Understanding localized burst trigger patterns in developing neural networks using deep learning

Smriti Singh

A thesis

submitted in partial fulfillment of the

requirements for the degree of

Master of Science in Computer Science & Software Engineering

University of Washington

2020

Committee:

Michael Stiber, Chair

Erika Parsons

Min Chen

Program Authorized to Offer Degree:
Computing and Software Systems

University of Washington

Abstract

Understanding localized burst trigger patterns in developing neural networks using deep learning

Smriti Singh

Chair of the Supervisory Committee:
Michael Stiber
Computing and Software Systems

The brain is a complex, interconnected network par excellence. It is made up of intricate connections of neurons, responsible for transmission of signals throughout the body. Over time, experimental studies have illustrated a variety of behavioral features of network dynamics, ranging from stochastic spiking to synchronized bursting observed in the living preparations of neuronal cultures.

A fundamental feature of developing neural circuits is the presence of spontaneous network activity. Such spontaneous activity plays putative roles ranging from synaptic development and maintenance to anticipatory states which assist animals in reaching rapid decisions with limited sensory input. Understanding the mechanisms of spontaneously generated activity and interaction patterns between neurons are, therefore, issues of substantial importance.

But while changes in the dynamics of coordinated, spontaneous spiking activity have been investigated in different in vivo and in vitro neuronal systems, the mechanisms underlying the generation of these spontaneous spiking activities still remain unclear.

Computational simulations help researchers to gain a more detailed understanding of activity patterns in large networks. The BrainGrid simulator is a neural simulator, based on a leaky integrate-and-fire computational model and developed in UW Bothell, that allows us to perform detailed analysis of the effects of model parameters on burst shape and timing, their changes, their patterns and the inter-relationship among these behaviors, gross network structure, and model parameters. Researchers draw inferences about dynamics of these networks from simulation results. These simulations' high spatiotemporal resolution and long duration produce data that, in terms of both quantity and complexity, challenge our interpretative abilities.

Therefore, this thesis focuses on uncovering underlying patterns during spontaneous activities using various machine learning and deep learning techniques and presents their comparison based on different slices from the data. To derive patterns in activity initiation, we also implement model interpretability analysis using SHAPley Analysis to obtain hidden patterns in data and understand the cause of a model's decision.

Concisely, this thesis applies AI techniques to clarify localized activity patterns that trigger network bursts in cortical neural networks.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to thank my supervisor, Professor Michael Stiber, for instilling curiosity in me with such a fascinating field and guiding me throughout my research. I thank my committee members, Professor Min Chen, and Professor Erika Parsons, for their precious time and inspiring comments. I would also like to thank the members of the UW Bothell Biocomputing Lab (BCL) for their helpful discussions and to Sinchai Delong for their timely technical support. I am very grateful to be surrounded by excellent professors and advisors at the University of Washington Bothell. I am forever grateful for all the support and encouragement my family has provided me throughout my graduate journey.

DEDICATION

To my beloved parents, Sudhir Kumar Singh and Namrata Singh, to my loving brother, Sanchit Singh and my best friend, Rishabh Chauhan, for being my pillars of support.

Chapter 1. INTRODUCTION

1.1 PURPOSE OF THIS WORK

A fundamental feature of developing neural circuits is the presence of spontaneous network activity. These activities are thought to have an important role in the development of parts of our body, for example, in developing sensory organs, the retina and cochlea, in development of our spinal cord, etc. [13]. Additionally, in the forebrain structures such as the hippocampus and the neocortex, as well as in the hindbrain, the midbrain, and the cerebellum, it has been postulated that spontaneous activity contributes to the development of local circuits in these brain regions [13].

A network burst or population burst is a type of spontaneous activity defined as a synchronized spiking event that involves most or all the neurons throughout the network. The goal of this thesis is to refine and apply a set of deep learning techniques to investigate the underlying patterns that may arise during network bursts initiation from a large amount of spike events. Through this research, our goal is to attain a certain level of clarity in the following research areas in computational neuroscience:

- How does the background activity that happens before network bursts, contribute to initiating/ triggering the network bursts?
- What kind of visual patterns lie during these whole network bursts?
- Apply techniques to confirm the previous work's performance.
- Are we able to correctly mark the beginning of a burst?
- How can we efficiently apply machine learning techniques to derive insights from this high-dimensional network spiking data?

The methods of data analysis proposed in this thesis are also intended to serve to help researchers to gain an intuitive and more analytical understanding of the principal activity patterns in large networks. We are interested in applying AI techniques to see if we can find localized patterns of activity that trigger network bursts initiation in the spike train data.

Chapter 2. BACKGROUND: NEUROSCIENCE BASICS

Computational neuroscience is the field of study in which mathematical tools and theories are used to investigate brain function, and the goal is to explain, in computational terms, how brains generate behaviors. It provides tools and methods for "characterizing **what** nervous systems do, determining **how** they function, and understanding **why** they operate in certain ways" [29]. With advancements in research in neuroscience, new treatments for devastating brain diseases can emerge from a deeper understanding of brain circuits [2].

Researchers continue to explore the mechanics behind a healthy brain that functions quickly and automatically – at the speed of thought. Truly understanding a circuit requires identifying and characterizing the components of these networks, defining their connections with one another, monitoring, and recording their activity patterns etc. Therefore, accurate and informative mapping of this human connectome from the biological concepts to a more analytical setting has become a central goal of neuroscience.

The following sections in this chapter provide a very brief introduction to several elementary notions of neuroscience and important dynamical properties of neural networks. The aim of this chapter is to provide the reader with the minimum amount of information necessary to relate the topics covered by this thesis.

2.1 NERVOUS SYSTEM BIOLOGICAL BASICS

2.1.1 *The Nervous System*

Neuroscience is the scientific study of the nervous system (the brain, spinal cord, peripheral nervous system) and its functions. The human brain contains approximately one hundred billion

neurons that have one hundred trillion connections with each other. The nervous system is organized into two main parts, the *central nervous system* (CNS) and the *peripheral nervous system* (PNS) [4]. The CNS is the processing center of the body and consists of the brain and the spinal cord. PNS includes a large system of nerves which are long fibers that connect the CNS to every other part of the body.

The nervous system performs three main functions: collecting information from sensory receptors that monitor the body's internal and external conditions, integrating this information and evaluating it for decision making, and stimulating motor neurons to carry these signals throughout the body. Based on their roles, neurons found in the human nervous system can be divided into three classes: sensory neurons that convert external stimuli from the surroundings and send signals to the CNS, motor neurons that receive signals from the CNS and covey commands to muscles, organs and glands, and interneurons that connect one neuron to another within the CNS [18].

2.1.2 *Neuron*

The core component of the nervous system is the *neuron* or nerve cell, the "brain cells" of popular language. A neuron is an electrically excitable cell that processes and transmits information by electrochemical signaling. This electrochemical signaling refers to the mechanism of activation of the neurons when they receive input from neighboring neurons through an electrical impulse. These electrical impulses will be explained in the upcoming sections.

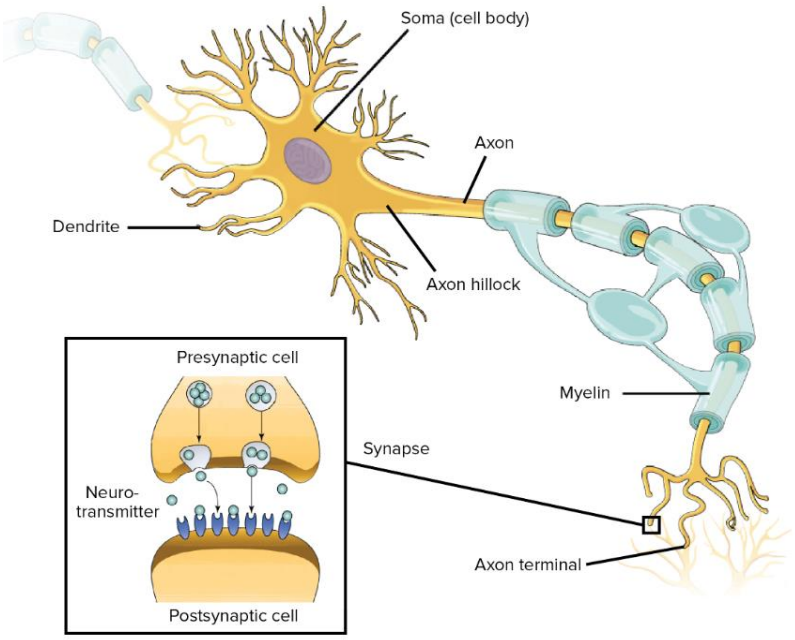

Figure 2.1. Schematic representation of the parts of a neuron [30].

All neurons have three essential parts: a cell body, an axon, and dendrites, as can be seen from Figure 2.1 [6]. One of the major defining features of neuronal cells is the polarized transmission of information through axons and dendrites [8]. The dendrites play the role of the `input device' that collects signals from other neurons and transmits them to the soma. The soma is the `central processing unit' that performs an important non-linear processing step. If the total input exceeds a certain threshold, then an output signal is generated. The output signal is taken over by the `output device', the axon, which delivers the signal to other neurons [7].

2.1.3 *Neurons and Synapses*

Neurons are the building blocks of the brain that communicate with each other by sending electrical signals. In neuroscience, the sending neuron and the receiving neuron are commonly known as presynaptic neuron and postsynaptic neuron, respectively. The site where the axon of a presynaptic neuron contacts the dendrite (or soma) of a postsynaptic cell is the synapse.

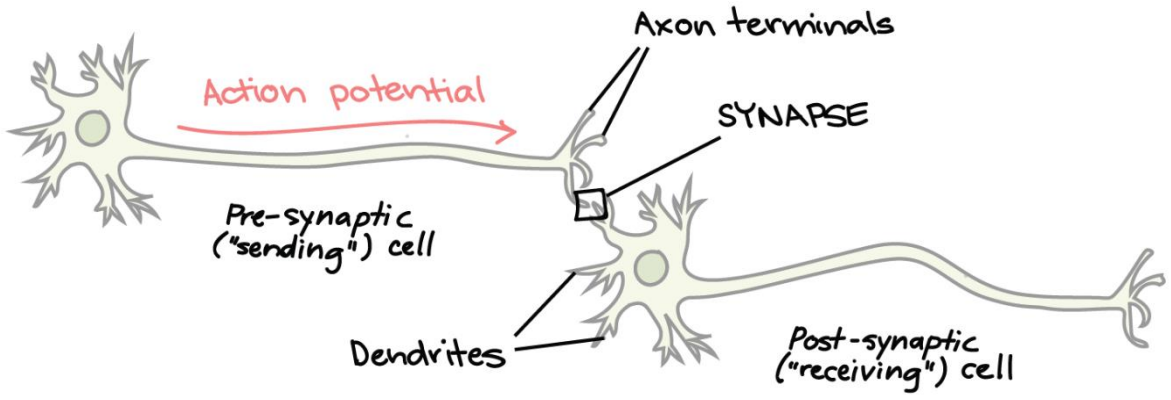

Figure 2.2. How an action potential travels through a neuron [31].

The junction between two neurons is called a synapse, as shown in Figure 2.2. As far as we know, there are more than 100 billion neurons in the human brain and each of them can have more than 10,000 synaptic connections with other neurons through synapses [7].

2.1.4 *Neuronal Spikes*

The neuronal signals that are transmitted among neurons consist of short electrical pulses and can be observed by placing a fine electrode close to the soma or axon of a neuron. When a neuron is stimulated enough, it fires an electrical impulse that moves along its axon to its neighboring neuron. There is only this one type of signal that they can send that transmits at a uniform strength and speed. These neuron impulses are called action potentials, or neuronal spikes. They have an amplitude of about 100 mV and typically a duration of 1-2 ms. A chain of action potentials emitted by a single neuron is called a spike train — a sequence of stereotyped events which occur at regular or irregular intervals. The frequency and timing of spiking activities reveal interesting properties of a neural network, as will be studied in the Methods chapter.

2.2 NEURONAL NETWORK BEHAVIORS

An *in vitro* culture of the neuronal network is a cell culture of neurons that is used as a model to study the central nervous system. The ability to produce *in vitro* cultures of neuronal cells has been fundamental to advancing our understanding of the functioning of the nervous system [8]. One thread of brain research today focuses on understanding neuronal activities from these cultures as they capture fundamental aspects of higher brain functions, like neuronal spiking, connectivity, activity stabilization etc. Such preparations allow investigation of network development, activity, plasticity, responses to stimuli, the effects of pharmacological agents, etc. [68]. Over time, experimental studies have given us a variety of observations on cortical neural networks. Some of the fundamental behaviors of network dynamics are discussed in this section.

2.2.1 *Spontaneous Network activity as Propagating Waves*

A fundamental feature of developing neural circuits is the presence of spontaneous network activity, taking the form of propagating waves. This spike propagation was observed as a cluster of excitation waves in both simulated and cultured neuronal networks [20]. This phenomenon is called spike wave propagation and describes how the neural activity patterns are organized spatiotemporally as synchronous waves which propagate from one site to neighboring sites in all directions. The aim of this thesis is to study about one such type of spontaneous network behavior, the network bursts.

2.2.2 *Self-Organized Criticality*

The propagation of network activity is synchronous in nature that obey a power-law relationship. Self-organized criticality (SOC) describes that large interactive systems naturally evolve toward a critical state in which any perturbation is capable of triggering cascades of events (or *avalanches*)

through the system, and these events can be well-described by power laws (small events happen more frequently than large events). It is a common phenomenon observed in certain complex systems with multiple interacting components, for example, neural networks, forest fires, and earthquakes [18]. A fundamental goal of this thesis is to study the trigger patterns behind these network activities.

2.2.3 *Bursting Behaviors in Neuronal networks*

The most striking behavior of network dynamics in cultured cortical networks is *network bursting* and results indicate that it could play an important role in information processing [70]. A network burst or population burst is defined as a synchronized spiking event that involves most or all of the neurons throughout the network. This behavior is characterized by long quiet periods, with almost no spike emission, punctuated by brief periods of intense spiking activity, where the whole network displays high firing rates—most neurons emit at least 2 closely-packed spikes. This particular pattern is then repeated, with varying regularity, over long time intervals [12].

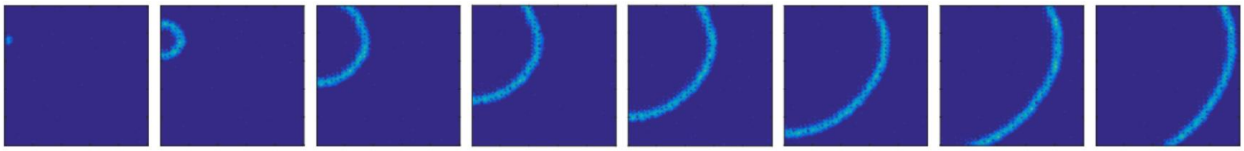

Figure 2.3. Evolution of a single burst. Images showing beginning to the end of a single burst event from left to right [18].

Figure 2.3 shows the evolution of a single burst, how it originates at a location and spreads across the network including most of the neurons in the form of a propagating wave.

Our goal is to find localized patterns that initiate these network bursts. In the next chapter we will study some of the research done previously on generation of these network bursts.

Chapter 3. BACKGROUND: NEURAL DATA ANALYSIS

As previously described, cultured neural networks spontaneously synchronize and generate bursts involving the whole network. These synchronized activities are said to play an important role in our body from development to effective information transmission [13]. Therefore, understanding the patterns that trigger these activities could be of crucial importance. In this chapter we will review some of the research done in the past for the initiation of network bursts. Understanding the mechanism behind network burst activation may shed more light onto what phenomena underlies in the development in the brain, but there is very limited work in this domain.

Previous work by Ham *et al.* [14] was done to analyze ignition sites and spread of spontaneous coordinated activity in cultured networks. It highlighted the presence of major burst leader (MBL) neurons which are defined as a small percentage of neurons (average 17%) that lead a majority of network bursts (84% of all network bursts). This work focused on understanding the characteristics of this leadership and argued that neuronal spike rates are not directly linked to burst leadership. It was found that the shortest path between two MBLs is a single synapse [20]. Since burst propagation and mutual information are both proportional to distance, the work suggested that burst leadership may be dependent on both spiking rate as well as distance of connectivity, i.e. MBLs form a highly connected 'primary circuit' responsible for initiating the majority of all network bursts and maintaining long term spontaneous activity. Another work by Lonardoni *et al* [69] revealed that network bursts are generated in specialized regions of the network called “functional neuronal communities” that appear to be a group of neurons responsible for most of the network bursting activity. Another study by Maeda *et al.* [14] also suggests that generation and propagation of spontaneous synchronous bursts in cultured cortical neurons is governed by the degree of connectivity of the network. This work also found that the source or

initiation point of spontaneous bursts is not at a fixed locus in the network since direction of propagation of spontaneous bursts varies from burst to burst, therefore suggesting that that multiple points in the network have the capability to initiate bursts.

Because of experimental limitations, it has been difficult to determine which network parameters have the greatest impact on observed bursting activity. One approach is to study the effect of the parameters using a computer model that captures many of the key features of *in vitro* networks. This thesis is based on the BrainGrid simulator, a neural simulator based on a leaky integrate-and-fire computational model [19], developed at the University of Washington, Bothell. Details about the simulator configurations will be explained in the Methods chapter. The most recent work done on this data also revealed some behaviors that relate to burst initiation.

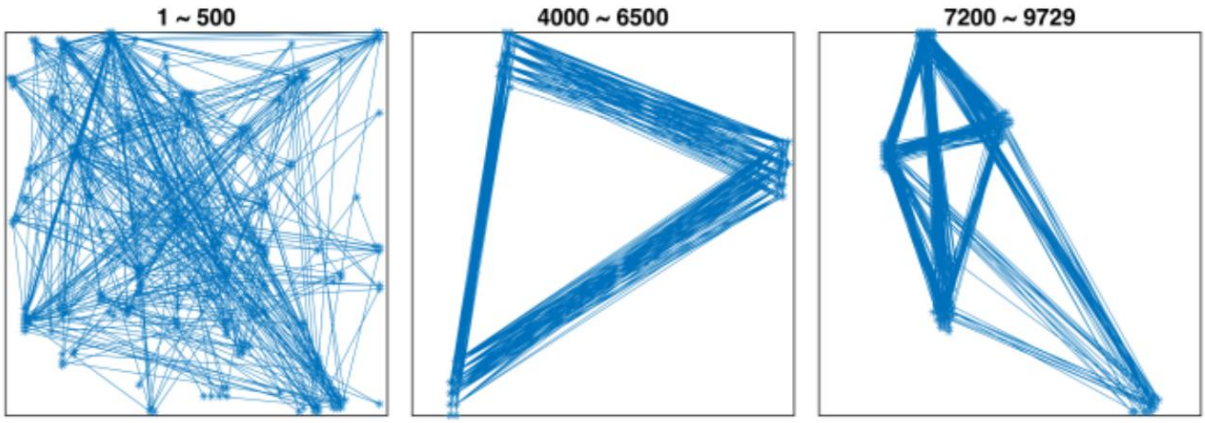

Figure 3.1. Work by Lee *et al* [18] showing burst origin locations in different stages of the BrainGrid simulator development. The left image indicates the first 500 bursts generated while the network is in immature state, the middle image shows the burst origin locations while the network is in development stage, and the last image shows the initiation locations of bursts in the stable state of the simulation [18].

Figure 3.1 shows that burst origins varied widely in early network development and clearly settled down to a small number of origins that occurred repeatedly in an irregular order, or “active origins", and these active origins changed to a different set of locations over time.

In summary, the work done on finding patterns leading to burst initiation are not many but they do reveal some interesting observations, shared among almost all the work, such as, burst leadership being dependent on the high spiking activity of the neurons as well as how closely are they connected to form functional communities, presence of multiple ignition sites and burst origin locations being more stable during the mature state of the simulations. Therefore, the aim of our thesis is to take these findings as a guideline and try to reveal some of these aspects in our BrainGrid Spike train data. This thesis uses data analysis and machine learning techniques to explore some visual interpretability techniques to understand how ensembles of neurons generate these network bursts.

Chapter 4. BACKGROUND: MACHINE LEARNING

The artificial neural networks now prominent in machine learning were, of course, originally inspired by neuroscience [17]. The availability of rich data sets has led researchers from classic hypothesis-driven approaches to innovative data-driven approaches when they can no longer comprehend the complexity and high dimensionality of data by human perception alone [18]. In parallel with the rise of interest in brain networks, there has been an increase in the use and development of machine-learning techniques in many aspects of brain research, for example, brain imaging [20, 21]. However, use of machine learning in identifying burst trigger patterns is quite novel.

Indeed, the high-dimensional nature of spike train data hinders the application of many multivariate methods from classical statistics, prompting an increasing number of researchers to rely on regularization methods common in machine learning and signal processing in addition to well-established mass-univariate analysis techniques [22]. This exponential growth of the amount of biological data available calls for addressing two major challenges, on one hand, efficient information storage and management and, on the other hand, the extraction of useful information from these data [22]. The second problem is one of the main challenges in computational neuroscience, which requires the development of tools and methods capable of transforming all these heterogeneous data into biological knowledge about the underlying mechanism.

In modern neuroscience, neurons are recognized as information processing units that perform complex computations on inputs including history, reflecting biophysical properties, and the activities of neighboring neurons that are synaptically connected [18]. Applying Machine Learning tools and methods would therefore allow us to go beyond a mere description of the data

and provide knowledge in the form of testable models. By inferring our understanding from these models, we may be able to obtain predictions of the system.

