Understanding Neural Burst Patterns Through Graph Neural Network Explainability in Simulated Neuronal Networks

Hari Priya Dhanasekaran

A Masters Thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science and Software Engineering

University of Washington

2025

Reading Committee:

Dr. Michael Stiber, Chair

Dr. Munehiro Fukuda

Dr. Wooyoung Kim

Program Authorized to Offer Degree:

Master of Science in Computer Science and Software Engineering

University of Washington

Abstract

Understanding Neural Burst Patterns Through Graph Neural Network Explainability in Simulated Neuronal Networks

Hari Priya Dhanasekaran

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

Spontaneous bursting activity in neural networks represents a fundamental mode of information processing in the brain, yet the mechanisms triggering these synchronized events remain poorly understood. While graph-based representations of neural networks are established, discovering the specific connectivity and activity patterns that predict burst initiation remains a significant challenge. This work uses GNNs to classify and explain burst initiation in Graphitti-simulated cortical networks by representing neurons as nodes with temporal firing statistics and synapses as edges, thereby integrating activity patterns with network architecture. To move beyond black-box classification, we applied GNNExplainer to identify the minimal neural connectivity patterns driving model predictions. This explainability analysis revealed which specific neurons and synaptic connections the model deemed most critical for each prediction. This work demonstrates how explainable AI can transform our understanding of complex neural dynamics, providing insights that pure predictive modeling cannot offer. By combining the representation power of graph neural networks with explainability techniques, we bridge the gap between prediction and understanding. Our findings challenge prevailing views of burst initiation as a localized phenomenon, instead revealing the critical role of distributed precursor patterns in driving network-wide synchronization. This methodology opens new avenues for investigating emergent behaviors in complex networks.

TABLE OF CONTENTS

LIST OF FIGURES

Figure Number Page

LIST OF TABLES

Table Number Page

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my advisor, Dr. Michael Stiber, for his unwavering mentorship, thoughtful feedback, patience and steady encouragement throughout this research journey. His expertise and vision have been invaluable to my growth. I am also thankful to my committee members, Dr. Munehiro Fukuda and Dr. Wooyoung Kim for their time and generous support.

I am sincerely grateful to the Intelligent Networks Lab for fostering a collaborative and inspiring research environment. Special thanks to Vanessa Arndorfer and Marina Rosenwald for their contributions to Graphitti that supported this work. I am grateful to the CSS department for providing financial support though Graduate Student Teaching Assistantship. My warmest thanks to my supervisor, Professor Yang Peng, for sharing his knowledge and making me a better teacher.

Finally, I owe everything to my family and friends. Thank you for your unconditional support, patience, and belief in me—especially during the most challenging phases of this thesis.

DEDICATION

To my parents — for their unwavering support, respect for my choices, and constant encouragement; you are my pillars of strength. To my sister — for travelling beside me through every step of my career and life, and for her steadfast support. To my friend Nikhil — for being a true pillar of support and a source of motivation. To my grandparents — for their selfless care, warm thoughts, and the quiet love that has always guided me.

And to all my family and friends whose patience, belief, and kindness have made this journey possible.

Chapter 1

INTRODUCTION

Decoding how large populations of neurons self-organize into coherent, information-bearing activity is a grand challenge that spans basic neuroscience. In dissociated cortical cultures and *in silico* growth simulations, networks routinely transition from sparse, uncorrelated firing to brisk, whole-culture bursts that sweep across the dish in wave-like fashion [16]. Closed-loop models that couple activity to neurite out-growth faithfully reproduce this developmental arc, and the addition of spike-timing-dependent plasticity (STDP) further sharpens synaptic weights, shortens bursts, and yields the heterogeneous hub-and-spoke topologies seen in mature tissue [1].

Classical tools peri-stimulus time histograms, correlation functions, avalanche counts have been indispensable, yet they treat each neuron as an independent time series, masking the graph substrate on which collective dynamics unfold. Pioneering machine-learning studies began to bridge the gap: [22] showed that off-the-shelf classifiers can predict burst onset from pre-burst spike patterns with high accuracy, while [39] demonstrated that convolutional neural networks trained on spatiotemporal images can localize "trigger patterns" that ignite network bursts. Nevertheless, these approaches flatten connectivity into pixels or feature vectors, forfeiting the explicit relational (graph) structure that shapes neural computation.

