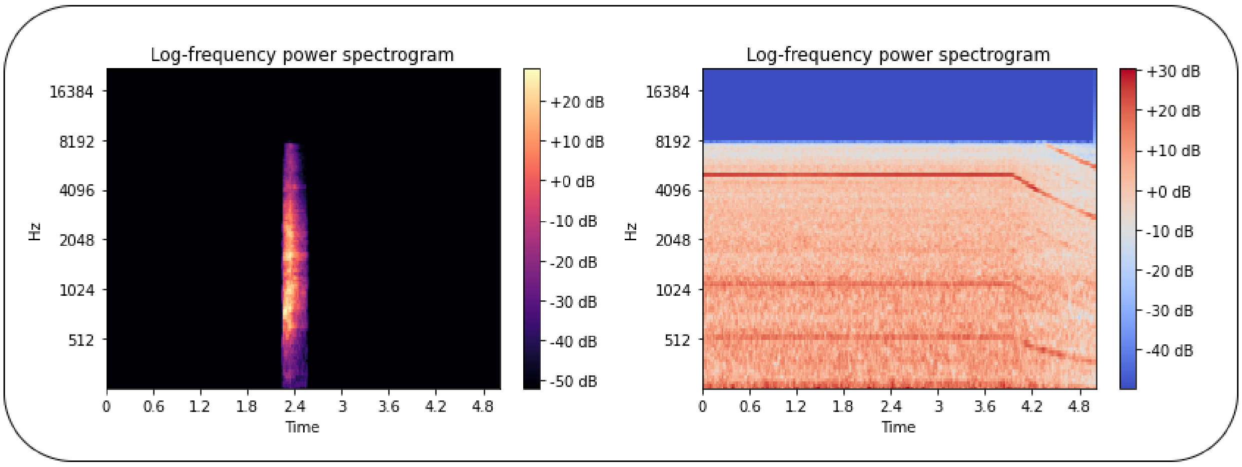


Figure 5.4: Random Audio Mel Spectrogram from ESC-50 Dataset.

Sound preparing was for the most part done utilizing **librosa**.

5.5 Nengo DL Framework

NengoDL is designed as a software framework which can amalgamate neuromorphic modeling and deep learning techniques. It basically enables the users to combine biologically inspired models with deep learning weights such as convolutional neural networks. It can also simulate those combined models profoundly. Moreover, it also has a feature to optimize biologically inspired model parameters by deep learning training. In detailed demostration can be found here at **https://www.nengo.ai/nengo-dl/** [96].

5.5.1 CNN to SNN: Performance Improvement for Weight Conversion Methods

As a huge number of trained networks already exists and neuromorphic computing still remains a fairly new technology with constrains on training and developing SNN models, conversion methods can are a great way to reduce energy waste in training and reuse existing models within neuromorphic hardware by converting those models to SNN. Neural Networks are composed of weights and biases. Optimized weights and biases can further be converted to be used in SNN.

Nengo-DL library offers ways to achieve that. After a trained model has been provided to the model, we can convert it to SNN. Performance of the converted models are not up the mark and performance drops significantly compared to the models, from which the SNN was created. there are 3 suggested methods of improving the performance, which are

- Spiking Aware Training
- Spike Rate Regularization
- Low Pass Filtering

Spiking Aware Training

This approach optimizes the activation function of the converted SNN. The initial model was trained using activation function suitable for non-spiking models. Spiking Neural Networks behave differently, thus, they require activation functions specially optimized for Spiking Neural Networks. The activation functions of the forward pass are all converted to Spiking Aware activation functions, however, the backward pass functions are kept same. This results in enabling the network to learn the temporal features better and improve performance significantly.

Spike Rate Regularization

Regularization is a method or optimizing the neuron firing time which is extremely essential for SNN models. It is because the results of the SNN models are always returned in the time domain. Therefore, neurons firing too fast or slow can affect the performance of the network significantly. Both cases can result in loss of data and deprive us from getting core benefits of SNN. In other words, optimizing neuron firing time can improve the network's performance. In machine learning, regularization is a method of preventing overfitting to adjusting the penalties, thus loss function. In spiking models, regularization can be adjusted to make the network much more robust. This approach can combined be used be with Spike Aware Training as well.

Low Pass Filtering

It is hard to understand what the output spiking layers are firing due to temporally sparse nature of the output itself as it gets very difficult to differentiate. At any given time point, it is almost impossible to determine what is the highest probably output. This problem can be rectified by computing a moving average over time. It smoothens the curve significantly amp; increases the accuracy.

Chapter 6

Experimental Results and Discussion

6.1 Evaluation Matrix

Here, the converted CNN-SNN model and CNN model are evaluated in terms of accuracy and their corresponding loss in this experiment.

$$Accuracy = \frac{m + n}{m + n + p + q} \times 100 \tag{6.1}$$

Where,

m = Correct Negative Predictions

n = Correct Positive Predictions

p = Incorrect Positive Predictions

q = Incorrect Negative Predictions

6.2 Experimental Details

The main objective of this experiment is to train a spiking neural network model on environmental sound recognition dataset. To train a robust solution to train spiking neural network model, we have utilized **nengo dl** framework. The nengo dl framework has a library called *keras_spiking* which mainly converts a traditional sequential model to spiking neurons.

Our experimental dataset has 50 classes with 2000 individual recordings of different sounds surrounded from environment. As we know, environmental sounds have various stationary or non-stationary noises. The main task is train spiking model on the dataset. In order to prepare audio recordings to 2D image, we have utilized the mel spectrograms to convert audio to image. At first, we split the dataset into 5 random individual folds. Random 4 folds are set to train the models and rest of one is set to test the models. At first, we have trained a 2D convolutional neural network model on the prepared data. For training the CNN model, we have used 'ReLU' activation function in the hidden layers. However, global average pooling is also introduced to reduce total number of parameters in the model. The dropout is set to 30%. The trained CNN model performed well on the test data, it shows 90.21% accuracy on the test data with 0.207 loss. For minimizing the loss, we have utilized the Categorical Crossentropy and Adam as an optimizer. Nengo DL mainly converts this CNN model to spiking neural model with a *SpikingActivation* function.