In this chapter, we therefore present some literature review of machine learning techniques applied on spike train data, challenges in their analysis as well as introduction of some concepts of deep learning necessary to understand some of the methods implemented in this thesis.

4.1 PREVIOUS WORK USING MACHINE LEARNING FOR SPIKE TRAIN DATA

Since the ability for neuronal networks to process information derives not only from neurons' individual abilities to generate temporal sequences of spikes, but also from their collective dynamics at the network level [38], it is essential to study activities from a larger number of neurons (thousands or more). When the number of observed neurons increases, the number of possible interactions, or patterns, grows rapidly as well. This requires these spatiotemporal patterns be studied by more advanced algorithms.

There is limited literature available in utilizing machine learning for spike train data. Information processing in the brain is carried out by a complex network of neurons communicating by sending reliable stereotypical electrical pulses known as action potentials, or spikes. The information encoded in spike train data is also in a sequence of events over continuous time. Therefore, machine learning techniques used for signal processing are often applied to spike train data [33].

When we look at the machine learning work done for finding patterns related to bursting and initiation, there is only one such research done. The most recent application of machine learning to find bursting initiation activities was performed by Lee *et al.* [18] on the BrainGrid simulation's spike train data. Machine learning classification and regression methods in this work were used to predict the burst timing and location. This work revealed that network bursts can be

predicted from the network activity just before the burst and achieved over 99% accuracy. This thesis analyses some of the results and methods produced by the recent work to form a baseline for comparison of the new methods that will be generated to uncover trigger patterns in network bursts.

From literature review it is found that most researchers who deal with brain data still use traditional statistical methods. While methods such as Support Vector Machines (SVM), Multilayer Perceptron neural networks (MLP), Bayesian methods and many more, have been successfully used so far mainly on brain data, they are not efficient in capturing complex spatio-temporal relationships from spatio-temporal brain data [25]. The enormousness of brain data available and the complexity of the research questions that need answering through integrated models for brain data analysis are grand challenges for the areas of machine learning and information science in general as already pointed out in various publications [34, 35, 36, 37]. Therefore, the next section explains a few introductory concepts of deep learning, a branch of machine learning, although similar in the process of training a model with knowledge from subsets of data, differ in the way they extract features from it.

4.2 DEEP LEARNING INTRODUCTORY CONCEPTS

A promising framework that emerged from the interactions between neuroscience and artificial intelligence is the Artificial Neural Network (ANN) [37, 38, 39, 40]. At their core, ANNs model neural computation using simplified units that loosely mimic the integration and activation properties of real neurons [42]. Artificial neural networks, which simulate brain function, are an attractive and powerful modeling approach widely used in complex feature extraction and pattern recognition [41].

Deep learning comprises simple but nonlinear processing units that each transform the representations or features at one level (starting with the raw input) into a representation at a higher, more representative level [45]. The ‘layers’ are best thought of as being analogous to brain regions, rather than as specific laminae in biological brains [46]. ‘Deep’ learning specifically refers to training hierarchical ANNs in an end-to-end manner, such that plasticity in each layer of the hierarchy contributes to the learning goals [47].

4.2.1 *Introduction to Deep Learning Architecture*

Non-Linear feature extraction:

As we move to much more complex problems, our data becomes extremely high dimensional, and the relationships we want to capture become highly nonlinear. The spike train data can also be considered non-linear in spatio-temporal sense, with undefined patterns.

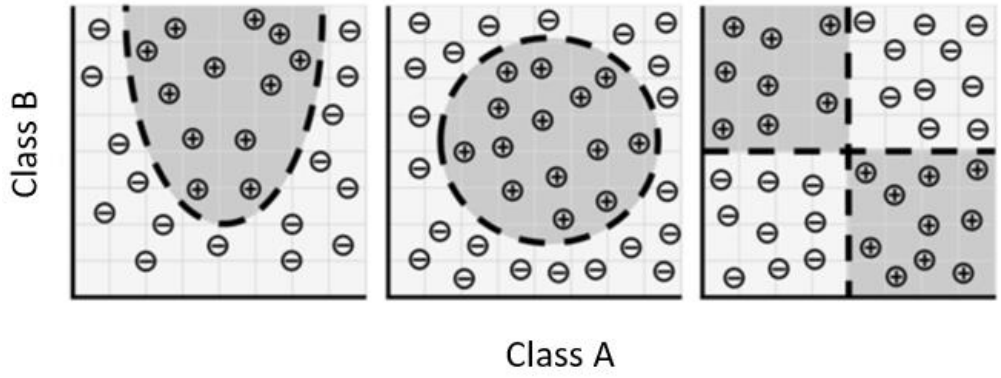

Figure 4.1. This figure shows how complex datasets may have patterns that simple or linear models would find it hard to extract.

As can be seen from Figure 4.1, the type of non-linear patterns in the data shown, can be hard to extract using simple linear models, but are easy to be extracted by non-linear models such as deep learning. These algorithms are therefore a good choice to utilize in finding patterns in high-dimensional and complex, spike train data, specifically in this work, in finding trigger patterns behind network bursts. We will now study in brief how these algorithms function.

Information Processing:

Just like a neuron processes the information it receives and passes it on to the next neuron, artificial deep neural networks take in some number of inputs, $x_1, x_2, \ldots, x_n$, each of which is multiplied by a specific weight, $w_1, w_2, \ldots, w_n$. These weighted inputs are summed together to produce the net

input to the neuron, $z = \theta w_i x_i$. The net input is then passed through a function f to produce the output $y = f(z)$. This output can be transmitted to other neurons [43].

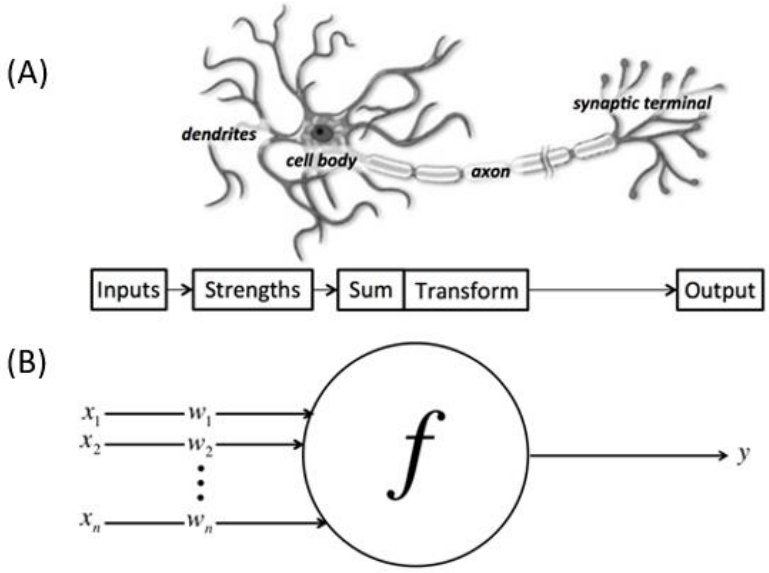

Figure 4.2. This figure shows the similarity in the information processing mechanism of a neuron in a biological neural network vs a neuron in an artificial neural network.

Figure 4.2 (A) shows data processing in a biological neuron vs (B) shows processing in an artificial neuron. In both, the inputs to the neurons are converted into output by applying some transformation. In this research, we will be utilizing deep learning models to identify burst initiation patterns in the image classification setting, the details about which will be defined in the Methods chapter. Therefore, for this thesis, it is important to understand how a model tries to extract features from images and is explained briefly in the next section.

How are images processed?

To explain the function of ANN with an example on a grayscale image, consider the model to be a function $h(x, \theta)$.

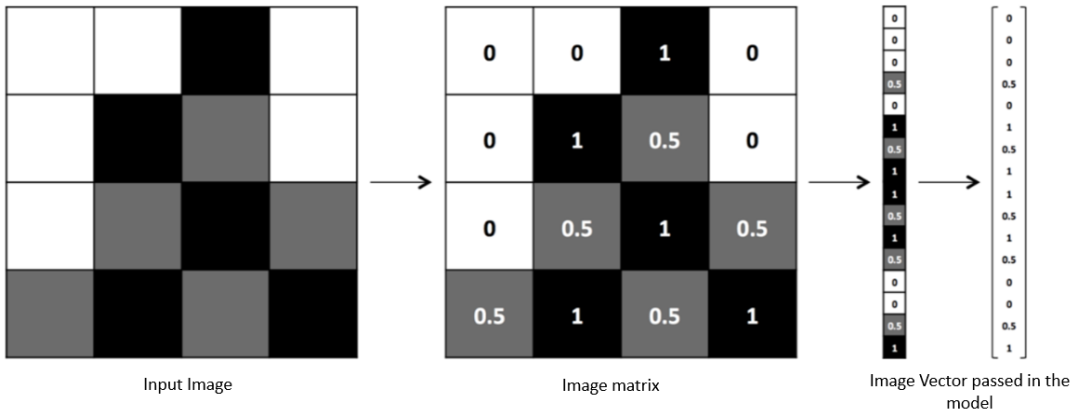

Figure 4.3. The process of vectorizing an image to pass into an artificial neural network for feature extraction.

A grayscale image (x) passed as an input is expressed in a vector form shown in Figure 4.3. The vector θ is the set of parameters that a model uses to process this grayscale image input to extract information. Any machine learning algorithm tries to perfect the values of these parameters as it is exposed to more and more training examples. For our spike train data, we will see in the next chapter, how "x" in the grayscale image will correspond to spiking activity in the network.

Concept of artificial layers in a neural network:

As we have seen how individual neurons process information, a neural network is formed when multiple neurons are connected to each other.

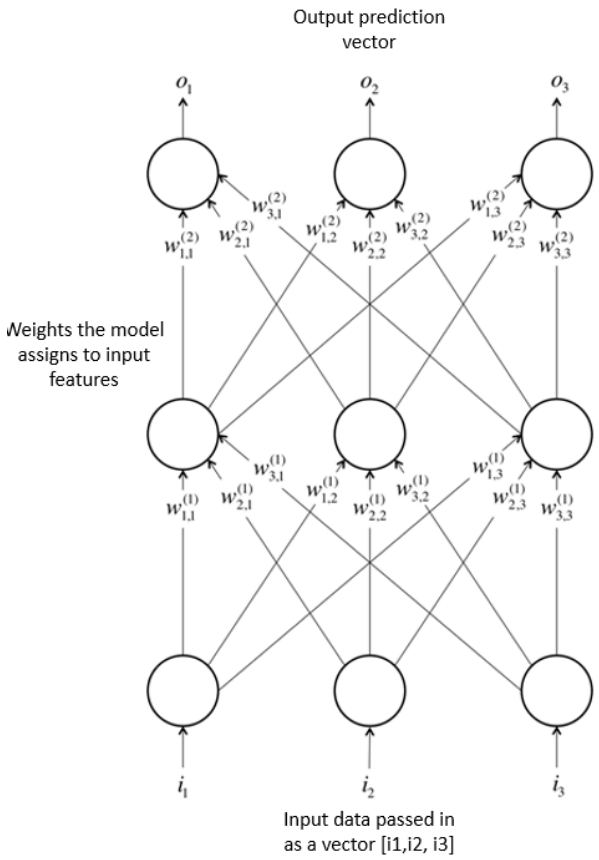

Figure 4.4. This figure explains a simple architecture of neurons in an ANN, how the layers process the input and assign weights as a process of learning to produce the optimum output.

Figure 4.4 demonstrates a simple example of an artificial neural network. The bottom layer of the network pulls in the input data. The top layer of neurons (output nodes) computes our final answer. The middle layer(s) of neurons are called the hidden layers, and we let $Wi,j\ \ (k)$ be the weight of the connection between the i^{th} neuron in the k^{th} layer with the j^{th} neuron in the $k+I^{st}$ layer. These weights constitute our optimizer parameter, θ, and the ability to solve problems with neural networks depends on finding the optimal values to plug into θ [43].

The hidden layers are where most of the magic happens when the neural network tries to solve problems. There are various kinds of hidden layers designed for specific types of complex

calculation and feature extraction. It will be discussed in the Methods chapter, which architecture and hidden layers are chosen for this thesis.

Lastly, it is important to determine three essential components in a neural network: the *objective function, learning rules* and *network architecture*. Objective functions quantify the performance of the network on a task, and learning involves finding synaptic weights that maximize or minimize the objective function. Learning rules provide a recipe for updating the synaptic weights. This can lead to the ascent of the objective, even if the explicit gradient of the objective function is not followed. Architectures specify the arrangement of units in the network as well as the type of layers used to learn representations of data with multiple levels of abstraction. They determine the flow of information, as well as the computations that are or are not possible for the network to learn.

We will see in the Methods chapter, which architecture, hidden layers, and optimization parameters are chosen for this thesis.

Chapter 5. METHODS: DATA ACQUISITION

Computational simulations of networks of cortical cultures help researchers to gain an analytical understanding of the activity in large networks. Although the transition from microscopic to macroscopic scales relies on purely mathematical arguments, simulations are important to add aspects of biological realism that are difficult to treat analytically [34]. More generally, the simulation environments are very useful to the community of theoretical and computational neuroscience where the ideas developed in the toy models could be tested on a larger scale, in a biologically plausible setting, and the ideas arising in different communities and labs are finally connected to the bigger whole. In this thesis we extracted the data for analysis from the BrainGrid simulator. The simulation setting, configuration and the data used is explained in this chapter.

5.1 THE BRAINGRID SIMULATOR

Kawasaki and Stiber developed a large-scale simulation of growth, network formation, and behavior in cultures of dissociated cortical cells [19]. The neuron model incorporates synaptic facilitation/depression and neurite outgrowth/retraction and was used to construct virtual cultures of 10,000 cells whose spiking behavior and evolution were investigated in closed-loop simulations. They developed a GPU-enabled neural simulator for closed-loop, MEA-scale simulations to model the entire network development at the temporal resolution of individual neuron spiking activities with the option of recording the location of each spike to provide spatial resolution. This simulation allows us to perform detailed analysis of the effects of model parameters on burst shape and timing, their changes, their patterns and the inter-relationship among these behaviors, gross network structure, and model parameters [49].

5.1.1 *Leaky Integrate and fire computation model:*

These simulations use a lumped, leaky integrator neuron model that includes synaptic, bias, and noise currents [48].

$$C_m = \frac{dV_m}{dt} = \frac{1}{R_m}(V_{rest} - V_m) + I_{syn} + I_{inj} + I_{noise} \quad (5.1)$$

With V_{rest} being both the asymptotic and reset potential, I_{syn} the total synaptic current, I_{inj} a constant depolarizing current, I_{noise} a noise current, and C_m and V_m the membrane capacitance and resistance, respectively. When V_m exceeds $Thresh$, the firing threshold, a spike event is generated and V_m is set to V_{rest}. This model also incorporates an absolute refractory period, $T_{refract}$. Parameters are set so that most neurons do not produce spikes in the absence of excitatory input; a subset of cells are selected as endogenously active and have their $Thresh$ reduced so that they will spontaneously fire [18].

5.1.2 *Simulation Configuration:*

The cortical culture model is composed of a 100×100 array of neurons, arranged in a rectangular grid as a 10×10 pattern of inhibitory, excitatory, and endogenously active cells that are tiled 10 times horizontally and vertically to form the whole network [49]. Existence and strength, W, of connections between any two neurons is governed by a neurite outgrowth model in which each neuron's set of neurites is modeled as filling a circle centered on that neuron with uniform density. The radius of each neuron's circle of connectivity varies according to [50, 51]:

$$\frac{dR_i}{dt} = \rho\, G(F_i) \quad (5.2)$$

$$G(F_i) = 1 - \frac{2}{1+exp((\epsilon - F_i)/\beta)} \quad (5.3)$$

Here, R_i is the radius of neuron i's circle of connectivity, F_i is neuron i's recent average firing rate (normalized within the range [0, 1]), ρ is a rate constant, is the outgrowth null point, and β controls the sensitivity of outgrowth to F_i. Connections are established between neurons that have circles of connectivity that overlap, with W for them proportional to area of overlap, consistent with the simplifying assumption that neurite density is uniform within such circles and thus number of synapses will be proportional to area of overlap. The network includes excitatory and inhibitory neurons. Most cells were not spontaneously active, but a small fraction have firing thresholds set to produce spontaneous firing at a rate of between 0.02 and 6 spikes/sec. The development rate constant was increased so that the network would recapitulate its full, multi-week development over 60,000s (around 17h) of simulated time [49].

The data was used from the last simulation run on a 2.4GHz Intel Xeon E5-2620v3 system running Ubuntu Linux 16.04.3 using an NVIDIA Tesla K80 GPU with CUDA 8.0 libraries. Every spike produced during the simulation had its time (as an integer time step value with each time step being 0:1 ms,) and (x; y) neuron position recorded. There were 6×10^8 time steps in a simulation and it took about 120 hours to run to completion. The resulting datasets were about 30GB in size, stored in HDF5 data format.

The key information collected from simulation that were used for the analysis in this thesis work are:

- probedNeurons: Unique ID for each neuron (0 ~ 9999)
- xloc: x location for each neuron (0 ~ 99)
- yloc: y location for each neuron (0 ~ 99)
- spikesProbed: Neuron spiking timing and location for every spike

- starterNeuron: List of neurons that are endogenously active (0 ~ 9999)

Chapter 6. METHODS: DATA ANALYSIS

The most recent research using BrainGrid simulator's neural spike data has established preliminary analysis to help us understand the dataset and aspects of bursting behavior such as burst origin location and timing. The main finding of this work was that the spiking activity right before bursts are highly predictive of burst initiation. In this thesis, we dig further in some of those analyses and develop new techniques to find patterns that determine burst initiation. For this purpose, a set of methods were developed to revisit the previous analysis developed for example, burst start times, location and visualize the spiking behavior around these bursts. Many of the results from this step later serve as input for machine learning analysis.

6.1 ANALYSIS OF SPIKE TRAIN DATA

Our cortical culture-simulation model is composed of a 100 × 100 array of neurons. The model simulates over 28 days of development and involves 600 million timesteps, where every timestep is 0.1 ms. Throughout this simulation, we collect spike train data about neuron spiking activity. Some of this data important for our analysis is information about which neuron spiked at which time step, we can identify each type of neuron and their location. We will now discuss in detail the analysis performed on spike train data to fulfil the aim of this work which is to understand the underlying burst initiation patterns.

6.1.1 *Pre-burst and Non-Burst Background Activity*

The central aim of our thesis is to see if there are any patterns in the background spiking activity other than the spikes involved in network bursts, that are predictive of burst initiation. We therefore need to transform the background activity in a form that can be fed into the machine learning models. Previously, experiments performed by Lee *et al.* [18], revealed that these background

activities are highly predictive of the burst location and time. We, therefore, take this work forward to refine and apply different ML models for finding visual patterns in burst initiation.

Taking the concepts defined in the previous research, *pre-bursts* are defined as spike sequences just before bursts, and the spike sequences temporally distant from bursts are *non-burst* precursors. To find the target data sequences to prepare the pre-burst and non-burst data, the start and end times of bursts, or burst boundaries, defined in time steps, are necessary.

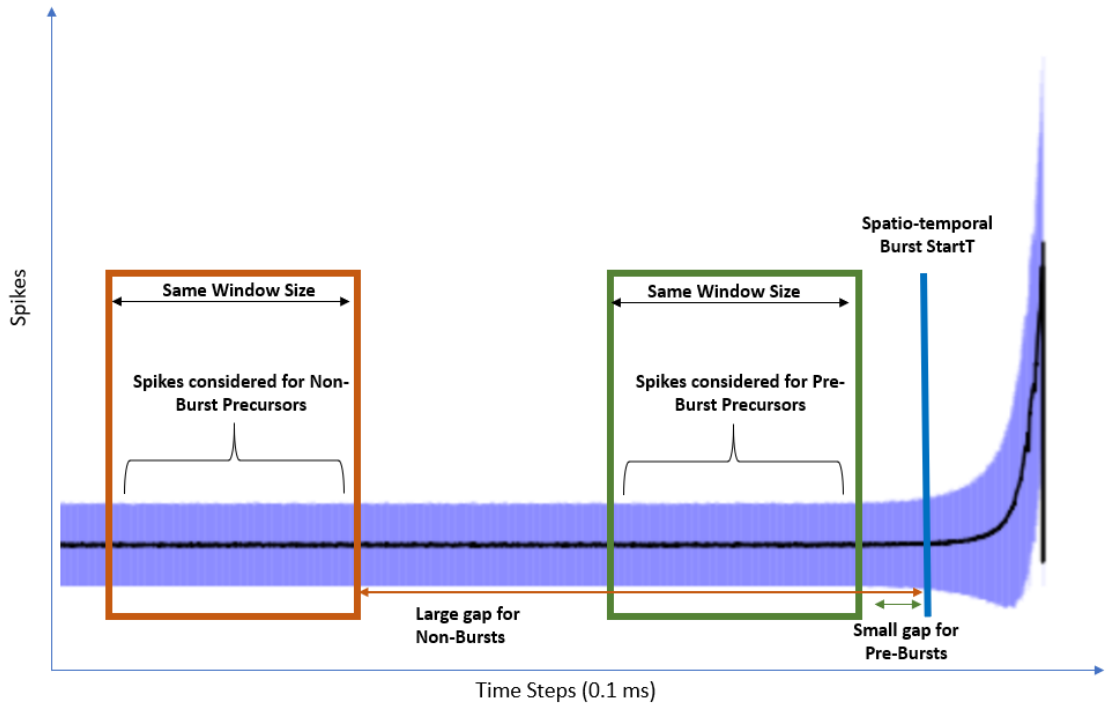

Figure 6.1. Diagram represents the time slices used to define pre-burst and non-burst windows, used for prediction of burst initiation.