Graph Neural Networks (GNNs) may provide the missing ingredient. By embedding neurons as nodes and functional or structural links as edges, GNNs perform message-passing that melds node-level firing statistics with network topology. However, a unified framework that represents neurons in their true graph context, learns multi-scale patterns directly from raw spikes, and yields interpretable insight remains elusive.

This thesis begins to address that gap. We construct inductive GNN model that ingest high-density spike trains from simulations, transforming raw rasters into time-resolved functional graphs. Coupling these models with GNNExplainer allows us to isolate minimal subgraphs and the critical millisecond-scale interactions within them that start spontaneous bursts.

Chapter 2

BACKGROUND

2.1 Neuroscience Background

2.1.1 The Nervous System, Neurons, and Spikes

The nervous system is a highly sophisticated network composed primarily of neurons (nerve cells) that communicate via electrochemical signaling. As shown in Figure 2.1, each neuron is structurally characterized by a soma (cell body), dendrites, and an axon. Dendrites function as input sites receiving signals from other neurons, while the soma integrates these signals and generates an output response if the input exceeds a certain threshold. This output response is an electrical impulse called an action potential or spike. Neuronal spikes, or action potentials, are brief electrical impulses generated by neurons when the voltage across their cell membranes surpasses a specific threshold due to excitatory inputs as shown by Figure 2.1. Axons transmit these spikes across synapses to other neurons, thereby forming complex neural networks. The propagation of spikes through these networks generates complex activity patterns critical for various brain functions including sensory processing, motor coordination, and cognitive activities.

2.1.2 Neural Network Behavior

Neural networks, whether biological circuits in vitro/in vivo or their computational models—exhibit rich, emergent dynamics that arise from the interaction of intrinsic neuronal excitability, synaptic connectivity, and external or spontaneous input. At multiple spatial and temporal scales, these systems show a range of activity patterns: irregular asynchronous firing, oscillations at characteristic frequencies, synchronized population events, and trav-

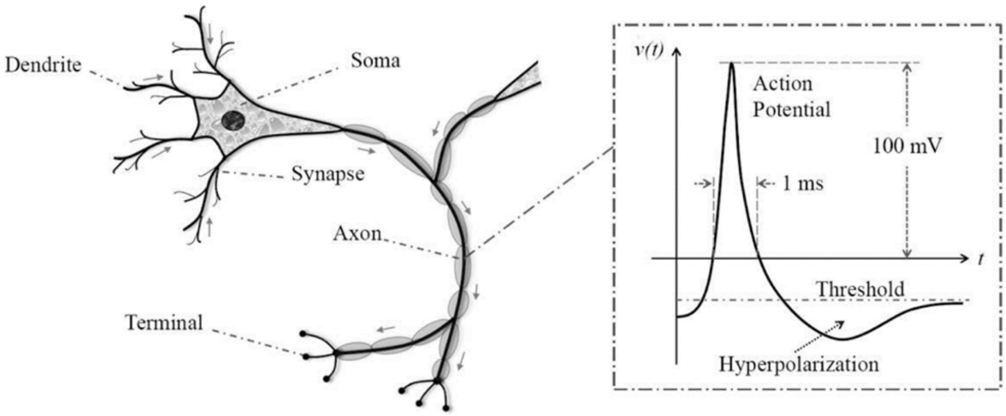


Figure 2.1: Anatomical figure of Neuron and Action Potential. Left figure shows anatomical components of a neuron- dendrites receive inputs, the soma integrates signals, and the axon transmits spikes to downstream neurons via terminals and synapses. Right shows the time course of an action potential, illustrating rapid depolarization, brief overshoot, and subsequent hyperpolarization relative to the threshold (100 mV peak, 1 ms duration). Taken from [10]

eling activity waves. These behaviors reflect the network's functional state, the balance of excitation and inhibition, ongoing plasticity, and the constraints imposed by anatomical and functional connectivity.

Studying neural network behavior therefore requires methods that connect single-neuron statistics (firing rates, spike timing) with network topology, motifs, and spatial embedding. Changes in these observables over development, learning, or pathological conditions often manifest as shifts in burst frequency, propagation speed, or the spatial extent of synchrony. In the rest of this section, we focus specifically on two closely related phenomena—bursting activity and wave-like spontaneous activity.