Spiking neural network mainly running over time. So we had to add a temporal dimension to the data before training the SNN model. It doesn't need to have batch shape, instead of this it needs to have step size. Spiking activation function primarily converts any CNN activation function to equivalent spiking neural function and produces a SNN implementation. For an example, if an activation function has 10 output value, then it wrapped to 10Hz output spike rate. However, the spiking neural network has given 74.55% accuracy on the test data with a 1.12 loss. Global pooling average 2D is used to average the output over time across the temporal data.

To improve the performance of the trained SNN model, nengo dl has added few feature to optimize. First one is spiking aware training. It mainly incorporate the spiking feature while training the model. Specifically, it balances the forward pass to spiking activation and backward pass to non-spiking activation. But in our observation, once we trained the spiking aware training model, it reduces the accuracy by 16% but the loss is slightly higher comparing to the standard spiking neural network model.

On the other hand, another feature is suggested by nengo dl called spike rate regularization. In spiking neural models, spiking rate of neurons are very crucial. It is not optimal for a neuron spiking to be very slow or very fast to determine the output value. To control the firing neurons in an optimal way, regularized penalties are used while training the model. It uses kearsSpiking regularizers for non-zero reference. The training accuracy of SNN model with spike rate regularization model is not improvised. However, it reduces the accuracy to around 40% with a massive loss of 5762.20.

Lastly, filtering can be used whenever we work with spiking activation. In low pass filtering method, it mainly work with fewer spikes than the traditional SNN. In this method, aggregating spikes on the data over the time is utilized to produce fewer spike. It is the most extensive way method of optimizing the performance of SNN. But in our experiment, it has performed worst than other SNN methods. It reduces around 51% of accuracy with a 2.01 loss. To train each SNN model, epochs have been set to 120 for better convergence. A summary of our experiment is shown in Table 6.1.

Table 6.1: Performance of CNN and SNN models on Mel-Spectrogram Feature (Test Data)

Models	**Accuracy (%)**	**Loss**	**Val. Accuracy**	**Val. Loss**
Convolutional Neural Network	90.21	0.207	0.7875	0.7035
Spiking Neural Network	74.55	1.121	0.400	5.22
SNN (Spike Aware Training)	58.21	3.81	0.325	7.00
SNN (Spike Rate Regularization)	35.12	5762.20	0.4375	5748.01
SNN (Low Pass Filtering)	23.78	2.01	0.15	4.126

6.2.1 Validation Graphs and Confusion Matrix

In this experiment, we have tested traditional convolutional neural network model and spiking neural network model with various features like aware training, rate regularization and low pass filtering. In this section we have shown validation accuracy and loss graphs for better understanding. We have also shown the confusion matrix of each cases. Confusion matrix is a comprehensive summary of the predicted results. In a concise way, confusion matrix shows how the developed model has been confused while it is making prediction. As we can see in the Figure 6.1 - 6.5 confusion matrix of each obtained models such as CNN, SNN, SNN (Spike Aware Training), SNN (Spike Rate Regularization) and SNN (Low Pass Filtering).

As we can see in the figure 6.1, the CNN validation and training graphs. The validation loss and training loss is almost very close to each other. So we can say the model is quite compatible to the dataset, not over or under fitted. In the Figure 6.2, the SNN validation and training graphs are shown. From the graphs and confusion matrix we can see that the validation and training accuracy have a higher difference between them, after it exceeds 5 epochs. But its validation and training loss are very similar. On the other hand, confusion matrix says that the prediction of the model is not accurate in each class.

However, in the following 6.3-6.5 figures, we can see in the spike aware training model the validation and training accuracy have a large gap but in the validation and training loss they are almost similar. But the prediction matrix is not good at all. In the spike rate regularization training Figure 6.4, we can see that the validation and training accuracy are quite similar and the validation and training loss are very diverse. Its prediction matrix is also not acceptable. In the figure 6.5, we can see that the low pass filtering validation and training accuracy are very much diverse and has a very unrealistic under fitting. But the validation and training loss are very similar. But the prediction matrix is also showing unacceptable result.

So, in our observation CNN model predicts the classes perfectly. The validation training accuracy-loss are also acceptable.