As we can see from Figure 6.1, the blue marker refers to the start time of a burst. As mentioned earlier, bursts are a network activity that have a large spiking rate and involve most of the neurons while bursting. Pre-bursts marked with a green window are defined as spike sequences just before bursts and therefore have a small gap. Non-bursts are marked with a red window and are defined as spike sequences temporally distant from bursts and therefore have a large gap. We

generated the pre-burst and the non-burst data based on the burst marker for our deep learning image classification techniques discussed in subsequent sections, as well as for some validation of some machine learning techniques applied in the previous work.

The size of gap and window chosen for the pre-burst and non-burst data is based on synaptic time constant and spike-transmission delay to incorporate synaptic facilitation/depression. *Spike transmission delay* is the time a spike takes to travel from one neuron to another directly connected neuron (neuron that is one synapse away). For the BrainGrid simulation the synaptic transmission delay is 0.8 ms or 8 timesteps [49]. It means that it takes 8 timesteps for a spike to reach a neighboring neuron. This helped us determine the minimum gap before the pre-burst window start time for data extraction. Choosing a pre-burst window too close to the burst start time is not relevant as they do not have enough time to reach the burst initiation location. As any granularity less than 0.8 ms or 8 timesteps would mean that the spikes in pre-burst cannot influence burst initiation, we chose a gap of 10 timesteps before the pre-burst window. We chose a gap of 3000 timesteps for the distance until the non-burst window. The gap for non-burst was chosen based on the inter-burst intervals (IBIs). The smallest IBI found in the data was 5445 timesteps and the mean spike rate. With these information, we assumed a gap of 3000 timesteps gave us sequences that were far enough from the upcoming bursts, eliminating the chances of a precursor window containing any pre-burst activities, and at the same time distant from the preceding bursts to avoid including activities that may pose post-burst behaviors.

For determining the appropriate window size, the *synaptic time constant* was used which determines the duration of memory of a synapse after it is perturbed by a spike before the synapse decays back to resting level. The synaptic time constant of the BrainGrid simulation for 1 synapse is 3ms-6ms or 30 timesteps-60 timesteps. This means that after 2-3 multiples of the time constant,

the synapse effectively forgets the spike. This helps us determine the maximum size of the window to extract pre-burst and non-burst data before the synapse forgets the spike to be able to influence the burst initiation. If the window is larger than 12-18ms or 180 timesteps the burst initiation point would have forgotten the spike and that the first spike cannot have any effects on triggering the burst memory. Therefore, for our experiments, the window size ranges from 50 timesteps to 100 timesteps for both pre-burst and non-burst.

Thus, the two categories, pre-burst and non-burst are our two classes for the classification work discussed in the paper. Our goal is to find if a set of data can be classified in either of the two classes, giving us some insights about burst initiation. Therefore, this makes it very important to make sure the burst start times are correctly identified. However, we found some discrepancies that will be discussed in the subsequent sections.

6.1.2 *Misidentification of Burst Start Time using Temporal Avalanche Method*

We found after analysis that there was some spurious behavior in the burst beginning times. Previous work considered the *temporal avalanche method* to identify burst start time. This method uses the *inter-spike interval* (ISI) in timesteps to identify each consecutive bursts' start times. ISI is defined as the time interval between two consecutive spikes in the spike train. Let N be the total number of spikes in the dataset and t_n be the occurrence time of the n_{th} spike and T_n be the ISI given by:

$$T_n = t_{n+1} - t_n, n = 1,2,..N-1 \tag{6.1}$$

So, the mean ISI, $\bar{T}$, is

$$\bar{T} = \frac{\sum_{n=1}^{N-1} T_n}{N-1} \tag{6.2}$$

A neuronal avalanche is defined as a sequence of spikes or groups of spikes that happen more frequently than the overall mean inter-spike interval (ISI) and all the large avalanche events are then labelled as bursts.

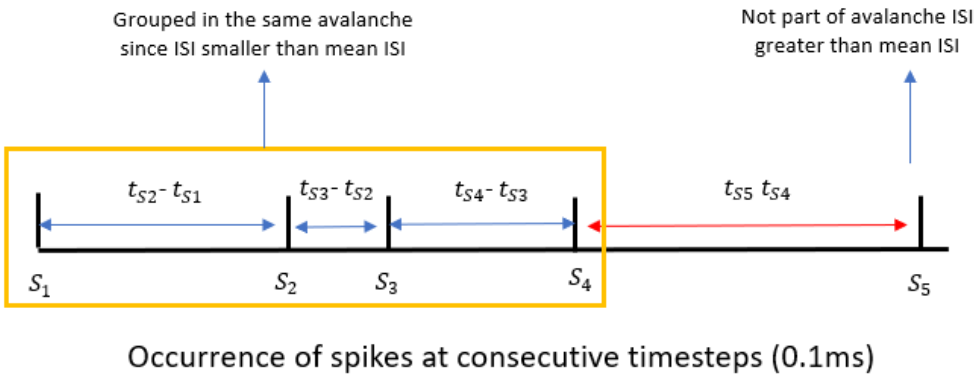

Figure 6.2. Diagram explaining grouping of spikes together into avalanches using the Temporal Avalanche method.

As seen from Figure 6.2, the x axis represents the sequence of spikes occurring at consecutive timesteps. The ISI is calculated for each spike and a mean ISI is then calculated. Any two consecutive spikes are grouped into the same avalanche if their ISI is smaller than the mean ISI. In Figure 6.2, spikes s_1 to s_4 belong to the same avalanche being temporally close, however $s_5$being distant is not grouped into that avalanche. In this way, once all the avalanches are created, all the large avalanches containing with more that 10^4spikes are marked as bursts. In this way the previous method used the Temporal Avalanche produced the burst start time. However, on some investigation, we found that this method of marking burst time is a little too late.

Based on these burst markers it was found that pre-burst activity already included the burst activity. The reason for the high predictability of the machine learning classification models

applied on this pre-burst and non-burst data was that inclusion of burst activity in the pre-burst data made it clearly distinct from the non-burst activity. This made the data biased for machine learning models to classify and therefore is not an accurate interpretation of burst initiation patterns.

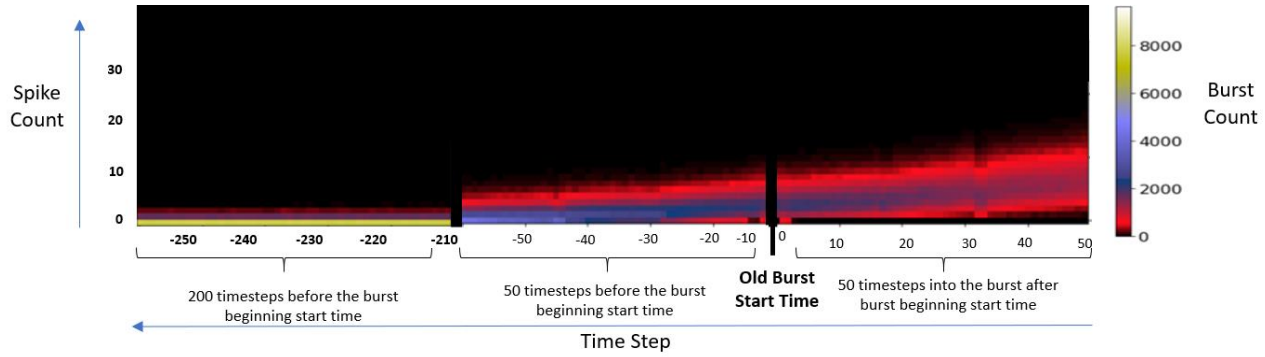

Figure 6.3. Histogram representing the spike count activity relative to the old burst start time marker. The color map represents the number of bursts that have several spikes at a timestep.

As seen from Figure 6.3, we created histograms where the x axis denotes the time steps ranging from 300 timesteps before the marked burst start time and 50 timesteps after the marked burst start time, where the burst start time is denoted with 0. The left axis in the figure indicates spike count or spike activity in every timestep and the right axis denotes color which indicates the number of bursts which show a common behavior of a spike activity at a time step. We see that after the burst start marker, the number of spike activity typically ranges between 0-10 spikes and increases in the later timesteps, as expected for burst activity. But looking closely, we see that the count of spikes starts growing much before the start marker, preceding into the pre-burst activity. This indicates that pre-bursts are not truly the activity preceding bursts as they contain the burst activity already. Therefore, in this thesis we used the spatio-temporal method for identifying true starts of bursts.

6.1.3 *Spatiotemporal Method*

It was found that Temporal avalanches contained spikes that were close in time but distant in space and therefore should be considered as background activity and excluded from the avalanche event. Therefore, we used the Spatiotemporal avalanche method to group spikes into avalanches that were in proximity in both space and time. To do so, we used parameters such as a spatial constraint r, as the radius for a circular spatial window to define a population of interconnected neurons in which their spiking activities directly influence each other as well as a temporal constraint τ based on the number of neurons within the spatial window Πr^2 and calculated τ as the mean ISI for this population:

$$\tau = \top \times \frac{M}{\Pi r^2} \tag{6.3}$$

M was the total number of neurons in the network (10,000) and T the temporal ISI constraint from temporal avalanche method. We modify the mean ISI formula by introducing spatial constraints.

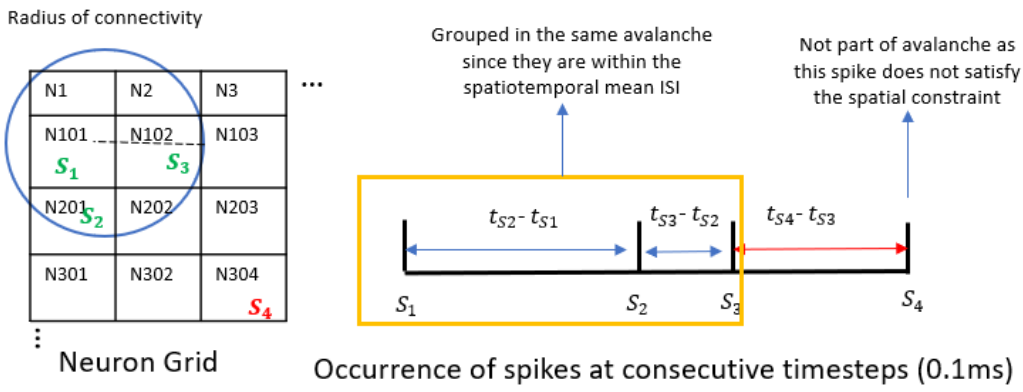

Figure 6.4. Diagram explaining grouping of spikes together into Avalanches using the Spatiotemporal Avalanche method.

As seen from Figure 6.4, the neuron grid refers to the 100×100 grid of neurons in our simulation. The radius of connectivity defines a radius to group neurons within an area. The plot on the right in Figure 6.4 represents the spikes at consecutive timesteps. The grouping of avalanches based on Spatiotemporal method now includes the radius of connectivity, which means when considering a spike to cluster, it is ensured that the neuron producing this spike is within the spatial constraint. In Figure 6.4, spikes s_1 to s_3 belong to the same avalanche being temporally close as well as the neurons producing these spikes are within the radius of connectivity, however the neuron producing $s_4$outside the radius of connectivity is not grouped into that avalanche. As discussed previously, again all the large avalanches containing with more that 10^4 spikes are marked as bursts.

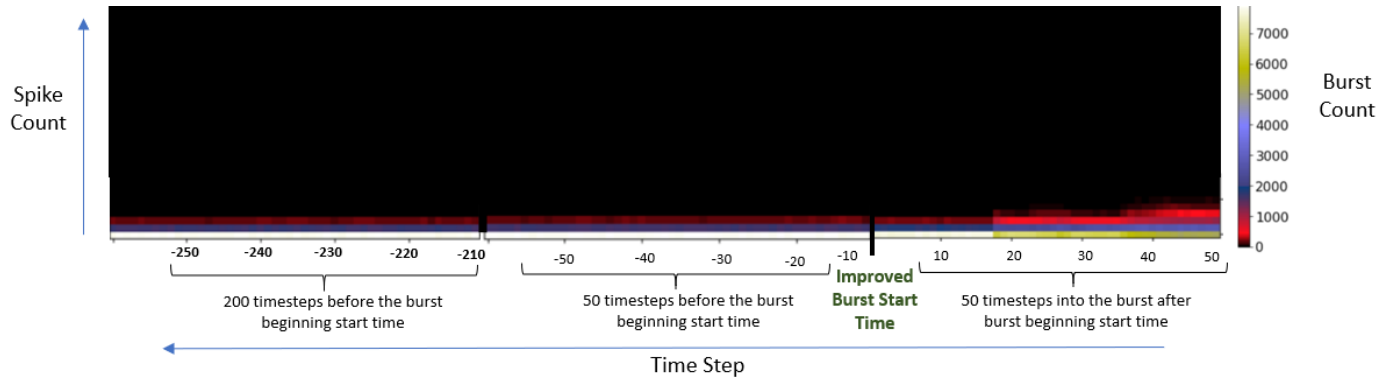

Figure 6.5. This figure is a histogram representing the spike count activity relative to the improved burst start time marker. The color map represents the number of bursts that have several spikes at a timestep.

This method of extracting start time was much better as we can see Figure 6.5 shows that there is no initial growth of spiking activity before the burst, indicating pre-burst activity does not directly contain any burst spiking activity. The new burst start times reduces the chances of the machine learning model being biased with its predictions for identifying whether the pre-burst activities are predictive of bursts or not.

As discussed, one of the goals of this work is to dig into deriving some visual patterns of burst initiation and therefore, we found image classification as a good technique to apply. The next section describes the process used to generate these images.

6.1.4 *Producing Spatiotemporal Images*

We wanted to formulate a way to analyze spiking precursor activities from both spatial and temporal aspects in the form of images. We therefore produced spatio-temporal images of pre-burst and non-burst precursors that can also be used in our machine learning techniques to visually find burst initiation patterns. We used the Spatiotemporal avalanche method discussed in the previous section to get the true start times of bursts. For the images we obtained the burst origin

(x, y) locations and all the spikes (x, y) locations in the pre-burst and non-burst window of every burst.

Temporal aspect:

The window of timesteps used to extract data for pre-burst and non-burst images as explained in Section 6.1.1 ranged between 50-100 timesteps for experimentation. The gap of timesteps before the per-burst window chosen was 10 timesteps and the gap of timesteps before the non-burst window chosen was 3000 timesteps. Therefore, we obtained the spikes for defined number of timesteps for pre-burst and non-burst data and were now fed into the image creation along with the (x, y) location of each burst and spike.

Spatial aspect:

From the temporal aspect, each burst now has a set of spike locations both pre-burst and non-burst. The images were created to denote how the regions of activities differ in the pre-burst and non-burst images, which will be found using the classification methods.

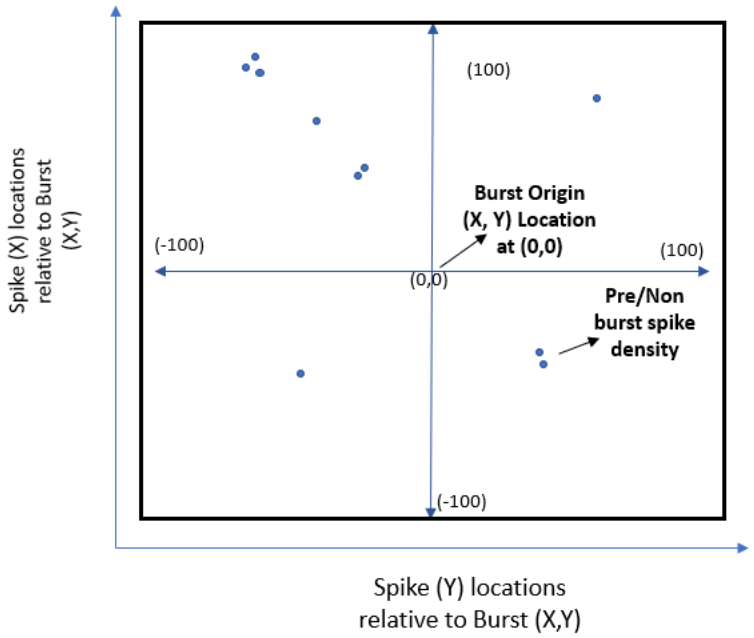

Figure 6.6. Spatio-temporal image description

Figure 6.6 explains the process of generating these images. As we know, our BrainGrid Simulator is composed of a 100×100 array of neurons, therefore the X and Y locations of spiking activities that occur range between 0-99. To understand the activities around bursts and for the sake of stability in the data, our images are centered relative to the burst origin, therefore our X and Y locations of the image can range from -100, 100, producing spatiotemporal images of size 200×200, regardless of the burst origin as can be seen from Figure 6.6. The spikes in each image represent pre-burst activity or non-burst activity relative to the burst origin. The intensity of these images is proportional to the number of spikes produced by that neuron at a location. Therefore, this process creates grayscale images taking in consideration both time and the intensity of activity based on spatial locations.

Bursts settled down to a few stable burst initiation locations at later stages of network development. To trust a prediction by a machine learning model, it is important that the data we

feed in is ground truth stable data. For this reason, we chose the last 25% of the bursts for data preparation and prediction as they are considered more stable than the rest.

When we created these images for both the methodologies of finding burst times-temporal and spatiotemporal avalanche methods, we found visual patterns that pointed to the burst markers being late in the temporal method, shown in the next section.

6.1.5 *Spatiotemporal Image Findings*

Previous work argued that pre-burst activities were highly predictive of bursts, but we found that the old method of producing burst start times were delayed times. To see this visually, we produced our spatiotemporal images on the previous temporal avalanche method of getting burst timings.

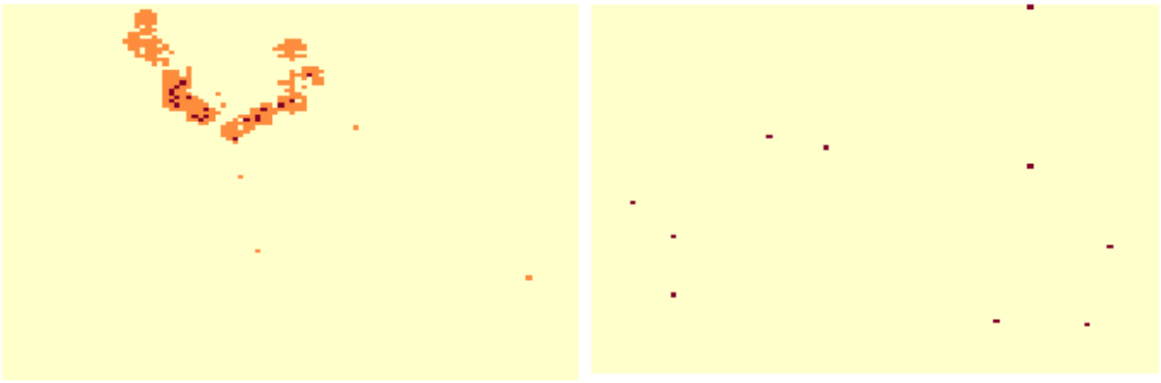

Figure 6.7. Image on the left shows the spatio-temporal image of 1 burst showing pre-burst activity produced by the old Temporal method of marking burst start time and the image on the right shows the spatio-temporal image of a pre-burst determined using the improved method of determining burst start time

As we can see in the images in Figure 6.7, the curved and circular shape of the spiking activity before the bursts reveal that we are already in the burst. For comparison, when we produced the same images for our spatiotemporal method of extracting burst times as well and as we can see, these images are less likely to be biased for our analysis.

6.1.6 *Understanding Endogenous Neural Patterns*

The BrainGrid simulation has two types of neurons, the endogenously active neurons, and the non-endogenously active neurons. The nature of endogenously active neurons is to randomly spike throughout the duration of the simulation in contrast with the non-endogenously active neurons that do not randomly spike. If we remove all the endogenously active, the network still produces the same bursts, but the only difference is that the number of spikes reduces and the background activity is minimal as most of the background activity is made up of these endogenously active neurons. After removing the endogenously active neurons we are left with 430128718 spikes as compared to 570189562 spikes. We also noticed that the number of non-burst large spiking activities, or, non-burst avalanches were significantly reduced as most of the background activity was reduced. Only ~100 non-burst avalanches were left, and they are trivially small. This behavior was very interesting and therefore, we fed in the spatiotemporal images produced after removing the endogenously active neurons for our machine learning analysis as well. We considered these images as denoised versions of images that could reveal clearer patterns of burst initiation.

Therefore, we now have our data prepared to be passed in the Machine learning analysis with spatiotemporal images. We perform the upcoming analysis on three slices of data, images of the data produced by the old method or temporal method of burst timing identification, images of all the bursts extracted from the new method or spatiotemporal method and lastly the images produced after removing endogenously active neurons.

Chapter 7. METHODS: MACHINE LEARNING AND DEEP LEARNING

In the previous chapter, we conducted a number of investigations of macroscopic, whole-network behaviors on our dataset; we found out that using spatio-temporal methods of data extraction would be more fair and would bring less bias when applying machine learning technique. To serve the goal of this thesis, which is to identify the burst initiation patterns in the high dimensionality, complexity, and volume of our dataset, we use machine learning to enable us to discover hidden patterns.

7.1 COMPARISON OF PREDICTABILITY OF OLD VS IMPROVED METHOD USING MACHINE LEARNING

To understand the differences in the precursor predictability we applied machine learning binary classification techniques on new burst markers produced by the improved spatiotemporal data produced. Binary classification is a supervised learning method of classifying a data into two predefined categories or labels, in our case, “no burst" and “burst". This would give us signals about whether the pre-burst precursors are capable of initiating network bursts. This is a similar analysis done in the previous work and our main goal described in this section is to verify the performance of this Machine Learning analysis on the old method of burst identification and the improved method.