A prominent feature in neuronal networks, particularly in cortical cultures grown on multi-electrode arrays (MEAs) and observed in computational models, is spontaneous synchronized spiking activity known as network bursts. Network bursts consist of near-simultaneous spikes fired by a significant portion of neurons, interspersed by relatively quiet periods. This behavior emerges prominently in developing neural circuits and contributes crucially to

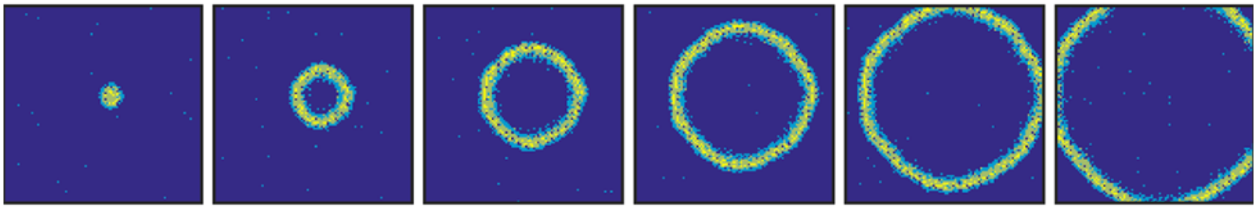


Figure 2.2: Spatiotemporal evolution of a Single burst (from left to right). Each image includes 10 ms of bursting activity and images are 30 ms apart [22]

synaptic refinement, network stabilization, and potentially plays a role in early developmental sensory system functions and rapid anticipatory responses to sensory stimuli. 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. Experimental and simulation studies reveal that neuronal bursts initiate in specific locations and propagate as spatiotemporal waves across the network. Computational simulations using frameworks like Graphitti (formerly BrainGrid), based on leaky integrate-and-fire neuronal models, provide insights into how these waves originate, propagate, and evolve. Such simulations replicate burst phenomena observed experimentally, revealing that initial random spiking activities gradually transition into organized bursts as network connectivity matures.

Wave propagation from these initiation points typically becomes faster and more structured as neural networks mature. Detailed analysis has demonstrated that local neuronal interactions and connectivity significantly influence the initiation and propagation dynamics of these bursts. Understanding such localized trigger patterns is essential not only for basic neuroscience and developmental biology but also for insights into pathological conditions, including epilepsy and other neurological disorders, characterized by abnormal burst dynamics. In summary, investigating spontaneous neuronal network activities especially burst behaviors, SOC, and wave-like propagation—offers profound insights into neuronal network dynamics, developmental neuroscience, and broader implications for neurological health and therapeutic interventions.

2.1.3 Consistent Themes in Burst Initiation

Across species and preparations, a small subset of highly connected “leader” neurons reliably initiate the majority of network bursts. Ham et al. showed that Major Burst Leader (MBL) neurons roughly 17% of the population drove 84 % of bursts in the dissociated rat cortex, with initiation efficacy linked to both high firing rates and short-path connectivity among leaders [12]. Contrary to the notion of a fixed ignition site, initiation loci shift over time and conditions. Maeda et al. (1995) found that spontaneous burst sources wandered across the culture dish from burst to burst, indicating that multiple regions possess initiation capability [30]. Lee (2018) further quantified this during network maturation: early development exhibited widely scattered origins, which gradually condensed into a small set of “active” sites that remained stable for weeks [22]. Recent work in patient-derived Rett syndrome cultures shows even greater fragmentation, with partial bursts emerging in peripheral subnetworks before full coalescence [33]. Episodes in which two or more bursts initiate nearly simultaneously so-called “twin bursts” have been observed both in silico and in vitro. Lee (2018) reported instances where spatially distinct leader ensembles fired in tandem, merged mid-propagation, and produced larger, compound burst events [22]. Such phenomena challenge single-source models and underscore the richness of burst dynamics. Substrate patterning and adhesion influence where bursts start. Chen et al. (2022) cultured rat cortical neurons on micro-patterned PDMS which is the creation of specific, microscopic patterns on the surface of a polydimethylsiloxane (PDMS) material, revealing that burst origins clustered in regions of higher cell–substrate adhesion, linking mechanical microenvironments to initiation sites [4]. The emergence of high-density MEAs, optogenetic “point-and-shoot” stimulation, and phase-based analytic frameworks has dramatically increased spatial and temporal resolution for burst mapping. Tanaka et al. (2024) combined HD-MEA with targeted optogenetic perturbations to causally confirm hub neurons as primary initiators [20].