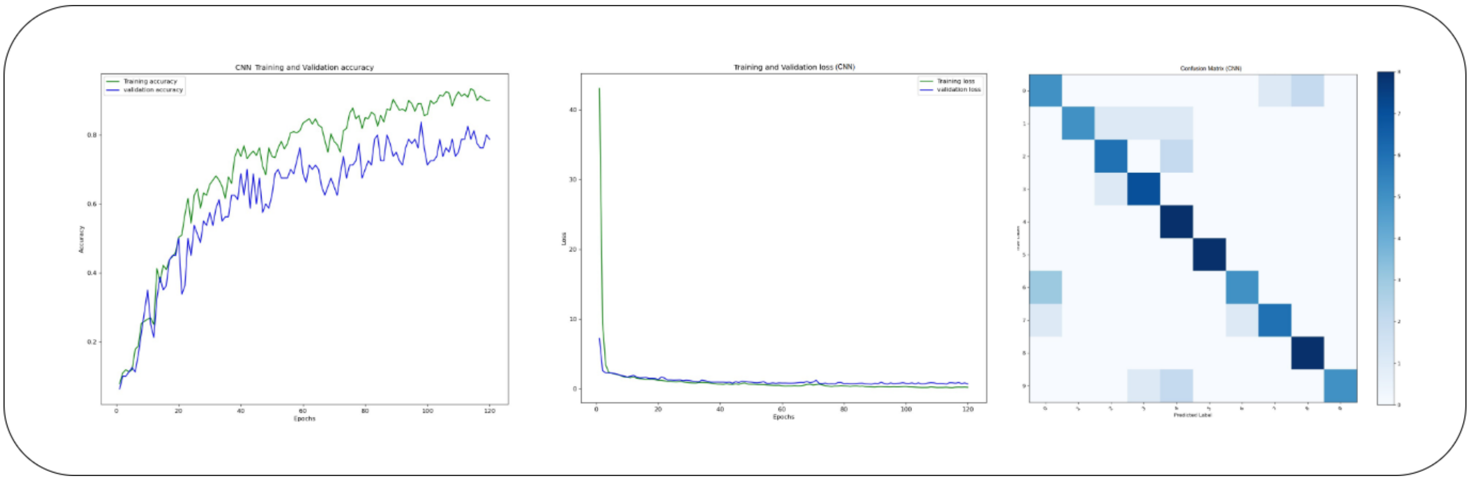


Figure 6.1: Validation, Training Graphs and Confusion Matrix of CNN Model.

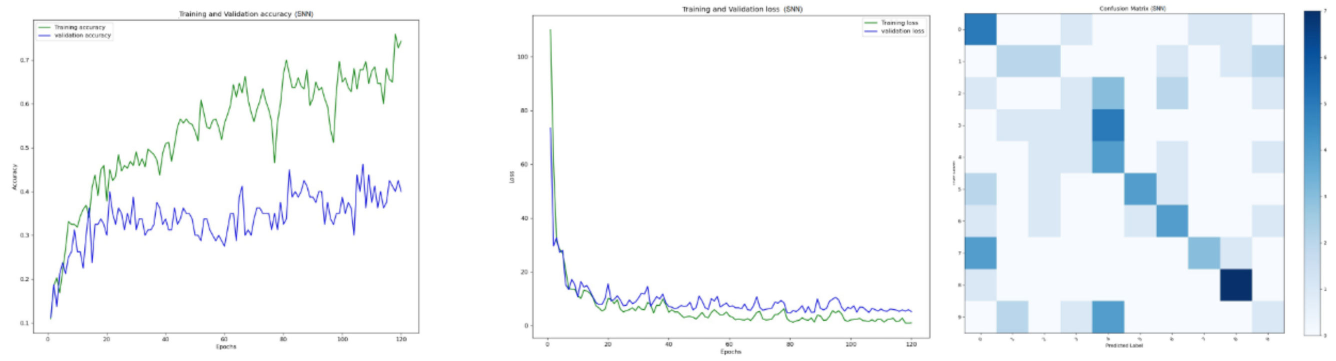


Figure 6.2: Validation, Training Graphs and Confusion Matrix of SNN Model.

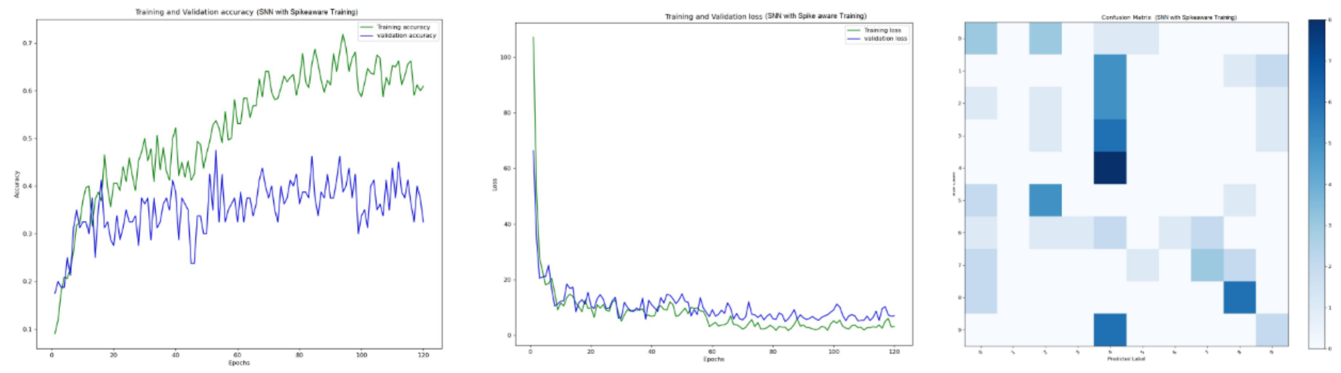


Figure 6.3: Validation, Training Graphs and Confusion Matrix of SNN (Spike Aware Training).

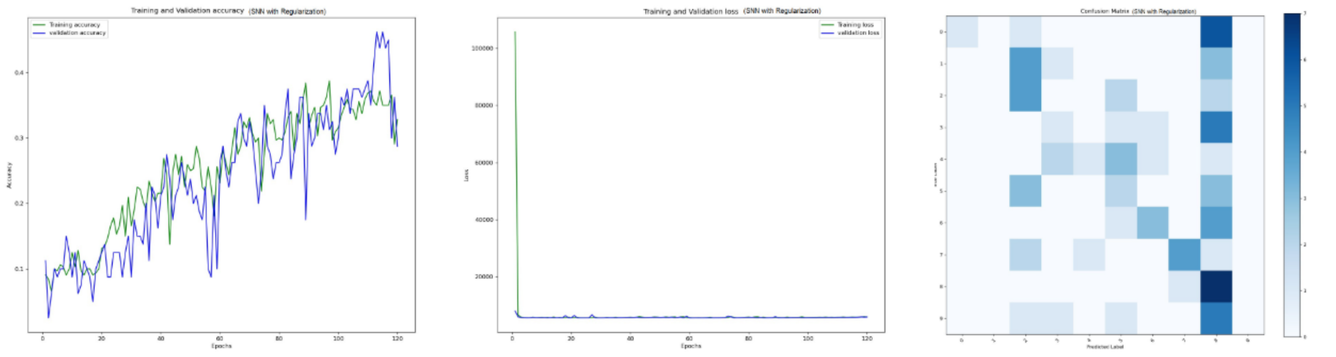


Figure 6.4: Validation, Training Graphs and Confusion Matrix of SNN (Spike Rate Regularization).