7.1.1 *Data Preparation and Labelling*

As input to the ML model for prediction, the temporal and spatial were combined and passed to the model. Based on the window of selection, we choose W number of spikes before the burst as pre-burst and W number of spikes distant from the burst as non-burst.

In the temporal dimension of the data, we retrieved the firing time of every spike in the window, $\phi i = t_i - t_0$, relative to the first spike (spike 0) in that data sample and for the spatial dimension we retrieved its neuron (x, y) location.

Example W=5, firing times and locations of 5 spike sequences pre-burst/ non-burst

Firing time:	77	23	43	2	12	Burst StartT
(x,y) Locations:	(3,44)	(12,22)	(43,22)	(4,21)	(77,99)	Burst StartT

Figure 7.1. Figure represents the data passed as input for the ML model in the pre-burst and non-burst window.

Figure 7.1 explains the data ingestion process. As we can see, each spike in the pre-burst and non-burst window was passed into the ML model with 3 input attributes—firing time, x and y location.

All the pre-burst precursors were labelled as 1, whereas all the non-burst spike sequences were labelled as 0. We also chose 30% of the data for testing the model's performance.

We performed these experiments on three sets of data:

1. Data generated by the old temporal avalanche method for comparison
2. Data generated for all bursts with new start times generated using spatiotemporal avalanche method
3. Data generated for all bursts with new start times by removing spiking activity generated by endogenously active neurons

7.1.2 *Applied algorithms*

We applied various machine learning classifiers from linear classifiers, to non-linear, to ensemble techniques. The approach of classification is like previous work however we experimented with

some additional ML models. Among various ML classifiers, decision tree (DT) and Polynomial support vector machine (SVM), Random Forest Ensemble Classifier and Multi-Layer Perceptron were chosen to predict burst initiation.

Decision Tree:

Decision Tree (DT) uses a tree-like graph for decision making; it constructs the tree using a top-down approach by considering information gain as a criterion and chooses the best feature to split each node so that it produces the “purest" subsets and stops when data cannot be split further. In other words, a decision tree is built by calculating feature importance. DT is easy to interpret by a non-statistician and is intuitive to follow; it copes with irrelevant features and can combine heterogeneous data types into a single model.

Support Vector Machines:

Support Vector Machines (SVM) finds the maximum margin hyper-plane that best separates two classes in a high-dimensional feature space by plotting each data sample as a point in *n*-dimensional space (where *n* is the number of features in the data sample) with the value of each feature being the value of a particular coordinate. SVMs are very universal learners, they can provide generalized models in the presence of many features if the data is separable with a wide margin.

Random Forest:

Random forest classifiers are ensemble classifiers that form a prediction by taking a majority vote amongst an ensemble of decision trees. Random forests are known for their competitive accuracy

and resilience to high-dimensional data; because of their use of decision trees, they can capture some non-linear interactions of multiple variables without the need to predefine interaction terms; because of their ensemble nature, they are more robust against overfitting than are individual decision trees

Multi-Layer Perceptron:

Multi-Layer perceptrons are the most basic form of an artificial neural network, the only difference being in the learning process. MLPs employ a feed-forward mechanism of processing the input data which is only in one direction. In comparison, the deeper neural networks also have a mechanism called backpropagation, which loops the data back through the network after every pass, to re-update the weights assigned to input features. MLPs are therefore, the simplest form of an ANN, and are said to be one of the superior non-linear classifiers.

7.2 SPATIO-TEMPORAL IMAGE CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORKS

After our first set of experiments using Machine Learning models and setting baseline performances of each data and window slices, we wanted to take a step further into drilling down the non-obvious patterns in the data. We therefore created Spatiotemporal images from the burst data as explained in the previous chapter, to be able to perform image classification, and find visual trigger patterns around burst initiation.

7.2.1 *Data Preparation and Labelling*

As discussed in the previous chapter, the idea behind generating Spatiotemporal images was to capture both spatial and temporal dimensions of the burst data in a single abstraction.

Pre-burst: *Non-Burst:*

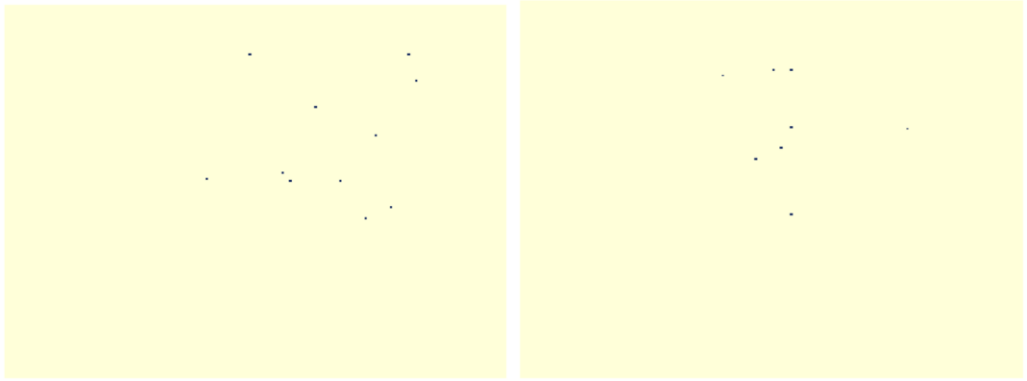

Figure 7.2. Image on left refers to a pre-burst image produced by the spatio-temporal method of image creation, whereas the image on the right is the non-burst image produced from the same burst

Some examples of our image are shown in Figure 7.2. These 200 x 200 images were passed into the deep learning models as vectors and are converted to RGB images just for visualization. All the pre-burst precursor images were labelled as 1, whereas all the non-burst images were labelled as 0. We split the data into 30:70 ratio for test: train respectively for measuring the model performance.

We performed these experiments on three sets of image data:

1. Spatio-temporal images generated on data with old burst-start time by Temporal avalanche method
2. Spatio-temporal images generated for all bursts on data with improved burst start times
3. Spatio-temporal images generated for all bursts with burst new start times and removing spiking activity generated by endogenously active neurons

7.2.2 *Data Shuffling*

One important macroscopic feature of the simulation behavior was that bursts settled down to a few stable burst initiation locations during development and most bursts originated from those

regions at later stages of network development. To trust a prediction by a machine learning model, it is important that the data we feed in is ground truth-stable data. For this reason, we chose the last 25% of the bursts (last 2500 bursts out of 10,000 bursts) for data preparation and prediction.

7.2.3 *Advantages of Convolutional Neural Networks*

Classifiers like logistic regression and linear SVM can be regarded as single-layer classifiers, while SVM with kernels and decision tree are classifiers with two layers [60]. The brain is excellent at tasks like image recognition because of its multilayer structure from retina to cortex [61]. As we studied in Chapter 4, multilayer-based classifiers lead to higher classification accuracies compared with the traditional classifiers with shallow layers in the fields of image, language, and speech recognition [52, 62] . Therefore, apart from applying traditional machine learning techniques, we wanted to use models that can derive deeper insights from our spike train data. For that reason, we even used the multi-layer perceptron mentioned in the previous section. However, it is noted that regular multilayer perceptrons work fine for small images (for example, MNIST or CIFAR-10), however, they break down for larger images because of the huge number of parameters required. For example, for our scenario, a 200 × 200 image has 40,000 pixels, and if the first layer has just 1,000 neurons (which already severely restricts the amount of information transmitted to the next layer), this means 40 million connections; and that is just for the first layer [55]. Therefore, one reason for choosing convolutional neural networks (CNN) is because they can solve this problem using partially connected layers. Because consecutive layers are only partially connected and because they heavily reuse weights, a CNN has far fewer parameters than a fully connected multi-layer perceptron (MLP), which makes it much faster to train, reduces the risk of overfitting, and requires much less training data. CNNs have lots of advantages over regular ANN such as the ability to detect a learnt feature anywhere in an image, much better generalization on images, etc.

In contrast, once a regular MLP has learned to recognize a pattern in one location, it can recognize it only in that location. However, the most important aspect of using CNN is how the CNN architecture embeds this prior knowledge using their feed-forward and back propagation mechanism, studied in Chapter 4. In our application learning such features from the training data and explaining the test data could be very useful in determining burst initiation patterns. Multiple different types of layers in the CNN architecture are used to effectively extract important features from images. The upcoming sections will discuss the basic structure and layers of CNN and define the models and parameters used for our experimentation.

7.2.4 *General CNN Architecture*

A convolutional neural network (CNN) is a very special kind of multi-layer neural network. CNNs are designed to recognize visual patterns directly from images with minimal processing. CNNs, just like regular neural networks, are also made up of neurons with weights that can be learned from data. We have already studied the process of neural networks performing operations on data in Chapter 4, but in brief, each neuron in a neural network is useful for performing operations on a set of data based on the activation function used. Therefore, the entire neural network learns to perform useful computations for recognizing patterns in an image. The neurons are connected in a feed-forward network architecture. The neurons in each layer feed their output forward to the next layer until we get a final output, in our case the classification of burst or non-burst.

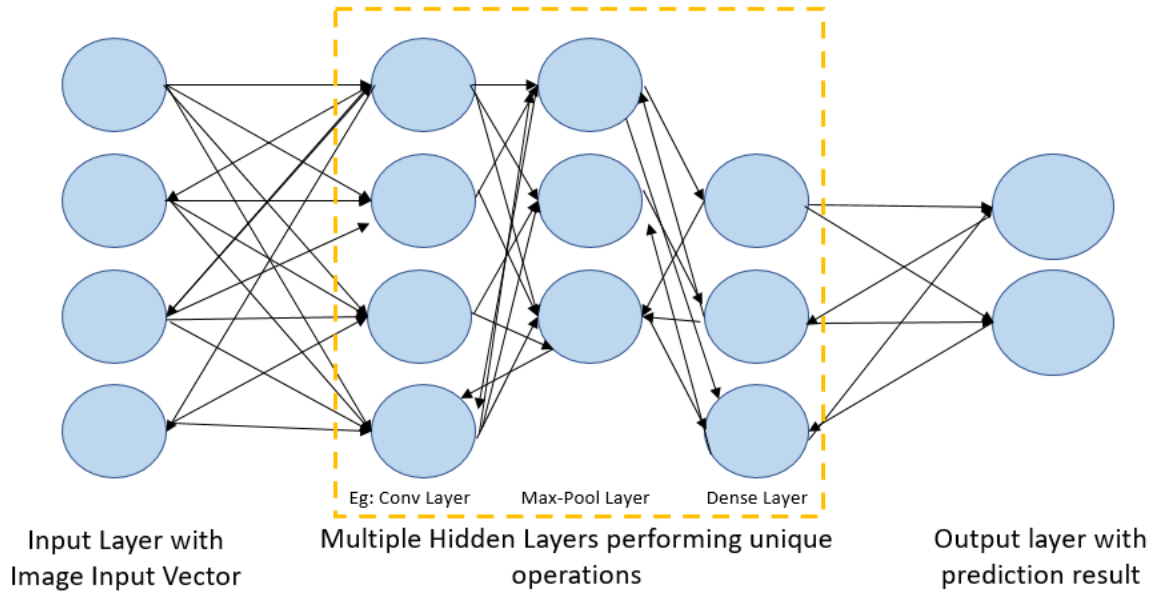

Figure 7.3. This figure shows the general architecture of a Convolutional neural network. It contains an input layer, an output layer, and various hidden layers-which are defined based on the type of problem.

This general neuron connectivity and layers in a CNN can be seen from Figure 7.3. The Figure 7.3 shows how a CNN receives input data from the input layer as a single vector and passes through a series of hidden layers. Every hidden layer consists of a set of neurons, wherein every neuron is fully connected to all the other neurons in the previous layer. Within a single layer, each neuron is completely independent, and they do not share any connections. Every layer has a unique functionality used to perform specific image feature extraction that we will study in the subsequent sections. The last fully connected layer, also called the output layer, contains class scores in the case of an image classification problem [52]. This architecture allows the network to concentrate on low-level features in the first hidden layer, and then assemble them into higher-level features in the next hidden layer, and so on.

The next section explains briefly without going into the mathematical details, about the different layers and how we used it in our experiments. In the upcoming sections we will also understand how each layer of CNN processes information and is able to derive insights from

images that is the main goal of our thesis. These concepts are crucial to understand the different models used for our experiments.

7.2.5 *Functionality of CNN Layers for Image Classification*

Generally, there are four main layers in a simple convolutional neural network. They are the input layer, convolution layer, the pooling layer, and the fully connected layer. Each of these is explained below to understand what aspects of Feature extraction from image do they contribute to.

Input Layer:

The input layer in CNN generally contains image data. Image data is represented by a three-dimensional matrix. Suppose the size of the input is $H_I \times W_I \times C$, here $H_I$represents the image height, $W_I$the image width, and $C$the channels (image color planes). In our experiments, since we passed raw image matrices and not the RGB images shown for visualization, our channel for the first few experiments are 1, and the height and width are 200×200. In the experiments where we implemented Alexnet, we chose 3 channels and converted our images into RGB as required by the pre-trained model, with height and weight as 227×227 (zero padded from original size; see later for more about this).

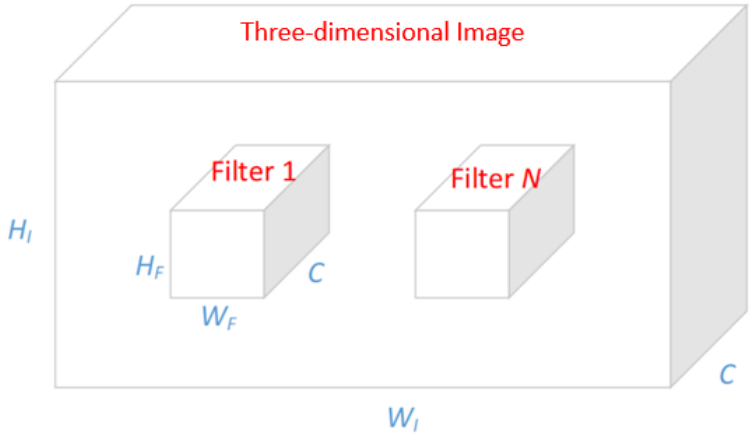

Figure 7.4. This figure depicts how an image is taken as an input into the input layer of CNN. $H_I \times W_I \times C$ are the dimensions of the input image. During data processing a filter of dimension $H_F \times W_F \times C$ is used for image feature extraction.

As shown in Figure 7.4, our image is shown with the bigger cube of dimensions $H_I \times W_I \times C$, and the filters used in convolutional layers, explained in the next section are used to convolve over our images to extract features of dimensions $H_F \times W_F \times C$.

Convolutional Layer:

The main objective of convolution in relation to CNN is to extract features from the input image. This layer does most of the computation in a CNN [55] The convolution layer performs the two-dimensional convolution for three-dimensional input and a three-dimensional filter. An important parameter in the convolutional layer is the filter and the size of the filter is $H_F \times W_F \times C$, here $H_F$and $W_F$represent the height and width of the filters. A filter is convolved with our image to extract features. A feature may be vertical edge or an arch, for example. The feature that the filter helps identify is not engineered manually but derived from the data through the learning algorithm. The region the filter covers at any time during the convolution algorithm is called the receptive

field. A dot product between the receptive field and the filter is performed and the result of the operation is a single integer of the output volume.

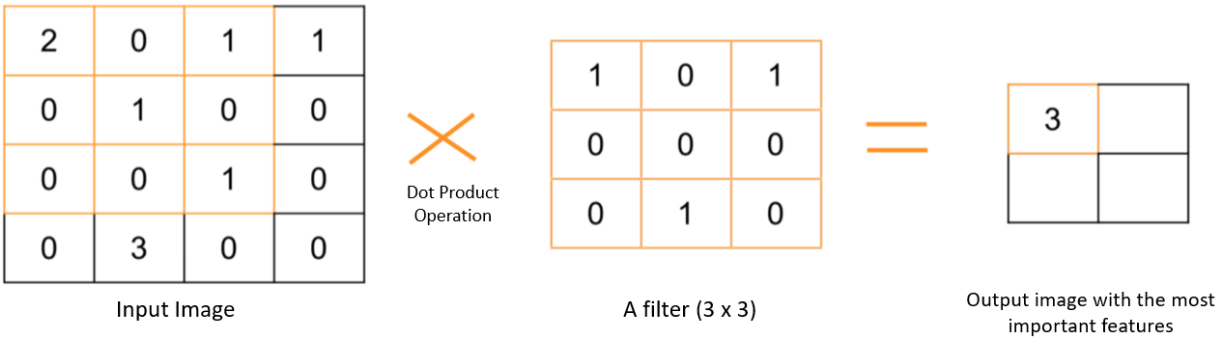

Figure 7.5. The figure shows how a filter operates on every region of the image to extract most useful information

Figure 7.5 shows how a filter of set dimensions is convolved over an input image and performs a dot product between the receptive field and image to extract the output important features. In our spatio-temporal image the pixel values represent the density of spiking activity at a neuron (x, y) location. These spikes can be chosen from the timesteps before the burst or after the burst.

Padding is another important parameter passed in the convolutional layer. If the filter extends outside of the image, then we can either ignore these unknown values (producing an output with dimensions smaller than the input) or replace them with zeros. This is known as padding [55].

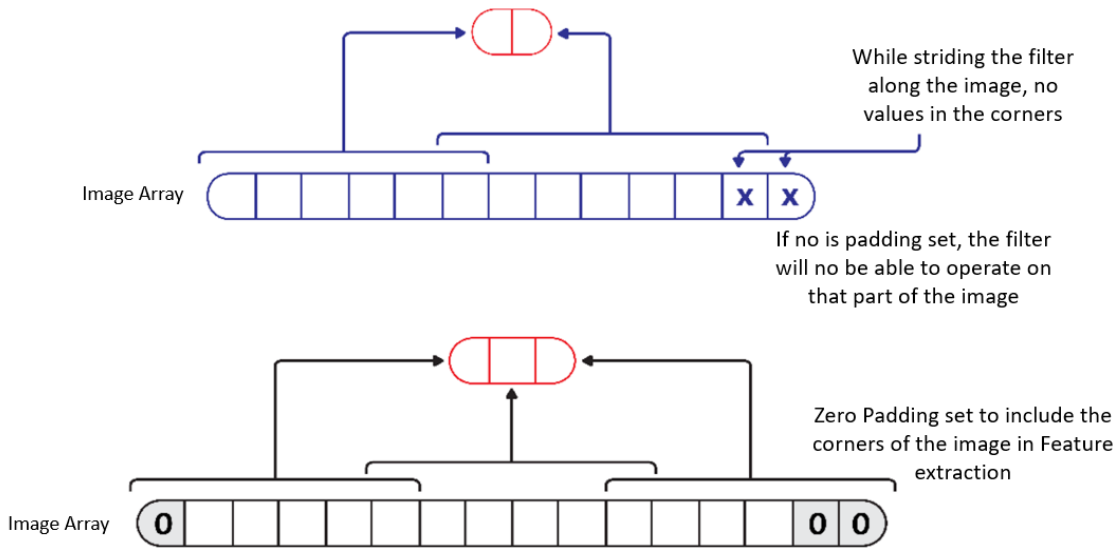

Figure 7.6. This figure shows padding in a CNN helps to avoid losing information while feature extraction.

As shown in Figure 7.6, if padding is not set, we could be ignoring the corners of our image to be used for feature extraction. This parameter is extremely important for our use case, as we do not want to lose any information while extracting the patterns for burst initiation, so we set zero padding in our convolutional layer.

Pooling Layer:

As we have seen, a convolutional layer is a stack of feature maps i.e. every filter stride in an input image in the convolutional layer produces what is called a feature map. More filters increase the dimensionality of convolution. So, the pooling layer controls overfitting by progressively reducing the spatial size of the representation to reduce the number of parameters and computation. The pooling layer is often attached right after the convolutional layer [55]. Due to the large set of experiments that we were performing on burst images from different slices of data, it was important for use to make the convolutional operations as efficient as possible. In other words, the goal of

using pooling is to subsample the input image to reduce the computational load, memory usage, and number of parameters. This helps to avoid overfitting in the training stage. The most used pooling approach is max pooling. In addition to max pooling, pooling units can also perform other functions such as average pooling [55].

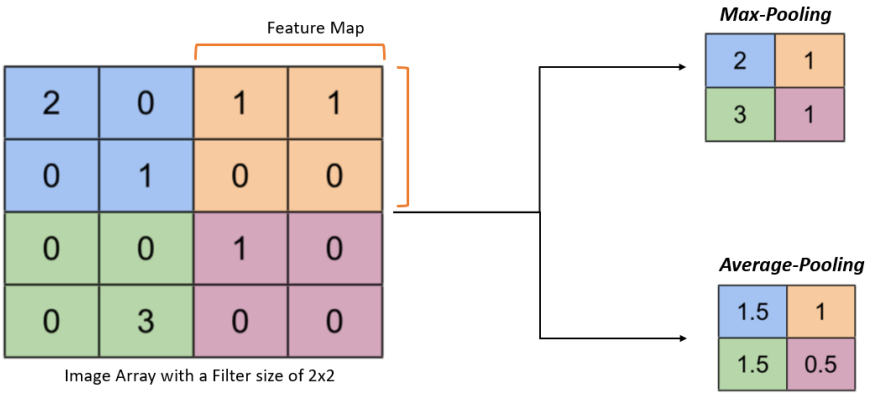

Figure 7.7. This figure shows the two types of pooling that help to extract important features from high-dimensional data and therefore decrease data size of the data

As we can see from Figure 7.7 , max-pooling reduces the image size by mapping the size of a given feature map into a single result by taking the maximum value of the elements in the feature map, whereas average-pooling averages the feature map instead of picking the maximum value. For our implementation we used max pooling, being the most widely used layer that seems to work for most image classification applications.