2.1.4 Gaps and the Need for Graph-Centered Analysis

Despite these rich insights, several critical gaps remain. Traditional thresholding and visualization techniques can pinpoint where bursts originate, but they fall short of explaining why particular neurons are able to tip the network into synchrony. Moreover, existing approaches often decouple node-level firing statistics from the underlying network topology, treating bursts either as purely spatiotemporal phenomena or as abstract graph metrics, but rarely integrating both perspectives. Even with advances such as optogenetic confirmation, the field still lacks a principled framework capable of extracting minimal causal subgraphs that mechanistically explain burst initiation.

The current thesis builds directly upon these previous studies by applying advanced graph neural network (GNN) models and modern explainability techniques. Our work specifically addresses gaps identified by [22] and [40], aiming to elucidate precise, interpretable neural subnetworks responsible for burst initiation in BrainGrid simulation spike data. Through rigorous visualization and analytical validation, we advance our understanding of the fundamental neuronal interactions underlying burst phenomena.

2.2 Machine Learning Background

2.2.1 Classical Statistical and Algorithmic Methods

Spike Train Statistics. Early computational neuroscience relied heavily on statistical techniques to analyze neuronal spike data. Methods such as peri-stimulus time histograms (PSTH) were pivotal in visualizing firing rate changes aligned to stimuli. PSTHs offered intuitive insights but were limited in capturing complex spatiotemporal dynamics. Autocorrelation and cross-correlation analyses further helped quantify neuronal synchrony and temporal patterns, while spectral analyses provided insights into frequency domain characteristics of neuronal oscillations. These methods relied predominantly on manually-tuned, handcrafted statistical features, limiting their scalability and interpretability when analyzing complex network-level behaviors. Despite their simplicity, these foundational techniques established the groundwork for quantifying neural activity and inspired the development of more sophisticated algorithmic approaches. Burst Detection Algorithms Specialized algorithms emerged to accurately identify bursts—periods of rapid neuronal firing. Classical algorithms such as Poisson Surprise, Rank Surprise, and MaxInterval were instrumental, utilizing thresholds on inter-spike intervals or firing rates to detect bursts [5]. Burst detection in our pipeline still relies on well-established methods rather than any newly devised algorithm. Having detected bursts with these classical approaches, our novel contribution lies in coupling each burst's initiation site to the underlying network topology. In summary, classical statistical and algorithmic methods provided valuable initial steps in neural activity analysis but faced inherent challenges, including manual tuning requirements and limited capacity for high-dimensional or network-scale data.

2.2.2 Traditional Machine Learning Approaches

Dimensionality Reduction: The increasing scale and complexity of neuronal datasets led to the adoption of dimensionality reduction techniques such as Principal Component Analysis (PCA). PCA effectively identified principal variance components from high-dimensional spike count matrices, facilitating a simplified representation of neuronal activity. This preprocessing greatly benefited subsequent classification and clustering analyses by reducing computational complexity and highlighting meaningful patterns. Supervised Classification: Supervised machine learning, including Support Vector Machines (SVM) and Random Forests (RF), began to dominate spike pattern classification tasks due to their robustness and high accuracy[35]. Extracted features from spike trains were leveraged by these models to categorize neural conditions or stimuli effectively. For instance, SVMs and RFs have been used to discriminate between pathological and healthy brain states, often outperforming simple threshold-based detectors[28]. Clustering and Unsupervised Methods: Unsupervised learning techniques, such as k-means clustering and Gaussian mixture models, became pivotal in exploring unlabeled neuronal datasets. These methods facilitated the identification of neuron clusters with similar firing patterns, aiding the discovery of distinct neuronal cell types and network states[9]. Despite their advancements, traditional ML methods still depended heavily on handcrafted features, somewhat limiting their scalability and adaptability.

2.2.3 Deep Learning for Neural Data

Building on traditional ML approaches, deep learning models particularly Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have more recently been applied to neural recordings to learn rich spatiotemporal features directly from raw or minimally processed data. Convolutional Neural Networks (CNNs). Deep learning methods, notably CNNs, revolutionized neural data analysis by automatically learning spatial and temporal features from minimally processed data. [46] notably demonstrated CNNs' effectiveness in classifying in vitro Multi-Electrode Array (MEA) recordings by genetic phenotype, significantly outperforming traditional logistic regression models. CNNs' inherent ability to learn meaningful patterns directly from raw data positioned them as superior alternatives to traditional feature-engineering-dependent approaches. [40] implemented a CNN-based approach to detect localized pre-burst spike patterns in simulated neuronal networks. In their method, spike data was discretized into 2D matrices (neurons × time), enabling CNNs to learn spatially organized activation patterns that consistently preceded network bursts. The study showed that CNNs could identify specific neuronal sub-populations (akin to ”leader neurons”) that triggered burst events. This work marked a significant step in combining high-throughput simulation data with interpretable deep models for functional network analysis. Recurrent Neural Networks (RNNs) RNNs and Long Short-Term Memory (LSTM) networks complemented CNNs by capturing complex temporal dependencies within spike sequences, proving invaluable in decoding dynamic states and predicting future neuronal activity. Applications of RNNs in brain-machine interfaces underscore their capability to handle sequential and temporally structured neural data. Despite their power, deep models bring new challenges: they demand large labeled datasets, considerable compute, and often lack interpretability in scientific contexts. [46] note that many CNN applications still rely on heavily preprocessed or simulated data to compensate for scarce experimental labels.