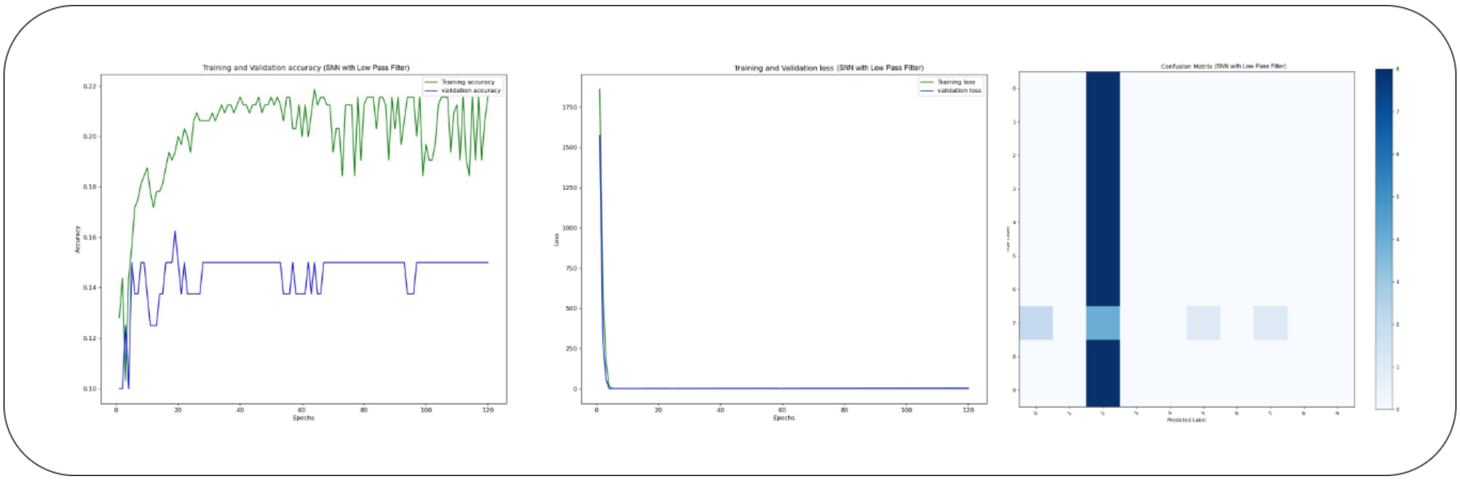


Figure 6.5: Validation, Training Graphs and Confusion Matrix of SNN (Low Pass Filtering).

6.3 Discussion

We are surrounded by several variants of noises. Sometimes crucial sounds can be affected by additional noises. Robust sound recognition is often needed to increment the lifestyle of living individuals and AI systems for well adaption.

In our experiment, our main goal is to implement a human brain structured SNN model for environmental sound recognition. There are mainly three additional SNN variants are deployed to improve the overall recognition accuracy on test data. The main conversion is done end to end SNN from CNN model. The validation result of the trained CNN model is satisfactory. Meanwhile, spike based model has not performed well in this occasion. The spike-based SNN model is mainly robust for temporal signal processing. But there is still a lack of its performance of traditional audio or image data. More research can be done to validate its usefulness and application like deep learning models.

Chapter 7

Conclusion

In this thesis, we explored the progress in the Spiking Neural Networks and Neuromorphic computing. We studied the motivations, architectures and theories regarding the construction of SNNs. We studied different techniques for audio processing, feature extraction methods and progress of audio detection techniques and how they can be aligned with CNN and SNN in particular. We attempted to utilized conversion-based SNN methods with Nengol-DL library to test the performance of SNN in environmental sound recognition. We evaluated the performance of conversion-based SNN from trained CNN models on the Environmental Sound Classificaiton Dataset. We extracted the mel-spectrogram features of the dataset and trained our CNN model. CNN model showed 90.21% accuracy. We constructed conversion-based SNN from the trained model using Nengo-dl dataset. The SNN showed accuracy of 74.55%. We took three approaches to further optimize the performance of the model. We implemented the Spike Aware Training approach, Spike Rate Regularization and Low Pass Filtering. The following methods showed an accuracy ranging from 16% - 51% at most. Overall, the attempted approaches did not show any promising results. But more research can be done to improve the validation of spike-based models like deep learning models.

Bibliography

[1] Gygi, B., Shafiro, V. Development of the Database for Environmental Sound Research and Application (DESRA): Design, Functionality, and Retrieval Considerations. J AUDIO SPEECH MUSIC PROC. 2010, 654914 (2010). https://doi.org/10.1155/2010/654914

[2] Alías, F.; Socoró, J.C.; Sevillano, X. A Review of Physical and Perceptual Feature Extraction Techniques for Speech, Music and Environmental Sounds. Appl. Sci. 2016, 6, 143. https://doi.org/10.3390/app6050143

[3] S. Ray, "A Quick Review of Machine Learning Algorithms," 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), 2019, pp. 35-39, doi: 10.1109/COMITCon.2019.8862451.