Fully Connected Layer:

Fully connected/dense layers are usually used as the last layers that also generate the output. The neurons in this layer have full connections to all activations in the previous layer. This is an important layer as it is responsible to utilize all the features extracted from the previous convolutional and pooling layers and use them to classify the image into a label [55]. The output of convolution/pooling is flattened into a single vector of values, each representing a probability

that a certain feature belongs to a label. In our use case, if the training image is of a burst, the features learned throughout the network, that are more likely to represent an image as burst than non-burst, should have high probabilities for the label “burst”. Basically, a fully connected layer looks at what high level features most strongly correlate to a particular class and has particular weights so that when we compute the products between the weights and the previous layer, we get the correct probabilities for the different classes.

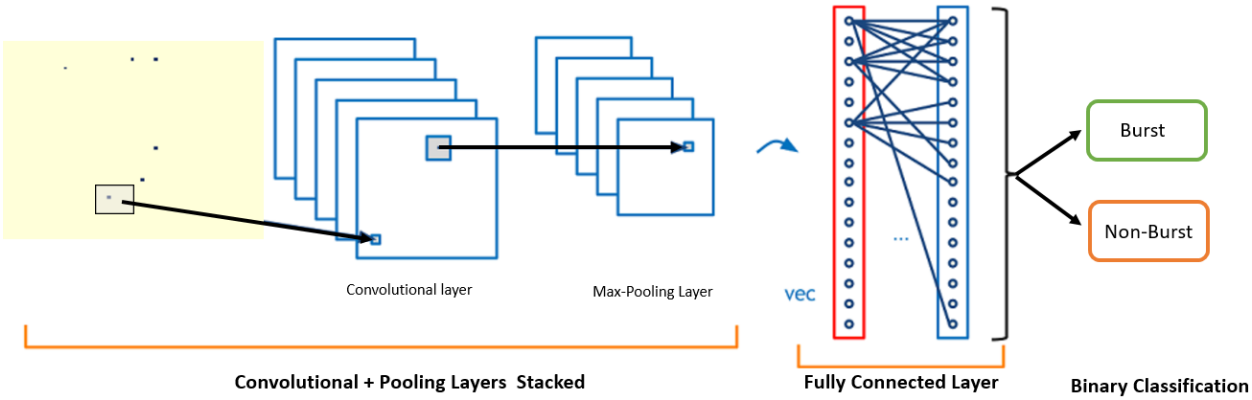

Figure 7.8. This figure shows the placement of the fully connected layer after all the feature extraction layer to transform the numerical insights into output predictions

As can be seen from Figure 7.8, the fully connected layer is the last layer before producing the output. In the next section we will discuss briefly how the training process happens in a convolutional neural network.

7.2.6 *Training process of CNN*

For the classification training process, we need a rich set of training data with the appropriate ground truth labels. As we saw from the previous sections, each layer is responsible for computing a weight for every feature they extract from the image. The way the algorithm can adjust its filter values (or weights) is through a training process called *backpropagation*. A full description of backpropagation is beyond the scope of this thesis; but in a nutshell the process of backpropagation

is divided into four steps. During the forward pass, a training image is taken and passed through the entire network and the weight and the filter values are randomly assigned. In the first pass, the network, with its current weights, isn't able to look for those low-level features or thus isn't able to make any reasonable conclusion about what the classification might be. To make the right prediction a loss function is used. Loss is computed as the difference between the actual label and the predicted label (or some other function of actual and predicted; there are pros and cons for different approaches to this). Therefore, our goal is to minimize the loss function. For this minimization, the model performs a backward pass through the network, to determine which weights contributed most to the loss and finding ways to adjust them so that the loss decreases. We use an optimization function to minimize this loss. This updating of weights and optimization to minimize the loss between the actual and the predicted label is how the model "learns".

7.2.7 *Our experimental setup*

The goal of this thesis is to get a greater understanding about the burst initiation patterns. For this purpose, we produced spatiotemporal images that can be fed into an image classification model so that we can visually see the extracted features that contribute to being in a class. As learnt in the previous sections, CNNs are very powerful in deriving insights from images, however, to really see which compilation and number of layers work for our data, we needed to do multiple experiments.

Model training parameters:

Our initial experiments include a lot of trial and errors using grid search techniques to find the best parameters, i.e. the loss function, the optimizer, activation functions and the learning rate for our

CNN architectures. We found that Stochastic Gradient Descent as an optimizer worked better for us, when compared to the other widely used counterpart adaptive methods (ADAM optimizer). To justify the reason for this, it is found that the solutions found by ADAM generalize worse than SGD, even when these solutions have better training performance [63]. In our scenario, the nature of our data is sparse, and the spiking activity spread out across the neurons, therefore choosing an optimizer which has a better generalization capability would be a better choice for our type of images. Our choice for loss function was Sparse Categorical Cross- Entropy, as it is suggested that this loss function is useful when our classes are mutually exclusive (e.g. when each sample belongs exactly to one class) which matches our use case. The rest of the parameters were kept at default: ReLu activation function and default learning rate due to popularity. Our next set of experiments involved choosing how deep we choose our models to be. Therefore, we included three levels of CNN architecture complexity for experimentation, explained in the next section.

Low Complexity Model:

For many problems, it is found that we can just begin with a single hidden layer to get reasonable results [55]. This architecture ensures that the low-level features are extracted by convolutional and max pooling layers and converted to high level predictions using the fully connected layer.

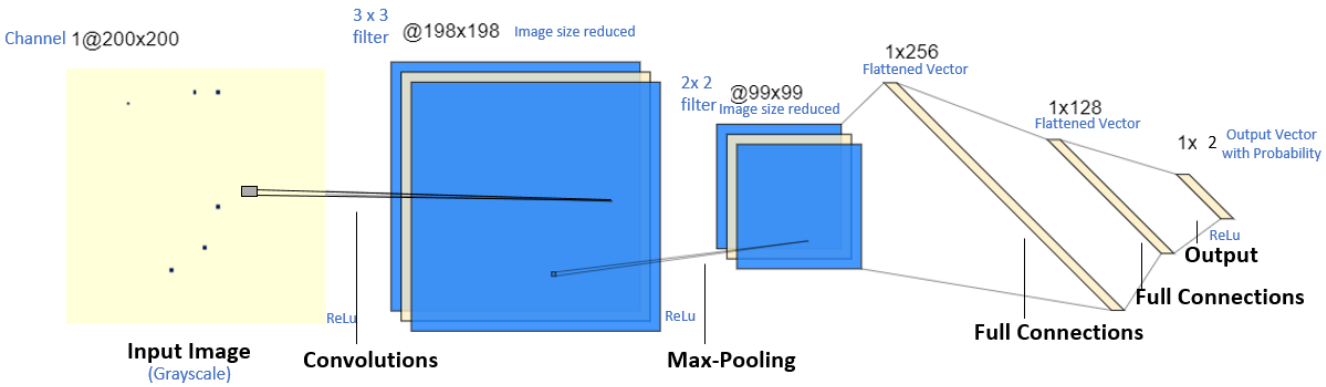

Figure 7.9. Architecture of the low-complexity CNN model used for our experiments

The architecture of the low complexity model is shown in Figure 7.9. In this CNN architecture, the image input layer takes in our spatiotemporal images of the same $200 \times 200 \times 1$, 1 denoting the channel for grayscale. The proposed low-complexity system uses one convolutional (CONV) layer followed by one max-pooling layer (POOL) and Rectified Linear Unit (RELU) as the activation function. For the first convolutional layer, filters of size 3×3 are used. This convolutional layer generates a feature map. The feature map of the first convolutional layer is used in combination with the pooling layer of filter size 2x2 and stride of 2×2. This generates the trainable feature maps, i.e., feature extraction phenomena are performed in these layers. These feature maps are subjected to fully connected layers (FC) and then ReLu activation is performed to determine the classification probabilities used by the final output classification layer for two classes- burst and non-burst.

Medium Complexity Model:

The medium complexity model was created to test the impact of deeper models, with multiple convolutional layers so that they are able to pick up more low level patterns from the images that could help us understand the burst initiation patterns.

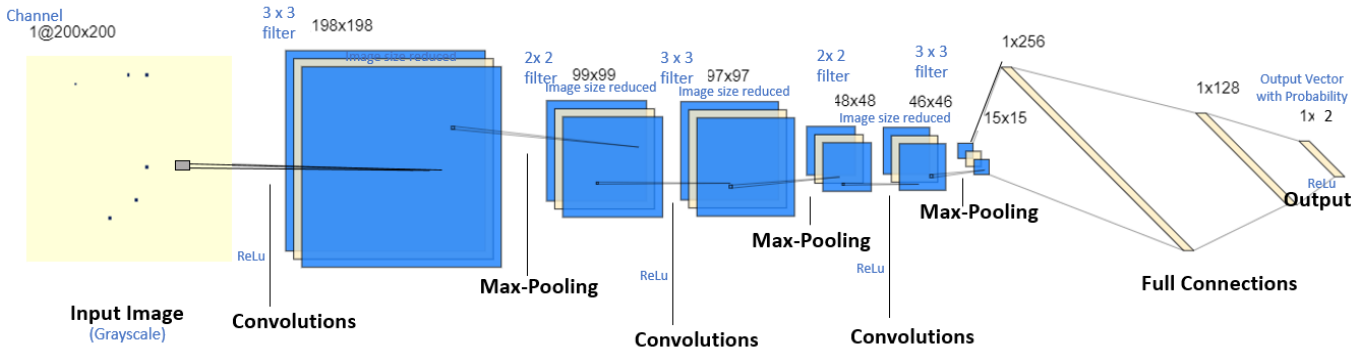

Figure 7.10. Architecture of the medium-complexity CNN model used for our experiments

In CNN architecture shown in Figure 7.10, the image input layer takes in our spatiotemporal images of the same $200 \times 200 \times 1$, 1 denoting the channel for grayscale. Since from our initial experiments we found the training parameters, we kept the optimizers and loss the same, only increased the layers to capture more feature extraction. The proposed medium-complexity system uses three convolutional (CONV) layers followed by three pooling layers (POOL) and Rectified Linear Unit (RELU). For the first convolutional layer, 32 kernels/filters of size 3×3 are used. For the second convolutional layer, 28 kernels of size 3×3 are used. For the third layer, 30 kernels of size 3×3 are used. Each convolutional layer generates a feature map. The feature maps of the first, second and third convolutional layer are used in combination with pooling layers of 2×2 and stride of 2×2. This generates the trainable feature maps, i.e., feature extraction

phenomena are performed in these layers. These feature maps are subjected to fully connected layers (FC) and then ReLu activation is performed to determine the classification probabilities used by the final output classification layer for two classes, burst and non-burst.

High Complexity Model- AlexNet:

It is to be noted that a relatively simple architecture might not always be ideal. Even though, for many problems, we can start with just one or two hidden layers but for a more complex problem, we can gradually ramp up the number of hidden layers, until we start overfitting the training set [55]. We wanted to try a more complex and better tuned architecture for our experiments. There are lots of pre-trained models that seem to outperform the rest of the machine learning models. Instead of building a CNN architecture and training it from scratch, it is also possible to take an existing pre-trained network and fine tune it to adapt it to a new and different dataset through a technique called *transfer learning* [55].

The primary role of transfer learning is to recycle knowledge attained in a previous training process from a pre-trained model, to boost the learning procedure in a new complex domain. It provides an appropriate key for speeding up the learning procedure in image classification [56]. Therefore, we proposed to use AlexNet, a state-of-the-art deep learning pre-trained model. This model was the first breakthrough in the architecture of CNN in 2012. [55]. It was developed by Alex Krizhevsky, Geoffrey Hinton and Ilya Sutskever who won ImageNet Classification Challenge (ILSVRC) in 2012. They trained their network on 1.2 million high-resolution images into 1000 different classes with 60 million parameters and 650,000 neurons.[57, 58]. We wanted to use this architecture as our high-complexity model to see the effectiveness of highly tuned, deep architecture.

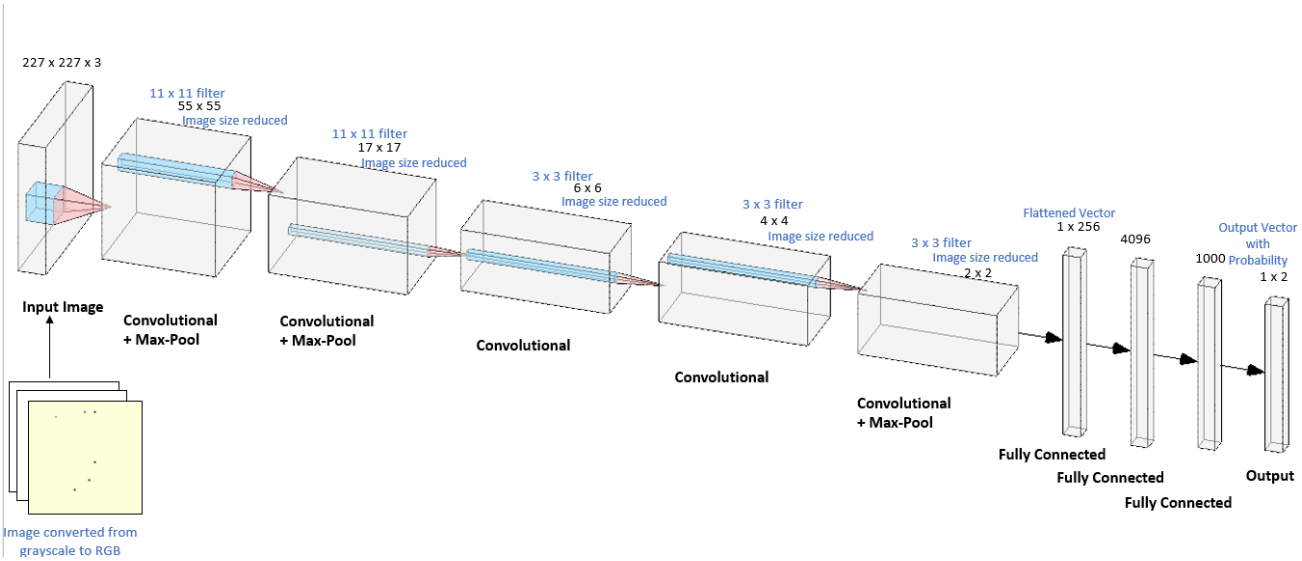

Figure 7.11. Architecture of the high-complexity CNN model, pre-trained model AlexNet used for our experiments

The AlexNet architecture used in our experiments is shown in Figure 7.11. AlexNet expects a $227 \times 227 \times 3$ pixel image, whereas our spatio-temporal images are $200 \times 200 \times 1$ pixels. To feed our images into AlexNet, we resized and padded our image with 0 to fit the dimensions that AlexNet expects, that is, $227 \times 227 \times 3$. In the proposed high-complexity system, the image input layer is defined as a pre-processing layer where the input frames are up-sampled. The network uses five convolutional (CONV) layers followed by three pooling layers (POOL) and Rectified Linear Unit (RELU). The first and second convolutional layer uses filters of size 11×11. The third, fourth, and fifth layers, use filters of size 3×3. The feature maps of the first, second and fifth convolutional layer are used in combination with pooling layers of 3×3 and stride of 2×2.. The framework comprises eight layered architecture with 4096 nodes. This generates the trainable feature maps, i.e., feature extraction phenomena are performed in these layers. These feature maps are subjected to fully connected layers (FC) and then Softmax activation is performed. The original AlexNet architecture used 1000 categories, but to tune it for our domain, we tweaked the last dense

layer to determine the classification probabilities used by the final output classification layer for two classes─burst and non-burst.

7.3 EXTRACTING VISUAL BURST INITIATION PATTERNS FROM MODEL INTERPRETABILITY ALGORITHMS

Model Interpretability is the degree to which a human can understand the cause of a decision. The goal of using a model interpretability technique on our image classification so that we can visually reveal the underlying patterns present in the spatiotemporal images, based on which a model determines whether an image is a burst image or non-burst image. Our goal to this approach was to understand which patterns present in the spiking activity of neuronal cultures initiate bursts. Which means, instead of using a machine learning model for a typical production environment for a software system, model interpretability is the output of interest for us as it has a significant contribution in understanding the visual patterns in a burst image. There are various contenders in libraries that help make machine learning/deep learning models interpretable. A few examples are DeepLift, SHAP, LIME, InterpretML etc.

In this thesis we use a method called DeepSHAP model interpretability. When training on a particular label, some of the features learnt by a model during training are the features of an image that relate the most to a particular label, in our case, which features make an image a burst-image, vs. a non-burst image. The Shapely values (colors in explainer images produced by DeepSHAP) “decompose” the final prediction of an image into an interpretable image showing the contribution of each feature extracted by the model [59, 64]. It is to be noted that not every image sent for prediction to the model has all the features that are ideal to be in a particular class, therefore shapely visualizes the “left out” features as well as present features belonging to each class to arrive at a decision for a predicted label. So SHAP chooses a set of background training

images to explain what the model additively learnt from all the training samples provided for an image to be in a category.

In this work, due to the nature of our sparse image, more than identifying which regions in one image enable that image to be a burst image or a non-burst image, what is found to be a better conclusion is, which collective behavior or set of features in all the burst images could be predictive of burst initiation. We used SHAP for a more global interpretation by calculating the Shapely values for a large amount of training data and aggregating them. Therefore, the explanation by SHAP involving background training data to show the highlighted features for both the classes revealed some interesting properties of an image being a burst vs not a burst that we will see in the Results chapter.

One limitation of using SHAP is the tradeoff between being able to identify the highest contributing regions from a high number of training samples vs computation time [65]. Therefore, without the code breaking we could choose a maximum of 800 training images to explain an output while 2000 training images were passed into the model. We still found some interesting insights that we will discuss further in the Results chapter.

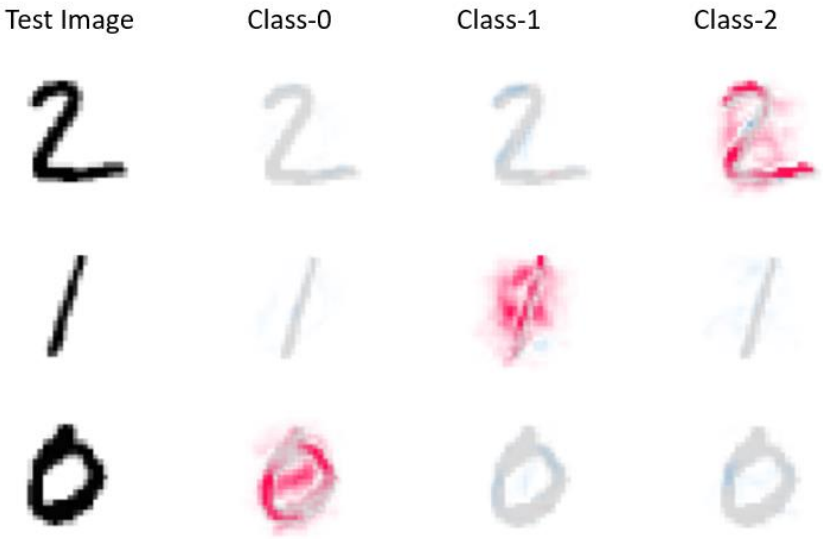

Figure 7.12. Example of MNIST to show how red and blue SHAP values are able to derive collective feature contributors for each category in a classification example

Figure 7.12 explains model interpretability insights generated by SHAP in the context of MNIST dataset. Here the leftmost images represent test images sent to SHAPLift. We provide SHAP with a set of background images that the model was trained on, to explain which regions of our test image have the regions that are more likely to represent a particular class based on all the background training images for that class. By integrating over many background samples DeepExplainer functions in SHAP estimates approximate SHAP values/colors such that they sum up to the difference between the expected model output on the passed background samples and the current model output $f(x) - E[f(x)]$[66]. The red areas increase the probability of that class, whereas blue areas decrease the probability. Therefore, if most of our test has the areas coinciding with the areas that are the most contributing to that particular class from a set of background images, we can explain why a particular test image is labelled as that class. Most importantly, we get information about which regions in the set of images for a class are the most contributive for classifying the image into either of the categories, which is an important insight in our use case.

7.4 METRICS USED TO MEASURE PERFORMANCE

The evaluation of the trained networks for each dataset was carried out using the test set. Together with the global accuracy (i.e. the overall percentage of correct classified images), we included three additional objective evaluation metrics to better understand how well the classifier is performing. We explain each of these metrics in this section.

7.4.1 *Confusion Matrix*

A confusion matrix is used as performance measurement in classification problems of two or more classes. It provides insights into the model by showing numbers such as True Positives, True Negatives, False Positives and False Negatives.

- True Positives (TP): True positives are the cases when the actual class of the data point was 1(True) and the predicted is also 1(True).
- True Negatives (TN): True negatives are the cases when the actual class of the data point was 0(False) and the predicted is also 0(False).
- False Positives (FP): False positives are the cases when the actual class of the data point was 0 (False) and the predicted is 1(True). False is because the model has predicted incorrectly and positive because the class predicted was a positive one.
- False Negatives (FN): False negatives are the cases when the actual class of the data point was 1(True) and the predicted is 0(False). False is because the model has predicted incorrectly and negative because the class predicted was a negative one.