2.3 Graph Based Learning and GNNs

Graph theory provided a natural representation of neural systems, with neurons as nodes and synaptic connections as edges. Graph metrics like node degree and clustering coefficient were initially used to characterize network structures. However, meaningful learning on these representations was limited until recent advancements in Graph Neural Networks (GNNs).

2.3.1 Graph Neural Networks (GNNs)

GNNs emerged as a transformative methodology for analyzing graph-structured data by leveraging message-passing mechanisms between nodes that capture both topological and feature-driven information. Applications of GNNs in neuroscience spanned diverse tasks, including brain-state classification, brain parcellation, and connectivity-based disease diagnosis. Wang et al. [42] demonstrated the potential of GNNs at the neuron-level, predicting motor behaviors in C. elegans from neural connectivity graphs. GNNs offer the potential to improve upon traditional techniques by incorporating not only temporal spike features but also topological and relational features from the data, thus enhancing detection accuracy. GNNs allow modeling of both the structural and functional relationships among neurons. Unlike CNNs that rely on spatial adjacency, GNNs can directly leverage the connectivity matrix (e.g., synaptic graph) and spike timing information to learn which subgraphs (sets of neurons and connections) are critical for triggering bursts. This allows for a richer, more biologically plausible interpretation of how local dynamics lead to emergent global behaviors. The ability of GNNs to capture non-linear and distributed connectivity patterns enables better identification of the conditions under which bursts occur, and occur and helps illuminate their functional roles in biological neural networks. These advantages make GNNs an ideal candidate for modeling emergent behaviors like network bursts and for identifying causal substructures within large-scale biological neuronal networks.

Fundamental Concepts and Architectures

Graph Neural Networks (GNNs) learn node representations by iteratively exchanging ("passing") and updating information over the graph structure. At each layer (or iteration) k, every node u has an embedding $\mathbf{h}_u^{(k)}$. To compute the next embedding $\mathbf{h}_u^{(k+1)}$, two steps are performed:

- **Message aggregation:** Each neighbor $v \in \mathcal{N}(u)$ contributes its current embedding $\mathbf{h}_v^{(k)}$, and these are combined via a permutation-invariant function:

  $$\mathbf{m}_u^{(k)} = \text{AGGREGATE}\big\{\mathbf{h}_v^{(k)} : v \in \mathcal{N}(u)\big\} .$$

- **Node update:** The aggregated message is concatenated with the node's previous embedding and passed through a linear transform plus nonlinear activation:

  $$\mathbf{h}_u^{(k+1)} = \sigma\big(W \left[\mathbf{h}_u^{(k)} \,\|\, \mathbf{m}_u^{(k)}\right] + \mathbf{b}\big) ,$$

  where W and $\mathbf{b}$ are learned parameters and σ is a nonlinear activation (e.g., ReLU).

Combining both steps gives:

$$\mathbf{h}_u^{(k+1)} = \sigma\big(W \left[\mathbf{h}_u^{(k)} \,\|\, \text{AGGREGATE}\big\{\mathbf{h}_v^{(k)} : v \in \mathcal{N}(u)\big\}\right] + \mathbf{b}\big) .$$

There are 3 types of tasks that can be performed using GNN as shown in Figure 2.3.

1. Node-level tasks: Classification, assign each node a category (e.g., user interests in a social network or paper topics in a citation graph). Regression, predict a continuous value per node (e.g., traffic volume at intersections).

2. Edge-level tasks: Link Prediction, estimate the likelihood or existence of an edge (e.g. friend or product recommendations). Classification/Regression, determine an edge's type or weight (e.g. relationship category or road-capacity).