[4] Dhall D., Kaur R., Juneja M. (2020) Machine Learning: A Review of the Algorithms and Its Applications. In: Singh P., Kar A., Singh Y., Kolekar M., Tanwar S. (eds) Proceedings of ICRIC 2019. Lecture Notes in Electrical Engineering, vol 597. Springer, Cham. https://doi.org/10.1007/978-3-030-29407-6_5

[5] Kodinariya, T., Makwana, P.R. (2013). Review on determining number of Cluster in K-Means Clustering.

[6] Sharma, V., Rai, S., Dev, A. (2012). A Comprehensive Study of Artificial Neural Networks.

[7] Justin Salamon, Christopher Jacoby, and Juan Pablo Bello. 2014. A Dataset and Taxonomy for Urban Sound Research. In Proceedings of the 22nd ACM international conference on Multimedia (MM '14). As-

sociation for Computing Machinery, New York, NY, USA, 1041–1044. DOI:https://doi.org/10.1145/2647868.2655045

[8] S. Chaudhuri and B. Raj, ”Unsupervised hierarchical structure induction for deeper semantic analysis of audio,” 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 2013, pp. 833-837, doi: 10.1109/ICASSP.2013.6637765.

[9] Y. -J. Park and H. -S. Cho, ”An Experiment of Sound Recognition using Machine Learning,” 2020 IEEE International Conference on Consumer Electronics - Asia (ICCE-Asia), 2020, pp. 1-3, doi: 10.1109/ICCE-Asia49877.2020.9277368.

[10] Vasileios Bountourakis, Lazaros Vrysis, and George Papanikolaou. 2015. Machine Learning Algorithms for Environmental Sound Recognition: Towards Soundscape Semantics. In Proceedings of the Audio Mostly 2015 on Interaction With Sound. Association for Computing Machinery, New York, NY, USA, Article 5, 1–7. DOI:https://doi.org/10.1145/2814895.2814905

[11] Michael Cowling, Renate Sitte, Comparison of techniques for environmental sound recognition, Pattern Recognition Letters, Volume 24, Issue 15, 2003, Pages 2895-2907, ISSN 0167-8655, https://doi.org/10.1016/S0167-8655(03)00147-8.

[12] Cowling, M.A. (2004). Non-Speech Environmental Sound Classification System for Autonomous Surveillance.

[13] Uzkent, B., Barkana, B., Çevikalp, H. (2012). NON-SPEECH ENVIRONMENTAL SOUND CLASSIFICATION USING SVMS WITH A NEW SET OF FEATURES.

[14] S. Chachada and C. -. J. Kuo, ”Environmental sound recognition: A survey,” 2013 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, 2013, pp. 1-9, doi: 10.1109/APSIPA.2013.6694338.

[15] J. Salamon and J. P. Bello, ”Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification,” in IEEE Signal Processing Letters, vol. 24, no. 3, pp. 279-283, March 2017, doi: 10.1109/LSP.2017.2657381.

[16] A. Khamparia, D. Gupta, N. G. Nguyen, A. Khanna, B. Pandey and P. Tiwari, "Sound Classification Using Convolutional Neural Network and Tensor Deep Stacking Network," in IEEE Access, vol. 7, pp. 7717-7727, 2019, doi: 10.1109/ACCESS.2018.2888882.

[17] S. Chu, S. Narayanan and C. -. J. Kuo, "Environmental Sound Recognition With Time–Frequency Audio Features," in IEEE Transactions on Audio, Speech, and Language Processing, vol. 17, no. 6, pp. 1142-1158, Aug. 2009, doi: 10.1109/TASL.2009.2017438.

[18] J. Salamon and J. P. Bello, "Unsupervised feature learning for urban sound classification," 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2015, pp. 171-175, doi: 10.1109/ICASSP.2015.7177954.

[19] J. Salamon and J. P. Bello, "Feature learning with deep scattering for urban sound analysis," 2015 23rd European Signal Processing Conference (EUSIPCO), 2015, pp. 724-728, doi: 10.1109/EUSIPCO.2015.7362478.

[20] Dash S.S., Nayak S.K., Mishra D. (2021) A Review on Machine Learning Algorithms. In: Mishra D., Buyya R., Mohapatra P., Patnaik S. (eds) Intelligent and Cloud Computing. Smart Innovation, Systems and Technologies, vol 153. Springer, Singapore. https://doi.org/10.1007/978-981-15-6202-0_51

[21] S. B. Kotsiantis. 2007. Supervised Machine Learning: A Review of Classification Techniques. In Proceedings of the 2007 conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies. IOS Press, NLD, 3–24.

[22] Zgheib, W.A., Barbar, A. (2017). A Study using Support Vector Machines to Classify the Sentiments of Tweets. International Journal of Computer Applications, 170, 8-12.

[23] Sutton R.S. (1992) Introduction: The Challenge of Reinforcement Learning. In: Sutton R.S. (eds) Reinforcement Learning. The Springer International Series in Engineering and Computer Science (Knowledge Representation, Learning and Expert Systems), vol 173. Springer, Boston, MA. https://doi.org/10.1007/978-1-4615-3618-5_1

[24] Alaeddine, H., Jihene, M. Deep network in network. Neural Comput Applic 33, 1453–1465 (2021). https://doi.org/10.1007/s00521-020-05008-0

[25] Q. Yu, R. Yan, H. Tang, K. C. Tan and H. Li, ”A Spiking Neural Network System for Robust Sequence Recognition,” in IEEE Transactions on Neural Networks and Learning Systems, vol. 27, no. 3, pp. 621-635, March 2016, doi: 10.1109/TNNLS.2015.2416771.

[26] Charles, E. (2006). Supervised and unsupervised weight and delay adaptation learning in temporal coding spiking neural networks.