7.4.2 *Accuracy*

Accuracy in classification problems is the number of correct predictions made by the model over all kinds of predictions made.

$$Accuracy = \frac{True\ Positives + True\ Negatives}{True\ Positives + False\ Positives + False\ Negatives + True\ Negatives} \quad (7.1)$$

Accuracy is a good measure when the target variable classes in the data are nearly balanced, as in our application.

7.4.3 *Precision*

Precision is a measure that tells us what proportion of images that we predicted as being Burst images are burst images. It is the fraction of positive predictions made by the classifier that are correct. The question that this metric answer is of all images that are labelled as bursts, how many bursts? High precision relates to the low false positive rate.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \quad (7.2)$$

7.4.4 *Recall*

Recall, also known as sensitivity, quantifies how well the model avoids false negatives, i.e. it answers the question, of all the images that truly are truly bursts, how many did we label?

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \quad (7.3)$$

7.4.5 *F1 Score*

F1 is an important score as it contains both the Precision and Recall scores. It is the weighted average of precision and recall.

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{7.4}$$

7.5 DETAILED EXPERIMENTAL WORKFLOW ARCHITECTURE

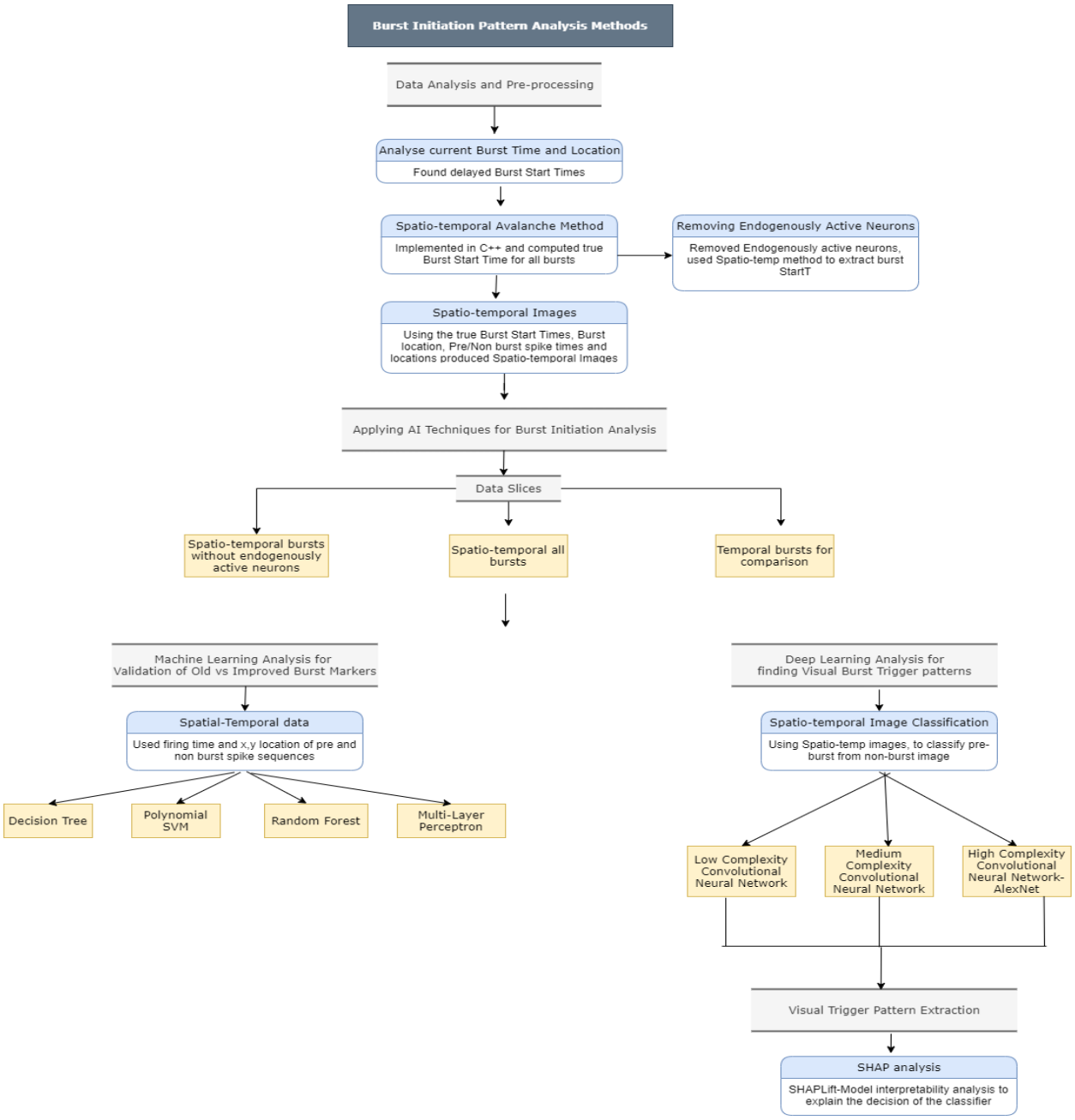

Figure 7.13. This diagram summarizes the entire workflow of data processing, analysis and experiments done in this thesis.

Chapter 8. RESULTS

This chapter discusses all the results from the experiments conducted. The main goal of our thesis is to see the level of interpretability we can achieve by using the Spatio-temporal images created on the slices of data. The data was re-created compared to the previous work using the improved spatiotemporal method of producing burst start time. The three data slices that we have used in these experiments to compare the predictability are all spatio-temporal bursts, spatio-temporal bursts after removing the endogenously active neurons and temporal bursts. Temporal bursts dataset apart for comparison, was also used to test if the predictability is dependent on our model architecture or data. The results were as expected for the Temporal dataset with high performance using the wrong burst markers, as we will see, which also confirm that the model architectures used for our experiments was not the reason behind performance but the dataset.

An important aspect of our work is to reveal visually which collective attributes from a set of training images did the model learn to be predictive of bursts and of non-bursts. The model interpretability algorithm used was SHAP-Lift to find visual trigger patterns. The following sections now explain these results.

8.1 BURST INITIATION MACHINE LEARNING CLASSIFICATION RESULTS FOR COMPARISON

We performed machine learning experiments to find the predictability of bursts vs non bursts from the data generated from the old vs improved method of burst marker generation. We ran these experiments to find how predictive the new spatio-temporal method is as compared to the old method. In the initial runs using raw data using machine learning we found that we can only reach up to 50-52% accuracy when compared to the old method of burst start time marker, suggesting

that our improved data no longer has the obvious behavior in them to be able to easily classify them as burst/non-burst. This behavior is expected and confirms that the old method of marking the burst beginning is already inside the burst therefore highly predictive. These results are shown below.

8.1.1 *Expected ML Results for the old method of marking burst beginning*

Firing time and (x,y) locations were passed for machine learning prediction of burst and non-burst for the burst data produced using the temporal avalanche method.

Table 8.1. Machine learning results with the old method of marking burst times.

ML Results	**Accuracy**		**Precision**		**Recall**		**F1-Score**	
Temporal all bursts	**W-50**	**W-100**	**W-50**	**W-100**	**W-50**	**W-100**	**W-50**	**W-100**
Decision Tree	0.96	0.94	0.96	0.94	0.96	0.94	0.96	0.94
Random Forest	0.94	0.93	0.94	0.94	0.94	0.93	0.94	0.93
Polynomial SVM	1.00	0.99	1.00	0.99	1.00	0.99	1.00	0.99
MLP	0.98	0.97	0.98	0.97	0.98	0.97	0.98	0.97

As seen in Table 8.1 the classification model run on the old temporal avalanches burst data have a very high performance due to the biased data. Another interesting find is that SVM and MLP machine learning methods have the best performance as compared to decision tree and random forest, suggesting that non-linear classifiers are good candidates for classification of spike train data.

8.1.2 *ML results for data generated from the improved burst markers*

For the sake of comparison, the same approach followed for machine learning prediction of burst vs non-burst by previous work was used on the new burst dataset produced using the spatiotemporal avalanche method. Therefore, firing time and (x,y) locations were passed for machine learning prediction of burst and non-burst for the two datasets, all bursts, and all bursts without endogenously active neurons. Table 8.2 highlights the classification results on all bursts dataset and Table 8.3 highlights the classification results on all bursts without endogenously active neurons.

Results of Spatio-temporal All Bursts:

Table 8.2. Machine learning results of spatio-temporal all bursts

ML Results	**Accuracy**		**Precision**		**Recall**		**F1-Score**	
Spatio-temp-all-bursts	**W-50**	**W-100**	**W-50**	**W-100**	**W-50**	**W-100**	**W-50**	**W-100**
Decision Tree	0.52	0.52	0.52	0.52	0.52	0.52	0.52	0.52
Random Forest	0.51	0.52	0.52	0.52	0.51	0.52	0.48	0.49
Polynomial SVM	0.51	0.52	0.51	0.52	0.51	0.52	0.50	0.52
MLP	0.50	0.52	0.51	0.61	0.51	0.51	0.48	0.38

The highest accuracy and f1-score both range between 50-55% suggesting that the images no longer have obvious behavior in them to be able to easily classify them as bursts vs. non bursts. This is a good finding as now we can ensure that the improved method of defining the burst start times are in fact true. As we can see from Table 8.2, when using a larger window size of 100 timesteps, results of random forest and SVM increase by a few percentages for both F1-score and accuracy. Further, this also calls for applying more advanced deep learning techniques as will be discussed in the next section, to finding out the subtle patterns in our data converted as spatio-temporal images.

Results of Spatio-temporal All Bursts generated after removing endogenously active neurons:

Table 8.3. Machine learning results of spatio-temporal all bursts after removing spikes from endogenously active neurons

ML Results	**Accuracy**		**Precision**		**Recall**		**F1-Score**	
Spatio-temp-bursts-no-endo-neurons	**W-50**	**W-100**	**W-50**	**W-100**	**W-50**	**W-100**	**W-50**	**W-100**
Decision Tree	0.65	0.67	0.65	0.67	0.65	0.67	0.65	0.67
Random Forest	0.66	0.66	0.73	0.73	0.65	0.66	0.63	0.63
Polynomial SVM	0.58	0.57	0.58	0.57	0.58	0.57	0.58	0.57
MLP	0.54	0.52	0.54	0.54	0.54	0.52	0.53	0.44

We see that both sets of datasets produced from the spatio-temporal method, with and without endogenously active neurons, are in general less predictive than the old method, indicating that there are definitely no obvious patterns from the bursts in the pre-burst data. It is interesting to note the jump in performance as compared to results in Table 8.2. Both the Decision Tree and Random Forest ensemble classifiers can achieve up to 20% increase in accuracy and F1-scores when the endogenously active neurons are removed. This could indicate that the removal of endogenously active neurons removed a lot of noise from the data, and we found that most of the activity happening in the background, between the bursts are reduced. This could be a reason for the high performance of the ML models. Since we wanted to dig further into deriving some visual patterns of predictability therefore, we applied image classification techniques the results for which are shown in the next section.

8.2 BURST INITIATION DEEP LEARNING SPATIO-TEMPORAL IMAGE CLASSIFICATION RESULTS

We converted the spike-train data into spatio-temporal images to perform image classification in order to see if we can get any improvement in the predictability of burst initiation patterns from using convolutional neural networks, additionally we wanted to use image classification as an input to model interpretability analysis that can reveal some visual patterns in the images.

We found that performing image classification on the old method of producing burst markers received exceptionally high performances reaching 100% because of the visual differences in the image. The performance of image classification on the spatio-temporal all bursts increased 10% in accuracy as compared to the machine learning results and an improvement of 30% accuracy as compared to traditional machine learning algorithms on the bursts data without the background activity generated by endogenously active neurons.

Results of Image Classification on Old method of burst start times:

Table 8.4. Image Classification results on old Temporally generated burst data

Image Class. results	**Accuracy**		**Precision**		**Recall**		**F1-Score**	
Temporal all bursts	**W-50**	**W-100**	**W-50**	**W-100**	**W-50**	**W-100**	**W-50**	**W-100**
Low Complexity CNN	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Medium Complexity CNN	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
AlexNet	0.57	0.71	0.60	0.82	0.57	0.71	0.53	0.68

As expected, all the results seen from Table 8.4 are 100% for the low and the medium complexity CNN models, however it is interesting to note that the high-complexity pre-trained model, AlexNet could achieve only upto 70% F1 score. This reveals that a model tuned for our images performs better than a model that is tuned for all the images in the world. Even though some studies reveal that AlexNet has a good performance for Sparse data, it is not suited for our images. We continue to use this model for comparison for the other slices of data, but we do the interpretability analysis only for the low and the medium complex models.

Results of Image Classification on Spatio-temporal All Bursts:

Table 8.5. Image classification results on Spatio-temporal all bursts without removing any neurons

Image Class. results	**Accuracy**		**Precision**		**Recall**		**F1-Score**	
Spatio-temp-all-bursts	**W-50**	**W-100**	**W-50**	**W-100**	**W-50**	**W-100**	**W-50**	**W-100**
Low Complexity CNN	0.54	0.62	0.54	0.64	0.54	0.62	0.54	0.60
Medium Complexity CNN	0.52	0.49	0.52	0.49	0.52	0.49	0.52	0.49
AlexNet	0.37	0.50	0.37	0.25	0.37	0.50	0.37	0.33

As seen in Table 8.5, comparing the image classification results with the previous ML approach (firing time and location), we see 7% improvement in the F1-score and a 10% increase in accuracy for window 100 results by using the low complexity CNN model for image classification. Getting scores in the range of 50-60% from all the three CNN models do reveal that this data may have a lot of noisy data and may not be able to accurately differentiate the two images. The best performance the CNN models could get on the dataset spatio-temporal all bursts was 62% accuracy score and 62% F1-score. By running the same models on the Temporal datasets, we confirmed that these performances are highly dependent on the kind of dataset we pass into the model and not the model architecture itself. However, with a score of 60% F1-score, we still have slightly better insights as compared to results of "chance" to trust the model's output for burst initiation patterns therefore we will continue to see model interpretability analysis on these models. It is also interesting to note that the performance of W-100 is better than W-50. This result is also consistent

with the other experiments. Also as pointed out in the previous section, AlexNet is unable to perform with suboptimal results due to its highly complex architecture. Some discussion on this is presented in the Conclusion chapter.

Results of Image Classification on Spatio-temporal All Bursts generated after removing endogenously active neurons:

Table 8.6. Image classification results on Spatio-temporal bursts data after removing endogenously active neurons

Image Class. results	**Accuracy**		**Precision**		**Recall**		**F1-Score**	
Spatio-temp-bursts-no-endo-neurons	**W-50**	**W-100**	**W-50**	**W-100**	**W-50**	**W-100**	**W-50**	**W-100**
Low Complexity CNN	0.85	0.96	0.88	0.96	0.85	0.96	0.85	0.96
Medium Complexity CNN	0.82	0.93	0.82	0.93	0.82	0.93	0.82	0.93
AlexNet	0.61	0.58	0.64	0.60	0.61	0.58	0.58	0.57

Image classification after removing endogenously active neurons shown in Table 8.6 turned out to get really good performance as compared to the data with all neurons. Comparing the results of image classification with previous ML results (firing time and location), we got an accuracy of about 70% from the machine learning techniques shown in Table 9.3, however image classification results have crossed the F1-scores and accuracy scores with a 20% increase. We have over 95% F1-score from the low complexity and the medium complexity CNN model indicating, after removal of the endogenously active neurons, the model was able to clearly find out burst initiation

patterns to some extent. Also note that both the low and medium complexity models have a better prediction for W-100 compared to W-50. As discussed before, AlexNet was unable to perform well on this dataset as well. Taking the results from the image classification forward, as the next step we performed the model interpretability analysis revealed in the next section.

8.3 VISUAL BURST TRIGGER PATTERNS

We ran a total of 12 experiments for visual pattern extraction using SHAPlift including the three datasets, the two types of window sizes-50, 100 (number of timesteps before burst/non burst) and the three CNN models. In this section we choose the results which clearly show some aspects of the collective features of a burst image and a non-burst image. In general, the red areas show high probable regions for a class, whereas the blue regions show low probable regions for a class. We present our findings in two sections, one exclusively talking about burst initiation patterns found from burst images and the other section to compare the differences in the collective burst and non-burst patterns the models learnt from the training process.

8.3.1 *Burst Initiation Patterns in Bursts from Burst-labelled images*

Results of Visual Model Interpretability on Spatio-temporal All Bursts Dataset:

From SHAPlift we get insights both about the global and local interpretability of our image data. We will first reveal findings from SHAP on global interpretability, i.e. understanding which collective patterns in the data make an image a burst image or a non-burst image.

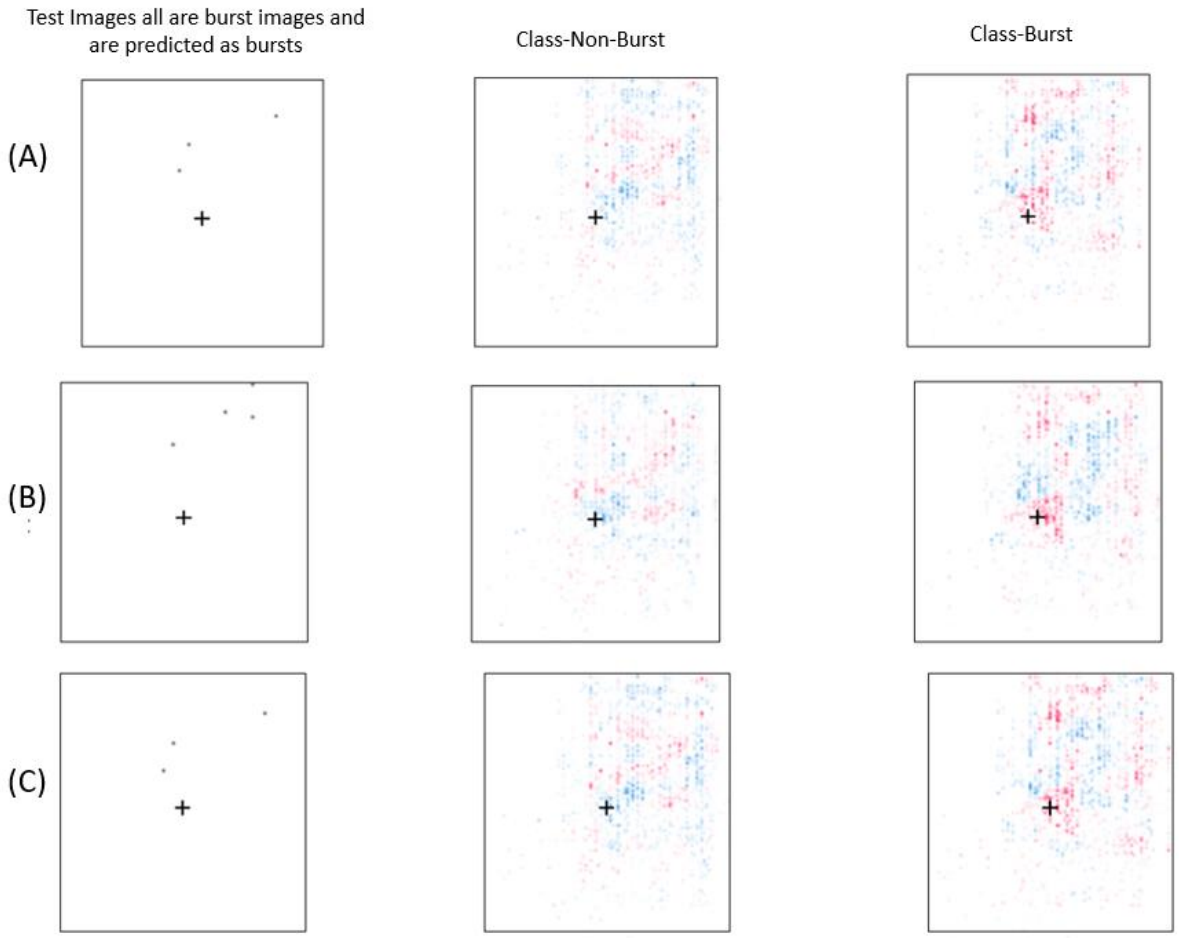

Figure 8.1. Rows (A), (B) and (C) describe the actual test image in the first column vs the interpreted output by SHAP in the middle and the right column, i.e., the first column of images represent the test images that were originally labelled as burst-images and predicted by the label also as burst-images. The middle and the right columns represent which collective patterns from each of the classes the test image matches to more, explaining its prediction. All the explanations are from the medium-complexity model on the dataset- spatio-temporal all bursts with a window of 50 timesteps.

As seen from Figure 8.1, the SHAP colors shown in the middle and the right column, represent which collective features made the model predict it as non-burst vs burst, i.e., the image with a more dense red area, represents our test image to be in that particular class , whereas blue areas decrease the probability. Also note the marker in the center of each image marks the burst origin however they are slightly off-center for visual clarity. When comparing the test image itself with its interpretable output image, note that SHAP images show both expected and actual contributing

features to give a more global interpretation as well. From the images produced for spatio-temporal all bursts, we see that , the CNN model learns to expect a group of regions to be around the center for the image, i.e., around the burst origin related to burst initiation patterns. This finding is interesting as none of the images inputted to the model had an aggregated behavior right at the origin, as our pre-burst image did not have any burst activity included. Additionally, we also see that the red regions, referring to expected regions for burst initiation are not just centered around the burst origin but also spread out throughout the network creating some form of clusters of spiking activities. This can be noted from red regions in the center and across the network in the right column representing burst image expectations in Figure 8.1. In all the three different test burst images A, B and C, the collective contributions learnt from the non-burst images, it is seen that the model does not expect any region in the middle for it to label an image as non-burst. This can be noted from the blue region in non-burst images indicating, the missing of blue regions make that region non-burst.