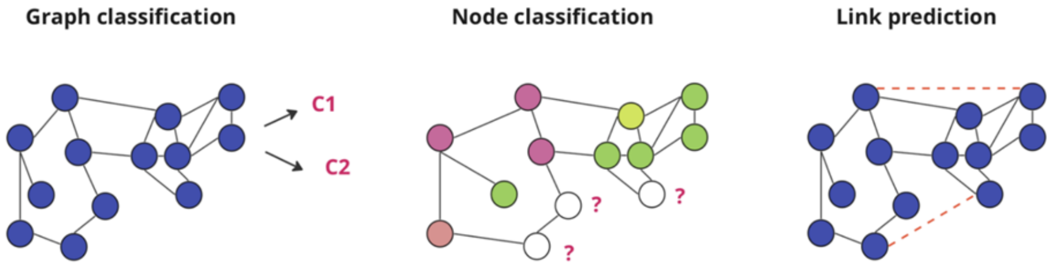


Figure 2.3: Different types of Graph Problems using GNN [6]

3. Graph-level tasks: Classification, label an entire graph (e.g., molecular toxicity). Regression, predict a numeric graph property (e.g., material stability).

Graph Neural Networks (GNNs) perform inference on graph-structured data by learning to map graph inputs to desired outputs. Inferring unknown properties from known graph structure and features is like how a CNN "predicts" that an image contains a cat by recognizing visual patterns. The GNN examines the current graph topology and node/edge attributes to determine properties that aren't explicitly visible in the raw data. For example, given a network of users and friendships, predict each user's interests (node classification); given a molecule's atomic structure, predict if it's water-soluble (graph classification). A typical GNN outputs raw values that are usually probabilities (after softmax) or continuous values. Dimensions depend on the task: Node tasks: N outputs (one per node); Graph tasks: 1 output (for entire graph); Edge tasks: N×N outputs (for all possible edges). Post-processing converts probabilities to class labels or thresholds continuous values.

Graph Neural Networks (GNNs) learn node representations by iteratively exchanging ("passing") and updating information over the graph structure. At each layer l, the node embeddings are stored in the matrix $\mathbf{H}^{(l)} \in \mathbb{R}^{N \times D}$, where N is the number of nodes and D the embedding dimensionality. The input is $\mathbf{H}^{(0)} = \mathbf{X}$, the initial node features. A generic message-passing layer updates embeddings by aggregating neighbor information and applying a transform plus nonlinearity.

Several notable GNN architectures have been proposed, each with distinct message-passing strategies:

Graph Convolutional Networks (GCN): Kipf and Welling [19] introduced GCNs, which simplify message passing via a normalized adjacency-based aggregation. GCNs extend the concept of convolution from images to graphs by aggregating information from a node's local neighborhood. To prevent uncontrolled growth of feature magnitudes, the adjacency matrix is symmetrically normalized. At each layer, a node's representation is updated through a weighted combination of its own features and those of its neighbors, followed by a nonlinear activation such as ReLU. This simple yet powerful formulation effectively captures local structural dependencies while remaining computationally efficient, making GCNs a standard baseline in graph learning. The layer-wise update is:

$$\mathbf{H}^{(l+1)} = \sigma\left(\widetilde{\mathbf{D}}^{-\frac{1}{2}}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right),$$

where $\widetilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$ is the adjacency matrix with self-loops added, $\widetilde{\mathbf{D}}_{ii} = \sum_j \widetilde{\mathbf{A}}_{ij}$ is the corresponding degree matrix, $\mathbf{W}^{(l)}$ is a trainable weight matrix for layer l, and $\sigma(\cdot)$ is a nonlinear activation (e.g., ReLU: $\text{ReLU}(x) = \max(0, x)$). The schematic of a single layer of GCN is showed in Figure 2.4.

Graph Attention Networks (GAT): Veličković et al. [41] introduced Graph Attention Networks, which incorporate an attention mechanism into the message-passing process. Instead of treating all neighbors equally, GAT learns dynamic attention coefficients that assign greater weight to the most relevant neighbors. This allows the model to adaptively prioritize certain nodes or edges, improving expressiveness and interpretability. Importantly, attention weights provide insight into which connections are most influential in the learned embeddings, offering a built-in explainability advantage over GCNs. GAT thus excels in settings where heterogeneity of node importance is critical.