[27] Paugam-Moisy H., Bohte S. (2012) Computing with Spiking Neuron Networks. In: Rozenberg G., Bäck T., Kok J.N. (eds) Handbook of Natural Computing. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-92910-9_10

[28] Ahmed, F.Y., Yusob, B., Hamed, H.N. (2014). Computing with Spiking Neuron Networks A Review. SOCO 2014.

[29] Jolivet R., J. T., Gerstner W. (2003) The Spike Response Model: A Framework to Predict Neuronal Spike Trains. In: Kaynak O., Alpaydin E., Oja E., Xu L. (eds) Artificial Neural Networks and Neural Information Processing — ICANN/ICONIP 2003. ICANN 2003, ICONIP 2003. Lecture Notes in Computer Science, vol 2714. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-44989-2_101

[30] K. He, X. Zhang, S. Ren and J. Sun, ”Deep Residual Learning for Image Recognition,” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.

[31] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, ”Gradient-based learning applied to document recognition,” in Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi: 10.1109/5.726791.

[32] Comparison of techniques for environmental sound recognition (2003).

[33] A COMPARISON OF DEEP LEARNING METHODS FOR ENVIRONMENTAL SOUND DETECTION (2017).

[34] Spectral images based environmental sound classification using CNN with meaningful data augmentation (2020).

[35] End-to-End Environmental Sound Classification using a 1D Convolutional Neural Network.

[36] Dilated Convolution Neural Network with LeakyReLU for Environmental Sound Classification.

[37] Improved Deep CNN with Reduced Parameters for Automatic Identification of Environmental Sounds.

[38] Sazlı, Murat Hüsnü. ”A brief review of feed-forward neural networks.” (2006).

[39] Wang, Sun-Chong. ”Artificial neural network.” Interdisciplinary computing in java programming. Springer, Boston, MA, 2003. 81-100.

[40] Bre, Facundo, Juan M. Gimenez, and Víctor D. Fachinotti. ”Prediction of wind pressure coefficients on building surfaces using artificial neural networks.” Energy and Buildings 158 (2018): 1429-1441.

[41] Tawfiq, Luma NM, and Ashraf AT Hussein. ”Design feed forward neural network to solve singular boundary value problems.” International Scholarly Research Notices 2013 (2013).

[42] Deb, A. K. ”Introduction to soft computing techniques: artificial neural networks, fuzzy logic and genetic algorithms.” Soft Computing in Textile Engineering. Woodhead Publishing, 2011. 3-24.

[43] Escamilla-Garcia, Axel, et al. ″Applications of Artificial Neural Networks in Greenhouse Technology and Overview for Smart Agriculture Development.″ Applied Sciences 10.11 (2020): 3835.

[44] Shrestha, Ajay, and Ausif Mahmood. ″Review of deep learning algorithms and architectures.″ IEEE Access 7 (2019): 53040-53065.

[45] Miralles-Pechuán, Luis, et al. ″A methodology based on Deep Learning for advert value calculation in CPM, CPC and CPA networks.″ Soft Computing 21.3 (2017): 651-665.

[46] Du, Xuedan, et al. ″Overview of deep learning.″ 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC). IEEE, 2016.

[47] Grüning, André, and Sander M. Bohte. ″Spiking neural networks: Principles and challenges.″ ESANN. 2014.

[48] Long, Lyle, and Guoliang Fang. ″A review of biologically plausible neuron models for spiking neural networks.″ AIAA Infotech@ Aerospace 2010 (2010): 3540.

[49] A. L. Hodgkin., and A. F. Huxley., "A quantitative description of membrane current and application to conduction and excitation in nerve," J. Physiol., Vol. 117, pp. 500–544, 1954.

[50] Izhikevich, E.M., "Neural excitability, spiking, and bursting" International Journal of Bifurcation and Chaos Vol. 10, 2000, pp. 1171–1266.

[51] van Schaik, André, et al. ″A log-domain implementation of the Izhikevich neuron model.″ Proceedings of 2010 IEEE International Symposium on Circuits and Systems. IEEE, 2010.

[52] Wilson, H. R., Spikes, Decisions, and Actions: The Dynamical Foundations of Neuroscience, 1st ed., Oxford Univ.Press, Oxford,1999, Chaps. 5,6,7.

[53] Tal, Doron, and Eric L. Schwartz. ”Computing with the leaky integrate-and-fire neuron: logarithmic computation and multiplication.” Neural computation 9.2 (1997): 305-318.

[54] Xiao, Rong, et al. ”Spike-based encoding and learning of spectrum features for robust sound recognition.” Neurocomputing 313 (2018): 65-73

[55] Y. Dan and M.-M. Poo, “Spike timing-dependent plasticity: from synapse to perception,” Physiological reviews, vol. 86, no. 3, pp. 1033–1048, 2006.

[56] S. M. Bohte, J. N. Kok, and H. La Poutre, “Error Backpropagation In temporally encoded networks of spiking neurons,” Neurocomputing, vol. 48, no. 1, pp. 17–37, 2002.

[57] W. Gerstner and W. M. Kistler, Spiking neuron models: Single Neurons, populations, plasticity. Cambridge University Press, 2002

[58] F. Ponulak , A. Kasi ´nski , Supervised learning in spiking neural networks with resume: sequence learning, classification, and spike shifting, Neural Comput. 22 (22) (2010) 467–510 .

[59] F. Ponulak , Resume–new supervised learning method for spiking neural networks, Technical Report, Institute of Control and Information Engineering, Pozno ´n University of Technology, 2005 .

[60] R. Gütig , H. Sompolinsky , The tempotron: a neuron that learns spike tim- ing-based decisions, Nat. Neurosci. 9 (3) (2006) 420–428 .

[61] Masquelier, Timothee, Rudy Guyonneau, and Simon J. Thorpe. ”Spike timing dependent plasticity finds the start of repeating patterns in continuous spike trains.” PloS one 3.1 (2008): e1377.