Due to the sparse nature of our images, it is hard to see how the spiking regions in our test images lead to a prediction, on zooming in, we see how each of the activity regions of our test images are colored red or blue, the more reds in either of the classes indicates the region for the predicted label.

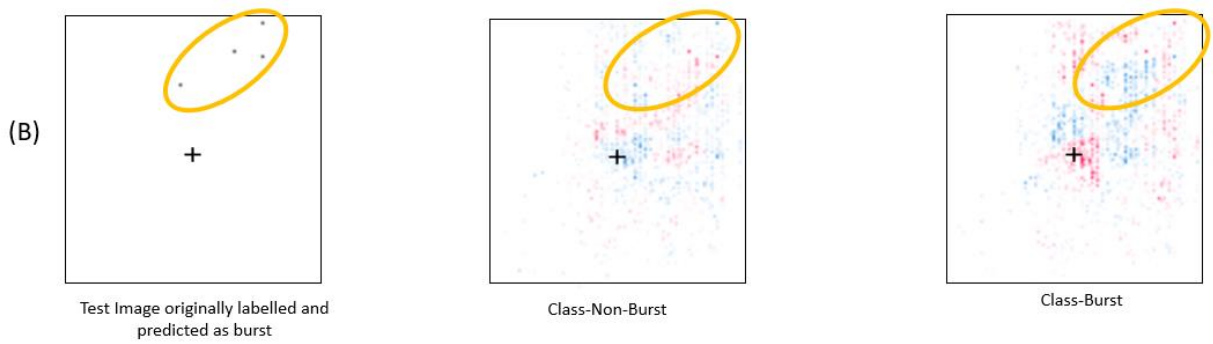

Figure 8.2. Explaining the patterns found from SHAPlift to label this test image as a burst

Figure 8.2 shows local interpretability patterns and explains why the test image B was predicted as a burst. The redder regions found in a class; the test image has a higher probability to be predicted as that class. Our test image has 4 spikes to be colored, From the image above we see that the class on-burst has 2 out of 4 spikes colored red whereas the class-burst has 3 out of 4 spikes colored red, which is slightly more than the non-burst class. For this reason, this image was labelled as a burst. Due to the sparse nature of our images, the major takeaway here is not the individual prediction explanations but the collective behavior of the burst class vs the non-burst class found by the CNN and the SHAPLift models. We see that in the spatio-temporal all burst datasets, the models expect a group of regions to be around the center for an image to be a burst image. We see from Figure 8.1 that this is common for multiple runs of the CNN model on this dataset.

Results of Visual Model Interpretability on Spatio-temporal All Bursts created after removing endogenously active neurons:

We first look at the global interpretable results generated from SHAP on the dataset without endogenously active neurons in Figure 8.3 and then discuss the local interpretable patterns in Figure 8.4.

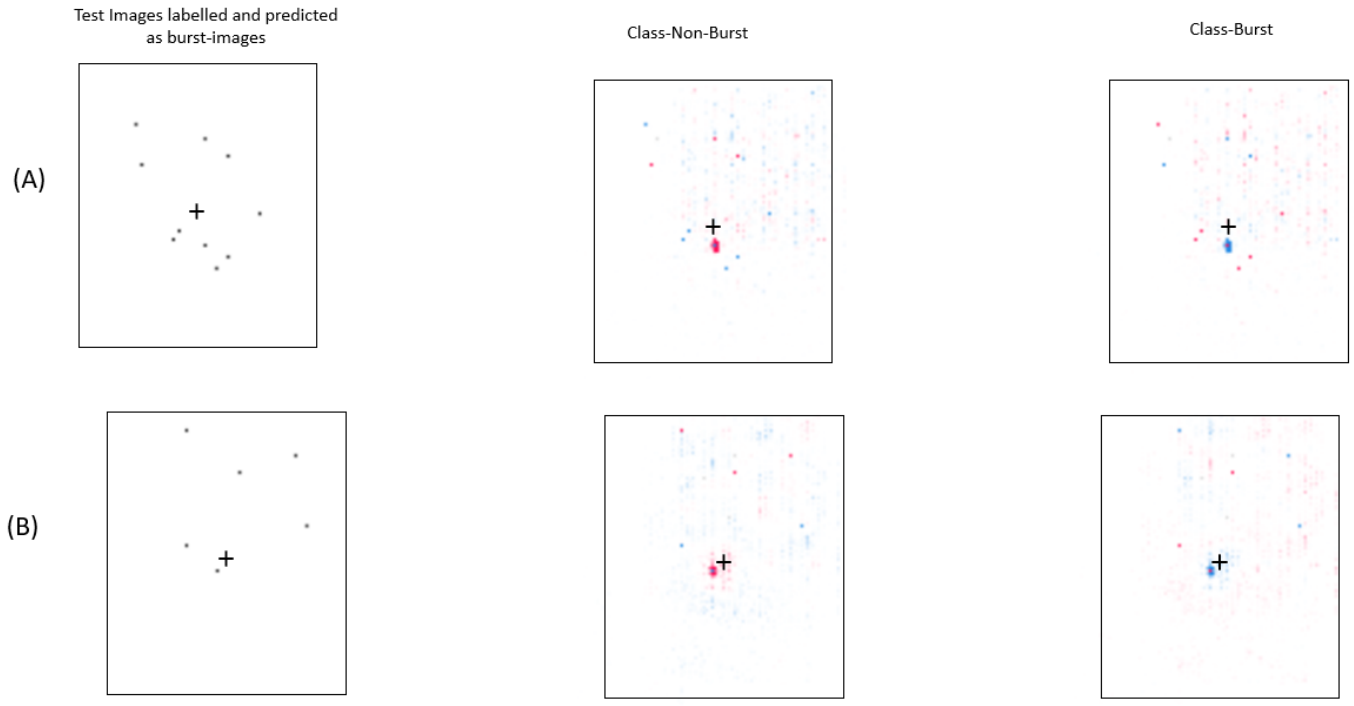

Figure 8.3. Rows (A) and (B) describe the actual test image in the first column vs the interpreted output by SHAP in the middle and the right column, i.e., the first column of images represent the test images that were originally labelled as burst-images and predicted by the label also as burst-images. The middle and the right columns represent which collective patterns from each of the classes the test image matches to more, explaining its prediction. All the explanations are from the medium-complexity model on the dataset- spatio-temporal all bursts without endogenously active neurons with a window of 100 time steps.

Firstly, from Figure 8.3 we notice that the images produced after removing endogenously active neurons were less noisy as compared to the all bursts dataset. The results of CNN predictions of image classification in burst and non-burst for the bursts produced after the endogenously active neurons are removed were significantly better than the dataset with all the bursts. It is interesting

to note that the interpretable images on the rightmost column representing the explanations for images being bursts, highlights that all the images that are predicted as a burst have a “red dot” right in the middle, even though our new spatio-temporal avalanche method produces burst start times that mark the true beginning and the pre-burst sequences do not have as much activity as the bursts. This result is consistent with the visual patterns seen in Figure 8.1 for the dataset without removing any neuron, even after denoising the images. Not only do we see the presence of burst initiation activities at the burst origin, but we also see that the clusters of spiking activity are spread out across the network. Thus, giving us confidence that the burst initiation patterns around the burst origin as well as the presence of neuronal clusters with or without endogenously active neurons are crucial features for an image being a burst image. The Figure 8.4 explains why the individual test images in Figure 8.3 were classified as bursts instead of non-burst.

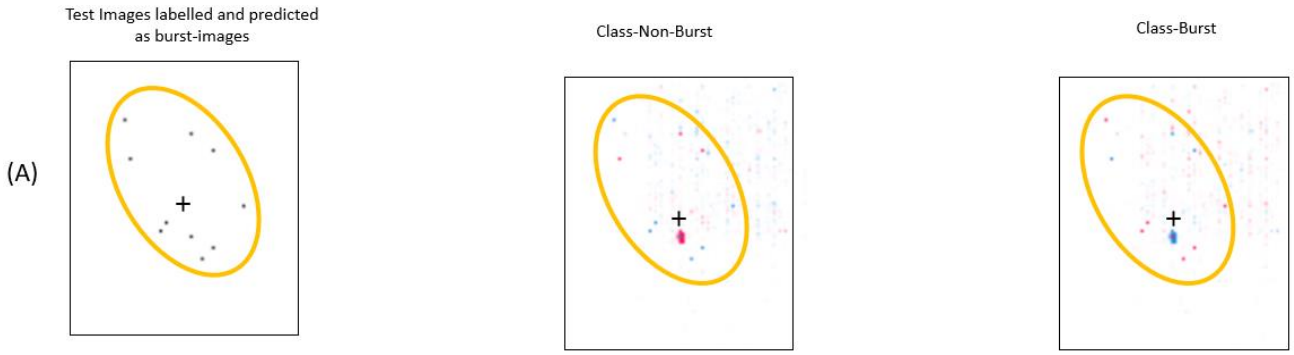

Figure 8.4. Explaining the patterns found from SHAPlift to label this test image as a burst for images without endogenously active neurons

Zooming into row (A) in Figure 8.4, we see that each of the spikes in the test data are colored with SHAP color, where presence of more red colors is predicted of the image being in that class. When we look at the interpretable image produced by SHAP for the burst class, we see that the number of red colored spikes from our test image is 7 out 10, whereas the number of “red dots” for the class non-burst is only 3 out of 10. This suggests why the test image was labelled as

a burst instead of non-burst. Re-emphasizing our discussion in the previous paragraph, we see from Figure 8.3 that the presence of this "red spike" at the center/ at burst origin is highly predictive of any image being a burst.

Results of Visual Model Interpretability on Temporal Bursts aka old method of Burst beginning:

The results from the old method of producing burst start times was as expected, we had 100% accuracy and F1-scores from the CNN models.

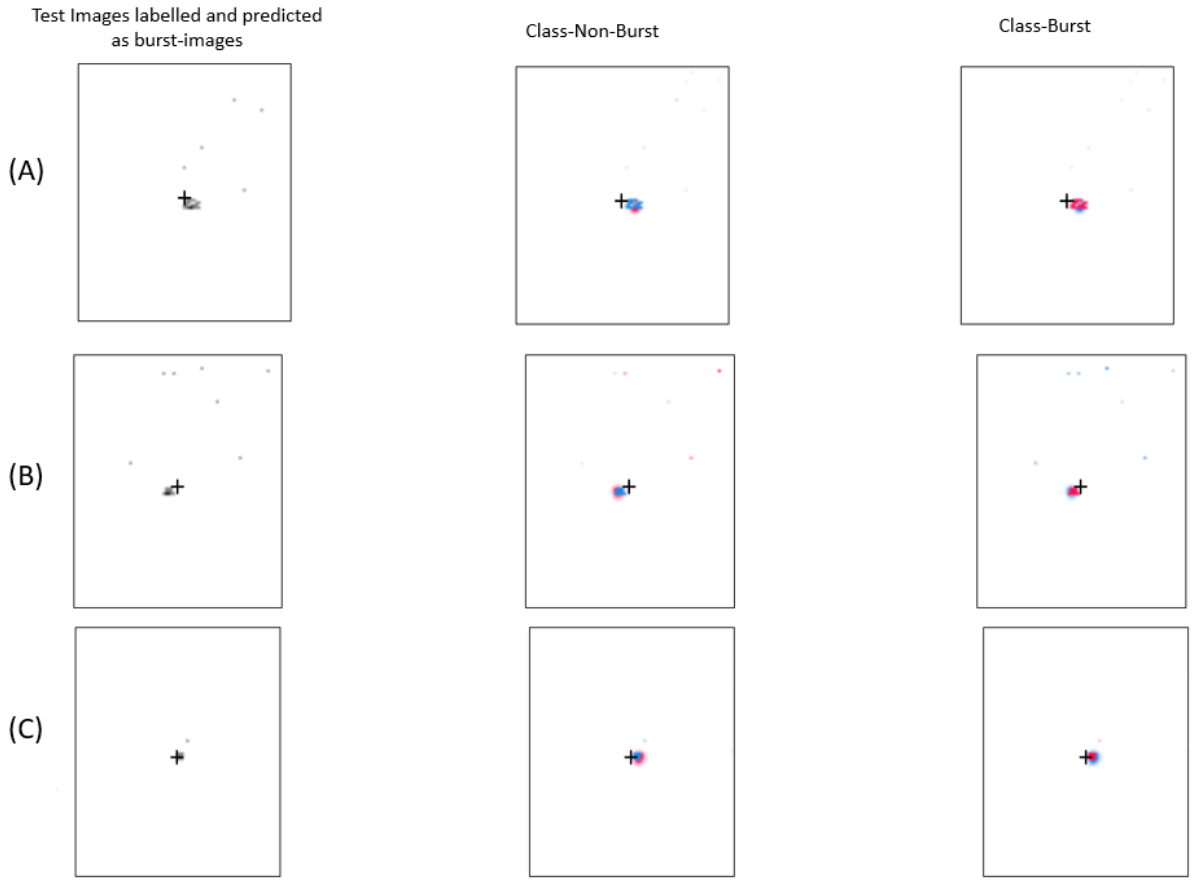

Figure 8.5. Rows (A) and (B) represent the test images that were originally labelled as burst-images and predicted by the label also as burst-images. The middle and the right columns represent which collective patterns from each of the classes the test image matches to more, explaining its prediction. Row (A) shows the predictions from low-complexity CNN with spiking activity in the window of 50 timesteps and row (B) shows the predictions from medium complexity CNN with spiking activity in the window of 50 timesteps and row (C) shows the results from medium-complexity model with a window of 100, all for the dataset- bursts extracted using the old-method of burst time markers.

As can be seen from Figure 8.5, the aggregation of a majority of spiking activity present in our test set and training data itself leads the model to be biased towards only considering this aggregation as a pattern to classify the images as being burst or non-bursts. Therefore, the model only expects this activity in the center, if the images have this aggregation activity present, it is predicted as burst otherwise not. An interesting point to note from these images here is that, there

is hardly any background activity in the interpretable images by the images produced on temporal method, that the model found important contributors to the classes. We see that the behavior around the burst is very faint, almost non-existent, indicating that the model is only looking for the activity in the center, due to the nature of our biased data. Comparing this with the result after removing the endogenously active neurons, even though the model expected activity at the center to be a burst image in Figures 8.3 and 8.4, it still had a faint presence of activity clusters around the center as contributors to be a burst image, whereas, here we have none. Therefore, confirming from images in Figure 8.5, the model is biased in its prediction due to the presence of burst activity.

Looking at local individual analysis we see from Figure 8.6, that with the late burst start time markers, the initial burst activity is already present in our images.

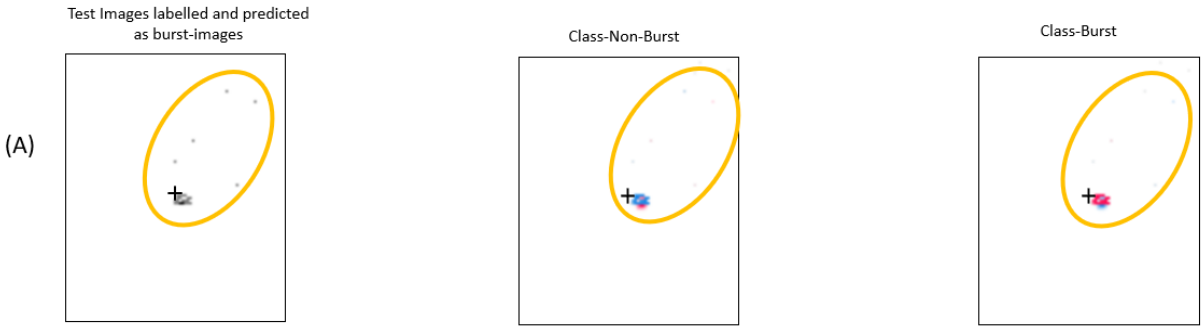

Figure 8.6. Explaining the patterns found from SHAPlift to label this test image as a burst

As we can see from Figure 8.6, the model has picked up this pool of spikes that are part of the bursts, as being predictive of being burst images as expected.

8.3.2 *Understanding Burst initiation patterns from the difference in the features from Burst and non-burst images*

Now that we have seen some results about which collective patterns in our data are predictive of an image being a burst, this section summarizes our visual results on which collective features from our data make an image a burst image or a non-burst image.

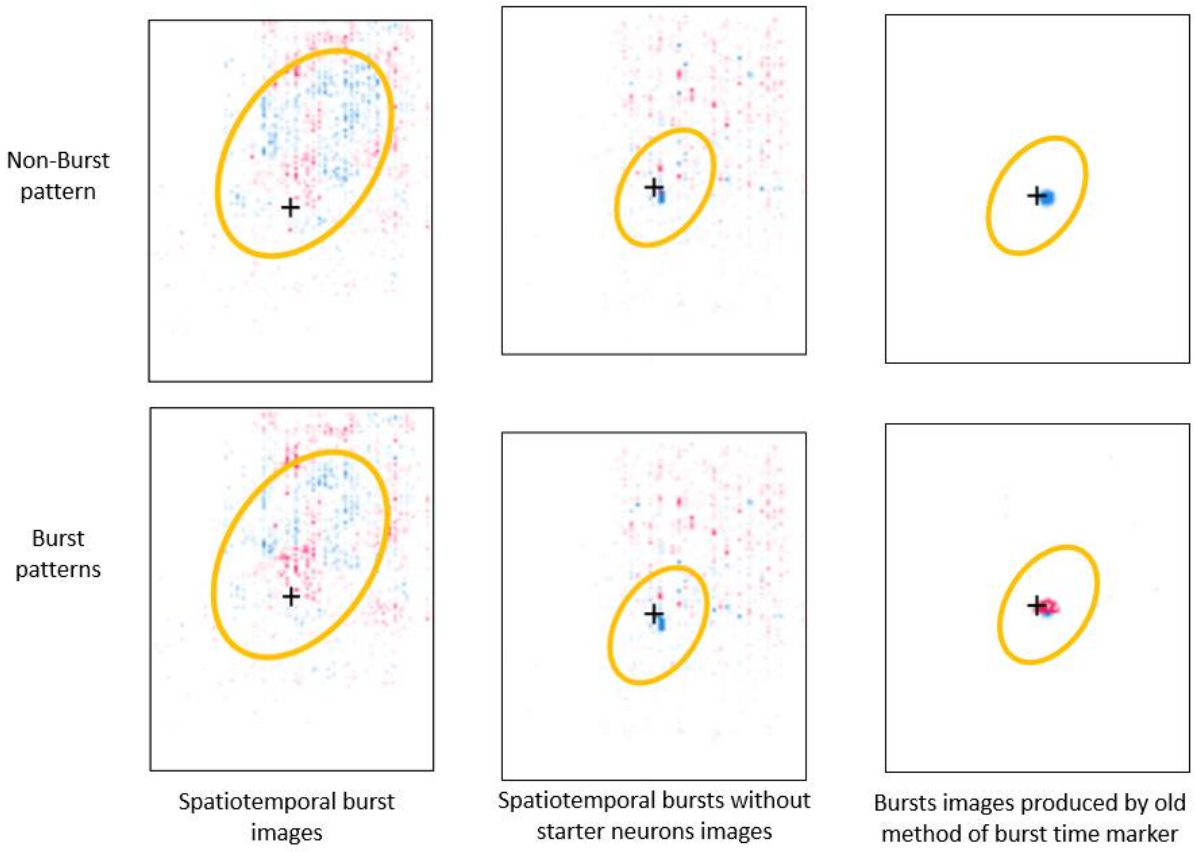

Figure 8.7. Side by side comparison of features corresponding to burst initiation from burst-images and non-burst images

Comparing the burst initiation behaviors learnt by CNN models and SHAPLift, from Figure 8.7, we find that regions of the spatio-temporal images that contribute to being a burst are more concentrated near the burst origin, in spite of having well-defined burst start times that make our pre-burst images non-connected with the burst activity. This finding is consistent in both the

datasets, with and without the endogenously active neurons' spikes. When the endogenously active neurons' s spikes are removed, our data becomes less noisy as the background activity between the bursts almost diminish, showing the underlying burst initiation patterns more clearly around the center. There is also a presence of a "red dot" right at the center of the results of the images produced after removing endogenously active neuronal spikes is consistent with all the tests done. Apart from expecting the activity at the center, we also see the presence of activity clusters spread across the network and this result is consistent with both the datasets with and without removing the endogenously active neurons. Lastly, this result confirms that, due to the nature of spurious data generated from the old method of marking burst times, the deep learning models are highly biased towards looking for the aggregation of activity only at the burst origin. This wraps up all the results from the experiments performed in this thesis.

Chapter 9. CONCLUSION AND FUTURE WORK

This thesis investigated spatiotemporal patterns originating during burst initiation in full spiking activity from a closed loop leaky-integrate-and-fire simulation of neuron-activity-driven network development. The goal of this work is to perform data analysis of this spatiotemporal data and apply machine learning techniques to investigate how patterns in the background activity contribute to predicting the burst initiation. We performed these experiments as a supervised classification model where we labelled the background activity closer to burst initiation points as pre-burst and activity farther away from the bursts as non-burst. These two categories then became our input to the classification model where understanding which category leads to a higher prediction of the burst category would give us some insights about burst initiation patterns.