GraphSAGE: Hamilton et al. [13] proposed Graph Sample and Aggregate (GraphSAGE) as a scalable framework designed for very large graphs. Rather than processing the full neighborhood of each node, GraphSAGE samples a fixed-size subset of neighbors and ap-

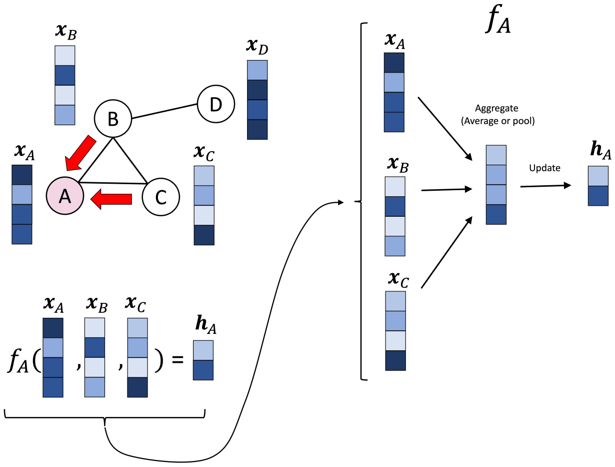


Figure 2.4: Schematic of one graph convolutional layer. The above figure shows working of a single GCN layer where neighbor embeddings are aggregated and fused with the node's previous embedding to produce the updated representation[2]

plies flexible aggregation functions such as mean, max pooling, or even LSTM-based pooling. This inductive formulation allows the model to generalize to unseen nodes or entirely new graphs without retraining, which is especially important for dynamic or growing networks. GraphSAGE's sampling mechanism strikes a balance between scalability and representational power, enabling efficient training on graphs with millions of nodes.

Graph Convolutional Networks (GCNs) were chosen as the primary architecture for this work because they provide a balanced trade-off between interpretability and computational efficiency. Unlike more complex models such as Graph Attention Networks (GAT) or GraphSAGE, GCNs implement a mathematically simple yet powerful message-passing scheme, where node embeddings are updated through symmetrically normalized aggregation of neighbor features. This formulation ensures stability during training and avoids overfitting, while still capturing essential local topological patterns and firing-rate statistics relevant to burst initiation. GCNs are also well-suited for neural network simulations, where the graph structure is relatively dense and global connectivity is important. The spectral foundation of GCNs enables them to exploit the relational structure of the network without requiring extensive parameterization or sampling strategies, making them efficient to train and robust across different simulation configurations.

2.4 Interpretability in GNNs

As CNNs and GNNs grow ever deeper and more complex, their decision processes become effectively "black boxes," making it difficult to trust or validate their predictions especially when these predictions inform scientific hypotheses or clinical interventions. Without insight into why a model flagged a particular neuronal subgraph as a burst origin, we cannot assess whether it is leveraging genuine biophysical signals or spurious correlations. Explainability tools (e.g., GNNExplainer) surface the key nodes, edges, or features driving each prediction, enabling researchers to verify biological plausibility, detect biases, and refine models for safer, more reliable use in neuroscientific and medical settings.

2.4.1 GNN Explainers

GNN-specific explainability methods such as GNNExplainer (Ying et al., 2019) and PGExplainer became prominent. GNNExplainer provides insights into crucial subgraphs (nodes and edges) and features responsible for predictions, revealing minimal neuron groups and connections critical for burst initiation. In the domain of neural burst analysis, applying GNN explainability allows researchers to pinpoint the minimal subgraphs responsible for burst initiation, revealing underlying structural motifs. By integrating GNNs with explainability frameworks, this thesis aims to uncover the functional patterns of spiking neural networks responsible for spontaneous bursts. This approach extends the CNN-based burst trigger identification [40] by incorporating both spatial and topological information, yielding interpretable, graph-level insights into emergent neural behavior. Ultimately, such methods bridge the gap between black-box learning and biological plausibility, enhancing our understanding of neural growth and development.

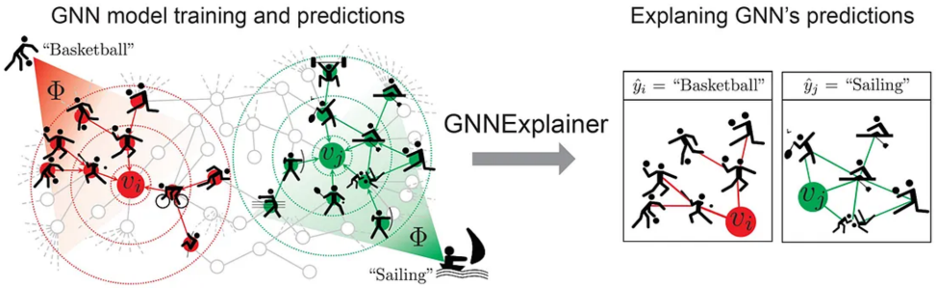

GNN model training and predictions
Explaning GNN's predictions
“Basketball”
Φ
v_i
v_j
Φ
“Sailing”
GNNExplainer
$\hat{y}_i$ = “Basketball”
$\hat{y}_j$ = “Sailing”
v_i
v_j