[62] K. Choi, G. Fazekas, M. Sandler and K. Cho, ”Convolutional recurrent neural networks for music classification,” 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2017, pp. 2392-2396, doi: 10.1109/ICASSP.2017.7952585.

[63] S. L. Ullo, S. K. Khare, V. Bajaj and G. R. Sinha, ”Hybrid Computerized Method for Environmental Sound Classification,” in IEEE Access, vol. 8, pp. 124055-124065, 2020, doi: 10.1109/ACCESS.2020.3006082.

[64] Nwankpa, Chigozie Ijomah, Winifred Gachagan, Anthony Marshall, Stephen. (2018). Activation Functions: Comparison of trends in Practice and Research for Deep Learning.

[65] Glorot, X. Bengio, Y.. (2010). Understanding the difficulty of training deep feedforward neural networks. Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, in Proceedings of Machine Learning Research 9:249-256 Available from http://proceedings.mlr.press/v9/glorot10a.html .

[66] Dauphin, Y.N., Fan, A., Auli, M. Grangier, D.. (2017). Language Modeling with Gated Convolutional Networks. Proceedings of the 34th International Conference on Machine Learning, in Proceedings of Machine Learning Research 70:933-941 Available from http://proceedings.mlr.press/v70/dauphin17a.html .

[67] Maas, Andrew L.. “Rectifier Nonlinearities Improve Neural Network Acoustic Models.” (2013).

[68] Ramachandran, P., Zoph, B., Le, Q.V. (2018). Searching for Activation Functions. ArXiv, abs/1710.05941.

[69] Strain TJ, McDaid LJ, McGinnity TM, Maguire LP, Sayers HM. An STDP training algorithm for a spiking neural network with dynamic threshold neurons. Int J Neural Syst. 2010 Dec;20(6):463-80. doi: 10.1142/S0129065710002553. PMID: 21117270.

[70] Wofgang Maas. 1997. Networks of spiking neurons: the third generation of neural network models. Trans. Soc. Comput. Simul. Int. 14, 4 (Dec. 1997), 1659–1671.

[71] Lüscher, C., Malenka, R. C. (2012). NMDA receptor-dependent long-term potentiation and long-term depression (LTP/LTD). Cold Spring Harbor perspectives in biology, 4(6), a005710. https://doi.org/10.1101/cshperspect.a005710

[72] Diehl PU, Cook M. Unsupervised learning of digit recognition using spike-timing-dependent plasticity. Front Comput Neurosci. 2015 Aug 3;9:99. doi: 10.3389/fncom.2015.00099. PMID: 26941637; PMCID: PMC4522567.

[73] R. Chen, H. Ma, S. Xie, P. Guo, P. Li and D. Wang, ”Fast and Efficient Deep Sparse Multi-Strength Spiking Neural Networks with Dynamic Pruning,” 2018 International Joint Conference on Neural Networks (IJCNN), 2018, pp. 1-8, doi: 10.1109/IJCNN.2018.8489339.

[74] Schuman, C.D., Potok, T., Patton, R., Birdwell, J., Dean, M., Rose, G., Plank, J. (2017). A Survey of Neuromorphic Computing and Neural Networks in Hardware. ArXiv, abs/1705.06963.

[75] Cao, Y., Chen, Y. Khosla, D. Spiking Deep Convolutional Neural Networks for Energy-Efficient Object Recognition. Int J Comput Vis 113, 54–66 (2015). https://doi.org/10.1007/s11263-014-0788-3

[76] Xiao, R., Yu, Q., Yan, R., Tang, H. (2019). Fast and Accurate Classification with a Multi-Spike Learning Algorithm for Spiking Neurons. IJCAI.

[77] Apicella, A., Isgrò, F., Prevete, R. (2019). A simple and efficient architecture for trainable activation functions. Neurocomputing, 370, 1-15.

[78] Kingma, D.P., Ba, J. (2015). Adam: A Method for Stochastic Optimization. CoRR, abs/1412.6980.

[79] R. Wang, C. S. Thakur, T. J. Hamilton, J. Tapson and A. van Schaik, ”A stochastic approach to STDP,” 2016 IEEE International Symposium on Circuits and Systems (ISCAS), 2016, pp. 2082-2085, doi: 10.1109/ISCAS.2016.7538989.

[80] Yan Xu, Xiaoqin Zeng, Lixin Han, Jing Yang, A supervised multi-spike learning algorithm based on gradient descent for spiking neural net-

works, Neural Networks, Volume 43, 2013, Pages 99-113, ISSN 0893-6080, https://doi.org/10.1016/j.neunet.2013.02.003.

[81] Nwankpa, C., Ijomah, W., Gachagan, A., Marshall, S. (2018). Activation Functions: Comparison of trends in Practice and Research for Deep Learning. ArXiv, abs/1811.03378.

[82] Kim, S.J., Park, S., Na, B., Yoon, S. (2019). Spiking-YOLO: Spiking Neural Network for Real-time Object Detection. ArXiv, abs/1903.06530.

[83] Thiele, J.C., Bichler, O., Dupret, A. (2020). SpikeGrad: An ANN-equivalent Computation Model for Implementing Backpropagation with Spikes. ArXiv, abs/1906.00851.

[84] Liu, Z., Chotibut, T., Hillar, C., Lin, S. (2020). Biologically Plausible Sequence Learning with Spiking Neural Networks. AAAI.

[85] Jordan, J., Schmidt, M., Senn, W., Petrovici, M. (2020). Evolving to learn: discovering interpretable plasticity rules for spiking networks. arXiv: Neurons and Cognition.