We found that previous machine learning classification work achieved a high accuracy due to wrong burst time markers. We found that the pre-burst activity passed as the ground truth input to the classification model included the start of burst activity. We therefore applied a new method of finding the burst start times based on a spatio-temporal method that groups avalanche together that are both temporally and spatially close. Upon re-running the experiments of classification in pre-burst and non-burst with the new spatiotemporal avalanche method, we found the degree of predictability reduced as expected due to the unbiased data, explained in detail in the Results chapter.

To further improve the predictability as well as derive visual patterns from the background activity, we converted our spike train data into spatio-temporal images and performed image classification. We used Convolutional Neural Networks of various complexities to test the effectiveness of various deep learning algorithms and network layers in extracting important information from images. We also used a deep pre-trained AlexNet using transfer learning known

for its effectiveness on most of the image classification tasks today. After performing experiments with image classification as well as non-image ML classification, we found that spatiotemporal image classification has 20% better performance than the previous method (firing time, location). For image classification, as explained in the Results chapter, we found that the model with the least complexity achieved the best results for our datasets. It is known that one reason to add more network layers to make a deeper model is so that the layers convolve over more of the input data. When a network does a convolution on an input, it extracts a relevant feature such as edges, shapes, colors, etc. Therefore, deeper models allow the network to perform more convolutions and let it extract more complex features with every convolution. However, the sparse nature of our spatio-temporal images restricted the effectiveness of deeper layers. It is possible that the first few layers were successful in capturing all the relevant information from the model, going deeper did help in capturing any more features rather, adding more layers beyond a certain threshold lead to finding irregularities in the data or information irrelevant to the classification task at hand. It is also possible that the low accuracy of the deeper models could be because the models overfit on the training data, leading to bad performance on testing data. We confirmed this when we found the accuracy of deeper models on training data is 20% higher than the performance on the testing data, whereas the performance was almost similar in terms of our low complexity CNN model.

Studies reveal that that positional information is used by convolutional neural networks to a strong degree as one of the important feature maps extracted from images and plays an important role in classifying [71]. The deeper layers, apart from learning the shape and size of patterns, also learn about location independence. Another reason why deeper and more complex CNNs might not have performed well in our scenario is that our images were already centered, therefore there was no need for position independence to be learned. It is found that zero padding, explained in

detail in the Methods chapter, acts as an anchor from which spatial information is derived, also useful for spatial abstraction and position independence CNNs [71]. Even though we have used zero padding for our experiments, as mentioned due the sparse nature of our images as well as centering, there is no aspect of position information required for our images. Therefore because of these reasons, we speculate that a simple CNN model with fewer layers can extract all major features and information from our images to differentiate between pre-burst and non-burst sparse images.

Additionally, to be able to visually reveal the underlying patterns present in spatio-temporal images based on which the CNN image classification model decides into either of the two classes, we implemented SHAP model interpretability analysis. This revealed findings about burst initiation patterns, as shown in the Results section. It is to be noted that instead of using a machine learning model for a typical production environment for a software system, we use model interpretability as it has a significant contribution in understanding the visual patterns in a burst image. From the visual patterns in model interpretability outputs, we found that burst initiation patterns denoted by high density of activity are found around the burst origins even after we confirmed with the spatiotemporal avalanche method that pre-burst activity is void of any bursting activity. We also found what might be functional communities of neurons as initiation patterns, spread across the network and not just centered around the origin, like the findings by Maeda *et al.* [15]. These putative functional communities were identified in the dataset that included all neurons' spikes and thought this could be because those spikes acted like background noise. However, on removal of endogenously active neurons' spikes we still found these clusters of activity across the network (although the signature of this was faint). When comparing the expected patterns to classify a burst image from a non-burst image using SHAP, we found that for

non-bursts, absence of concentration of activity around the center of images makes it unsuitable for burst initiation. It was also found that burst initiation patterns were more evident, and we had a higher predictability with larger window sizes, for example, window size of 100 has 10% better performance that window size of 50 for the experiments.

This thesis work is among the first few applications of machine learning to investigate burst initiation patterns in spike train data of cortical neural networks. The results and visualizations allow us to conclude that there is presence of localized spatiotemporal patterns for characterizing burst trigger or initiation at the burst origin as well as spread across the network in functional communities or clusters of spiking activity.

There are a few directions for future research work in terms of deeper dives into burst initiation patterns. Firstly, work by Lonardoni *et al.* [69] revealed that network bursts could be sorted into a few classes (i.e., < 10) based on their spatiotemporal patterns, with each class sharing properties with all the bursts in that class, such as a similar average propagation trajectory and origin. In our data, different phases of the simulation have bursts originating from different confined locations, as shown in Figure 3.1. We can consider these as burst classes and extend our approach of finding visual patterns in each of these classes separately as one of the directions of future work.

Secondly, since we found that the performance of the models' predictability was significantly better with denoised versions of images, i.e. after removing endogenously active neurons' spikes, one could investigate further on the burst initiation patterns after removing these spikes. Additionally, we also found that only 100 non-burst avalanches remain after removing the endogenously active neurons' spikes; therefore, this raises questions about whether there is much participation in background activity / avalanches from anything other than endogenously active

neurons. Further analysis can therefore be performed to analyze the behavior of non-burst avalanches, whether they exist or are truly a cause from the uncorrelated, random firing of the endogenously active cells.

Lastly, this work was done to find burst initiation patterns without considering position dependence in the images; another possibility for future work could be to consider position dependence in order to predict burst origin itself on the spatiotemporally created burst start times. Apart from using images, one could also use time-series based analysis to help with burst start time prediction and find seasonality and trends in the activity relative to timesteps.

BIBLIOGRAPHY

[1] A. Fornito, A. Zalesky, and M. Breakspear, "Graph analysis of the human connectome: Promise, progress, and pitfalls," *NeuroImage*, vol. 80, pp. 426–444, Oct. 2013, doi: 10.1016/j.neuroimage.2013.04.087.

[2] "Why Study the Brain?," *The BRAIN Initiative*. https://www.braininitiative.org/why-study-the-brain/ (accessed Jun. 02, 2020).

[3] J. M. Beggs and D. Plenz, "Neuronal Avalanches in Neocortical Circuits," *J. Neurosci.*, vol. 23, no. 35, pp. 11167–11177, Dec. 2003, doi: 10.1523/JNEUROSCI.23-35-11167.2003.

[4] "Central Nervous System: brain and spinal cord," Nov. 10, 2017. https://qbi.uq.edu.au/brain/brain-anatomy/central-nervous-system-brain-and-spinal-cord (accessed Jun. 02, 2020).

[5] "Brain Neurons & Synapses | Action Potentials & Neurotransmission," *The Human Memory*, Sep. 25, 2019. https://human-memory.net/brain-neurons-synapses/ (accessed Jun. 02, 2020).

[6] "What Is a Neuron? Function, Parts, Structure, Types, and More," *Healthline*. https://www.healthline.com/health/neurons (accessed Jun. 02, 2020).

[7] "The synapse (article) | Human biology," *Khan Academy*. https://www.khanacademy.org/science/biology/human-biology/neuron-nervous-system/a/the-synapse (accessed Jun. 02, 2020).

[8] J. Gordon, S. Amini, and M. K. White, "General overview of neuronal cell culture," *Methods Mol Biol*, vol. 1078, pp. 1–8, 2013, doi: 10.1007/978-1-62703-640-5_1.

[9] "Brain Cell Culture - an overview | ScienceDirect Topics." https://www.sciencedirect.com/topics/medicine-and-dentistry/brain-cell-culture (accessed Jun. 02, 2020).

[10] L. Volobueva, X. Sun, Y.-B. Ouyang, and R. G. Giffard, "Cell Culture: Primary Neural Cells☆," in *Reference Module in Neuroscience and Biobehavioral Psychology*, Elsevier, 2017.

[11] I. Y. Tyukin *et al.*, "Simple model of complex dynamics of activity patterns in developing networks of neuronal cultures," *PLoS ONE*, vol. 14, no. 6, p. e0218304, Jun. 2019, doi: 10.1371/journal.pone.0218304.

[12] T. Fardet, M. Ballandras, S. Bottani, S. Métens, and P. Monceau, "Understanding the Generation of Network Bursts by Adaptive Oscillatory Neurons," *Front Neurosci*, vol. 12, Feb. 2018, doi: 10.3389/fnins.2018.00041.

[13] A. G. Blankenship and M. B. Feller, “Mechanisms underlying spontaneous patterned activity in developing neural circuits,” *Nat. Rev. Neurosci.*, vol. 11, no. 1, pp. 18–29, Jan. 2010, doi: 10.1038/nrn2759.

[14] M. I. Ham, V. Gintautas, and G. W. Gross, “Spontaneous coordinated activity in cultured networks: Analysis of multiple ignition sites, primary circuits, burst phase delay distributions and functional structures,” *arXiv:1004.2072 [q-bio]*, Apr. 2010, Accessed: Jun. 02, 2020. [Online]. Available: http://arxiv.org/abs/1004.2072.

[15] E. Maeda, H. P. Robinson, and A. Kawana, “The mechanisms of generation and propagation of synchronized bursting in developing networks of cortical neurons,” *J. Neurosci.*, vol. 15, no. 10, pp. 6834–6845, Oct. 1995.

[16] N. Maheswaranathan, S. P. D. Ferrari, A. M. J. VanDongen, and C. P. D. Henriquez, “Emergent bursting and synchrony in computer simulations of neuronal cultures,” *Front. Comput. Neurosci.*, vol. 6, 2012, doi: 10.3389/fncom.2012.00015.

[17] “Frontiers | Toward an Integration of Deep Learning and Neuroscience | Frontiers in Computational Neuroscience.” https://www.frontiersin.org/articles/10.3389/fncom.2016.00094/full (accessed Jun. 02, 2020).

[18] J. Y. Lee, M. Stiber, and D. Si, “Machine Learning of Spatiotemporal Bursting Behavior in Developing Neural Networks,” *Conf Proc IEEE Eng Med Biol Soc*, vol. 2018, pp. 348–351, 2018, doi: 10.1109/EMBC.2018.8512358.

[19] F. Kawasaki and M. Stiber, “Stimulation effects on cortical culture development,” p. 1.

[20] J. Richiardi, S. Achard, H. Bunke, and D. Van De Ville, “Machine Learning with Brain Graphs: Predictive Modeling Approaches for Functional Imaging in Systems Neuroscience,” *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 58–70, May 2013, doi: 10.1109/MSP.2012.2233865.

[21] F. Pereira, T. Mitchell, and M. Botvinick, “Machine learning classifiers and fMRI: a tutorial overview,” *Neuroimage*, vol. 45, no. 1 Suppl, pp. S199-209, Mar. 2009, doi: 10.1016/j.neuroimage.2008.11.007.

[22] “Machine learning in bioinformatics | Briefings in Bioinformatics | Oxford Academic.” https://academic.oup.com/bib/article/7/1/86/264025 (accessed Jun. 02, 2020).

[23] I. M. Park, S. Seth, A. R. C. Paiva, L. Li, and J. C. Principe, “Kernel methods on spike train space for neuroscience: a tutorial,” *IEEE Signal Process. Mag.*, vol. 30, no. 4, pp. 149–160, Jul. 2013, doi: 10.1109/MSP.2013.2251072.

[24] A. R. C. Paiva, I. Park, and J. C. Príncipe, “Chapter 8 - Inner Products for Representation and Learning in the Spike Train Domain,” in *Statistical Signal Processing*

for Neuroscience and Neurotechnology, K. G. Oweiss, Ed. Oxford: Academic Press, 2010, pp. 265–309.

[25] N. K. Kasabov, "NeuCube: A spiking neural network architecture for mapping, learning and understanding of spatio-temporal brain data," *Neural Networks*, vol. 52, pp. 62–76, Apr. 2014, doi: 10.1016/j.neunet.2014.01.006.

[26] N. Paltrinieri, L. Comfort, and G. Reniers, "Learning about risk: Machine learning for risk assessment," *Safety Science*, vol. 118, pp. 475–486, Oct. 2019, doi: 10.1016/j.ssci.2019.06.001.

[27] C. Teeter *et al.*, "Generalized leaky integrate-and-fire models classify multiple neuron types," *Nature Communications*, vol. 9, no. 1, p. 709, Feb. 2018, doi: 10.1038/s41467-017-02717-4.

[28] Z. C. Chao, D. J. Bakkum, and S. M. Potter, "Region-specific network plasticity in simulated and living cortical networks: comparison of the center of activity trajectory (CAT) with other statistics," *J. Neural Eng.*, vol. 4, no. 3, pp. 294–308, Jul. 2007, doi: 10.1088/1741-2560/4/3/015.

[29] N. R. C. (US) C. on R. O. in Biology, *The Nervous System and Behavior*. National Academies Press (US), 1989.

[30] "Overview of neuron structure and function (article)," *Khan Academy*. https://www.khanacademy.org/science/biology/human-biology/neuron-nervous-system/a/overview-of-neuron-structure-and-function (accessed Jun. 02, 2020).

[31] "Self-Directed Neuroplasticity," *FitMind*. https://www.fitmind.co/blog-collection/self-directed-neuroplasticity (accessed Jun. 02, 2020).

[32] "Dissociated Cell Culture - an overview | ScienceDirect Topics." https://www.sciencedirect.com/topics/nursing-and-health-professions/dissociated-cell-culture (accessed Jun. 02, 2020).

[33] "Signal Processing in Neuroscience | Xiaoli Li | Springer." https://www.springer.com/gp/book/9789811018213 (accessed Jun. 02, 2020).

[34] W. Gerstner, H. Sprekeler, and G. Deco, "Theory and Simulation in Neuroscience," *Science*, vol. 338, no. 6103, pp. 60–65, Oct. 2012, doi: 10.1126/science.1227356.

[35] C. Koch and R. C. Reid, "Neuroscience: Observatories of the mind," *Nature*, vol. 483, no. 7390, pp. 397–398, Mar. 2012, doi: 10.1038/483397a.

[36] J.-B. Poline *et al.*, "Data sharing in neuroimaging research," *Front Neuroinform*, vol. 6, p. 9, 2012, doi: 10.3389/fninf.2012.00009.

[37] B. A. Richards *et al.*, "A deep learning framework for neuroscience," *Nature Neuroscience*, vol. 22, no. 11, pp. 1761–1770, Nov. 2019, doi: 10.1038/s41593-019-0520-2.

[38] D. L. K. Yamins and J. J. DiCarlo, “Using goal-driven deep learning models to understand sensory cortex,” *Nat. Neurosci.*, vol. 19, no. 3, pp. 356–365, Mar. 2016, doi: 10.1038/nn.4244.

[39] “Reinforcement Learning, Fast and Slow: Trends in Cognitive Sciences.” https://www.cell.com/trends/cognitive-sciences/fulltext/S1364-6613(19)30061-0 (accessed Jun. 02, 2020).

[40] N. Kriegeskorte and P. K. Douglas, “Cognitive computational neuroscience,” *Nat. Neurosci.*, vol. 21, no. 9, pp. 1148–1160, 2018, doi: 10.1038/s41593-018-0210-5.

[41] L. Zhang, J. Tan, D. Han, and H. Zhu, “From machine learning to deep learning: progress in machine intelligence for rational drug discovery,” *Drug Discov. Today*, vol. 22, no. 11, pp. 1680–1685, 2017, doi: 10.1016/j.drudis.2017.08.010.

[42] “Parallel Distributed Processing, Volume 1 | The MIT Press.” https://mitpress.mit.edu/books/parallel-distributed-processing-volume-1 (accessed Jun. 02, 2020).

[43] S. B. Online, “1. The Neural Network - Fundamentals of Deep Learning.” https://learning.oreilly.com/library/view/fundamentals-of-deep/9781491925607/ch01.html (accessed Jun. 02, 2020).

[44] H. Kamioka *et al.*, “Effectiveness of music therapy: a summary of systematic reviews based on randomized controlled trials of music interventions,” *Patient Prefer Adherence*, vol. 8, pp. 727–754, May 2014, doi: 10.2147/PPA.S61340.

[45] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[46] L. Breiman, “Random Forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct. 2001, doi: 10.1023/A:1010933404324.

[47] L. Breiman, “Random Forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct. 2001, doi: 10.1023/A:1010933404324.

[48] D. Newby, A. A. Freitas, and T. Ghafourian, “Decision trees to characterise the roles of permeability and solubility on the prediction of oral absorption,” *European Journal of Medicinal Chemistry*, vol. 90, pp. 751–765, Jan. 2015, doi: 10.1016/j.ejmech.2014.12.006.

[49] F. Kawasaki and M. Stiber, “A simple model of cortical culture growth: burst property dependence on network composition and activity,” *Biol Cybern*, vol. 108, no. 4, pp. 423–443, Aug. 2014, doi: 10.1007/s00422-014-0611-9.

[50] “(6) (PDF) Implications of activity dependent neurite outgrowth for neuronal morphology and network development.” https://www.researchgate.net/publication/15308679_Implications_of_activity_dependent_ne

urite_outgrowth_for_neuronal_morphology_and_network_development (accessed Jun. 02, 2020).

[51] "(6) (PDF) Complex Periodic Behaviour in a Neural Network Model with Activity-Dependent Neurite Outgrowth." https://www.researchgate.net/publication/14441033_Complex_Periodic_Behaviour_in_a_Neural_Network_Model_with_Activity-Dependent_Neurite_Outgrowth (accessed Jun. 02, 2020).

[52] Zhouhan Lin, Yushi Chen, Xing Zhao, and Gang Wang, "Spectral-spatial classification of hyperspectral image using autoencoders," in *2013 9th International Conference on Information, Communications & Signal Processing*, Tainan, Taiwan, Dec. 2013, pp. 1–5, doi: 10.1109/ICICS.2013.6782778.

[53] "Image based fruit category classification by 13-layer deep convolutional neural network and data augmentation | SpringerLink." https://link.springer.com/article/10.1007/s11042-017-5243-3 (accessed Jun. 02, 2020).

[54] R. A. Minhas, A. Javed, A. Irtaza, M. T. Mahmood, and Y. B. Joo, "Shot Classification of Field Sports Videos Using AlexNet Convolutional Neural Network," *Applied Sciences*, vol. 9, no. 3, p. 483, Jan. 2019, doi: 10.3390/app9030483.

[55] M. K. Pujari Mohit Sewak, Pradeep, *Practical Convolutional Neural Networks*. .

[56] A. A. Almisreb, N. Jamil and N. M. Din, "Utilizing AlexNet Deep Transfer Learning for Ear Recognition," 2018 Fourth International Conference on Information Retrieval and Knowledge Management (CAMP), Kota Kinabalu, 2018, pp. 1-5, doi: 10.1109/INFRKM.2018.8464769.

[57] M. Rizwan, "AlexNet Implementation Using Keras," *engMRK*, Oct. 05, 2018. https://engmrk.com/alexnet-implementation-using-keras/ (accessed Jun. 02, 2020).

[58] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.

[59] C. Molnar, *Chapter 2 Interpretability | Interpretable Machine Learning*.

[60] Y. Bengio, A. Courville, and P. Vincent, "Representation Learning: A Review and New Perspectives," *arXiv:1206.5538 [cs]*, Apr. 2014, Accessed: Jun. 02, 2020. [Online]. Available: http://arxiv.org/abs/1206.5538.

[61] "Krüger A, et al. (2013) | SGD." https://www.yeastgenome.org/reference/S000155488 (accessed Jun. 02, 2020).

[62] C.-C. Lin, J. Seikowski, A. Pérez-Lara, R. Jahn, C. Höbartner, and P. J. Walla, "Control of membrane gaps by synaptotagmin-Ca 2+ measured with a novel membrane distance

ruler," *Nature Communications*, vol. 5, no. 1, p. 5859, Dec. 2014, doi: 10.1038/ncomms6859.

[63] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht, "The Marginal Value of Adaptive Gradient Methods in Machine Learning," *arXiv:1705.08292 [cs, stat]*, May 2018, Accessed: Jun. 02, 2020. [Online]. Available: http://arxiv.org/abs/1705.08292.

[64] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning Important Features Through Propagating Activation Differences," *arXiv:1704.02685 [cs]*, Oct. 2019, Accessed: Jun. 02, 2020. [Online]. Available: http://arxiv.org/abs/1704.02685.

[65] "kundajelab/deeplift: Public facing deeplift repo." https://github.com/kundajelab/deeplift (accessed Jun. 02, 2020).

[66] "Explainers — SHAP latest documentation." https://shap.readthedocs.io/en/latest/#shap.summary_plot (accessed Jun. 02, 2020).

[67] "5.9 Shapley Values | Interpretable Machine Learning." https://christophm.github.io/interpretable-ml-book/shapley.html (accessed Jun. 02, 2020).

[68] "Accelerating large-scale simulations of cortical neuronal network development" (2014)

[69] D. Lonardoni, H. Amin, S. D. Marco, A. Maccione, L. Berdondini, and T. Nieus, "Recurrently connected and localized neuronal communities initiate coordinated spontaneous activity in neuronal networks," PLOS Computational Biology, vol. 13, no. 7, p. e1005672, Jul. 2017, doi: 10.1371/journal.pcbi.1005672.

[70] F. Zeldenrust, W. J. Wadman, and B. Englitz, "Neural Coding With Bursts—Current State and Future Perspectives," *Front. Comput. Neurosci.*, vol. 12, 2018, doi: 10.3389/fncom.2018.00048.

[71] M. A. Islam*, S. Jia*, and N. D. B. Bruce, "How much Position Information Do Convolutional Neural Networks Encode?," presented at the International Conference on Learning Representations, Sep. 2019, Accessed: Jun. 19, 2020. [Online]. Available: https://openreview.net/forum?id=rJeB36NKvB.