[86] Wunderlich, T., Pehle, C. (2020). EventProp: Backpropagation for Exact Gradients in Spiking Neural Networks. ArXiv, abs/2009.08378.

[87] Caporale N, Dan Y. Spike timing-dependent plasticity: a Hebbian learning rule. Annu Rev Neurosci. 2008;31:25-46. doi: 10.1146/annurev.neuro.31.060407.125639. PMID: 18275283.

[88] L. P. Maguire, T. M. McGinnity, B. Glackin, A. Ghani, A. Belatreche, and J. Harkin. 2007. Challenges for large-scale implementations of spiking neural networks on FPGAs. ¡i¿Neurocomput.¡/i¿ 71, 1–3 (December, 2007), 13–29. DOI:https://doi.org/10.1016/j.neucom.2006.11.029

[89] Grüning, A., Bohté, S. (2014). Spiking Neural Networks: Principles and Challenges. ESANN.

[90] G. Kim, K. Kim, S. Choi, H. J. Jang and S. -O. Jung, ”Area- and Energy-Efficient STDP Learning Algorithm for Spiking Neural Network SoC,” in IEEE Access, vol. 8, pp. 216922-216932, 2020, doi: 10.1109/ACCESS.2020.3041946.

[91] J. Schemmel, A. Grubl, K. Meier and E. Mueller, ”Implementing Synaptic Plasticity in a VLSI Spiking Neural Network Model,” The 2006 IEEE International Joint Conference on Neural Network Proceedings, 2006, pp. 1-6, doi: 10.1109/IJCNN.2006.246651.

[92] Karol J. Piczak. 2015. ESC: Dataset for Environmental Sound Classification. In Proceedings of the 23rd ACM international conference on Multimedia (MM ’15). Association for Computing Machinery, New York, NY, USA, 1015–1018. DOI:https://doi.org/10.1145/2733373.2806390

[93] Tian, D. (2013). on Image Feature Extraction and Representation Techniques.

[94] Zohaib Mushtaq, Shun-Feng Su, Environmental sound classification using a regularized deep convolutional neural network with data augmentation, Applied Acoustics, Volume 167, 2020, 107389, ISSN 0003-682X, https://doi.org/10.1016/j.apacoust.2020.107389.

[95] Huzaifah, M. (2017). Comparison of Time-Frequency Representations for Environmental Sound Classification using Convolutional Neural Networks. ArXiv, abs/1706.07156.

[96] K. J. Piczak, ”Environmental sound classification with convolutional neural networks,” 2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP), 2015, pp. 1-6, doi: 10.1109/MLSP.2015.7324337.

Appendices

Appendix A

Code Attachments

A.1 CNN-SNN Conversion Model with Keras-Spiking

```
n_steps = 10
train_sequences = np.tile(x_train[:, None], (1, n_steps, 1, 1))
test_sequences = np.tile(x_val[:, None], (1, n_steps, 1, 1))

spiking_model = tf.keras.Sequential(
    [
        # add temporal dimension to the input shape; we can set it to None,
        # to allow the model to flexibly run for different lengths of time
        tf.keras.layers.Reshape((-1, 128,318), input_shape=(128,318,3)),


        # we can use Keras' TimeDistributed wrapper to allow the Dense layer
        # to operate on temporal data
        tf.keras.layers.TimeDistributed(tf.keras.layers.Dense(128)),

        # replace the "relu" activation in the non-spiking model with a
        # spiking equivalent
        keras_spiking.SpikingActivation("relu", spiking_aware_training=False),
        # use average pooling layer to average spiking output over time
        tf.keras.layers.GlobalAveragePooling2D(),
        tf.keras.layers.Dense(10),
    ]
)

# train the model, identically to the non-spiking version,
# except using the time sequences as inputs
p = train(spiking_model, train_sequences, test_sequences)
```

A.2 Spike Aware Training

```
spikeaware_model = tf.keras.Sequential(
    [
        tf.keras.layers.Reshape((-1, 128,318), input_shape=(128,318,3)),
        tf.keras.layers.TimeDistributed(tf.keras.layers.Dense(128)),
        # set spiking_aware training and a moderate dt
        keras_spiking.SpikingActivation("relu", dt=0.01, spiking_aware_training=
    True),
        tf.keras.layers.GlobalAveragePooling2D(),
        tf.keras.layers.Dense(10),
    ]
)
```

```

p = train(spikeaware_model, train_sequences, test_sequences)
```

A.3 Spike Rate Regularization

```
regularized_model = tf.keras.Sequential(
    [
        tf.keras.layers.Reshape((-1, 128,318), input_shape=(128,318,3)),
        tf.keras.layers.TimeDistributed(tf.keras.layers.Dense(128)),
        keras_spiking.SpikingActivation(
            "relu",
            dt=0.01,
            spiking_aware_training=True,
            # add activity regularizer to encourage spike rates between 10 and 20
    Hz
            activity_regularizer=keras_spiking.regularizers.L2(
                l2=1e-4, target=(10, 20)
            ),
        ),
        tf.keras.layers.GlobalAveragePooling2D(),
        tf.keras.layers.Dense(10),
    ]
)

p = train(regularized_model, train_sequences, test_sequences)
```

A.4 Low Pass Filtering

```
keras_spiking.default.dt = 0.01

filtered_model = tf.keras.Sequential(
    [
        tf.keras.layers.Reshape((-1, 128 * 318), input_shape=(128,318,3)),
        tf.keras.layers.TimeDistributed(tf.keras.layers.Dense(128)),
        keras_spiking.SpikingActivation("relu", spiking_aware_training=True),
        # add a lowpass filter on output of spiking layer
        keras_spiking.Lowpass(tau=0.1, return_sequences=False),
        tf.keras.layers.Dense(10),
    ]
)

p=train(filtered_model, train_sequences, test_sequences)
```