Figure 2.5: GNNExplainer identifies a compact explanatory subgraph. Hypothetical node classification task: a GNN model Φ trained on a social interaction graph predicts future sport activities. For node v_i with prediction $\hat{y}_i$ = “Basketball”, GNNExplainer identifies a compact explanatory subgraph and relevant node features showing that many friends in one community enjoy ball games. For node v_j with prediction $\hat{y}_j$ = “Sailing”, the explanation highlights friends and second-order neighbors who favor water and beach sports[44]

Chapter 3
METHODS

3.1 Data Acquisition

All simulated data in this study come from the Graphitti simulator, the GPU-enabled successor to BrainGrid [16] Graphitti implements a leaky-integrate-and-fire (LIF) neuron model and a synapse model that includes synaptic facilitation and resource depletion that captures realistic spike timing while remaining computationally tractable. This section details how the growth simulation is configured, and which artefacts are extracted for downstream graph-based analysis.

3.1.1 Network Initialization

The initialization process involves defining the network layout, using XML file including the number of neurons, synaptic connections, and other parameters for the simulation. To minimize spatial bias, a 10 × 10 tiling pattern was introduced by (Kawasaki & stiber), which produces a regular arrangement of the three different neuron types, Inhibitory, Excitatory and Endogenously Active. Repeating this tile 10 × 10 times yields a 100 × 100 culture of 10 000 neurons composed of excitatory and inhibitory neurons. A GraphML parameter file records this layout so the simulation can be reproduced exactly.

Growth Simulation

The growth phase replicates 28 days in vitro (DIV) of neurite outgrowth and synapse formation:

- **Temporal resolution:** 0.1 ms

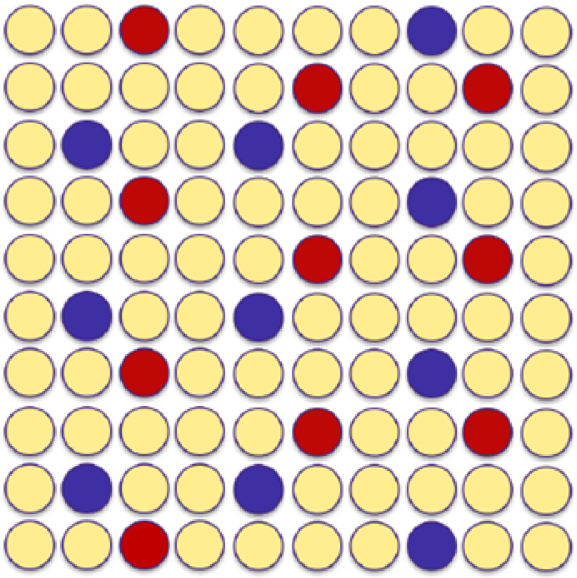


Figure 3.1: Layout for different neurons. It includes endogenously active neurons (blue dots), inhibitory neurons (red dots) and excitatory neurons (yellow dots) [16].

- **Epoch:** 100 seconds of biological time

- **Total epochs:** 600, corresponding to 60,000 seconds

The growth simulation uses a time step of 0.1 milliseconds and consists of 600 epochs. This phase is crucial for understanding the emergence of burst activity and the evolving behavior of the simulated neural network.

During each epoch, membrane potentials are integrated and spikes are logged at every 0.1 ms (i.e., one million timesteps per 100-second epoch). Synaptic topology is only modified at the end of each epoch in a *growth-update* step. In that step, synapses form when the growth radii of two neurons overlap, creating bidirectional connections whose weights are proportional to the overlap area. The result is an undirected, weighted connectivity graph that gradually densifies over the simulated 28-day period, eventually approaching a mature stable structure.

Following [16], this study uses a target firing rate $\varepsilon = 1.0\,\text{Hz}$ with 98% excitatory cells, a configuration known to produce robust network bursting in silico.

Recorded Data Structures

After the final (600th) growth epoch, Graphitti outputs two key artifacts:

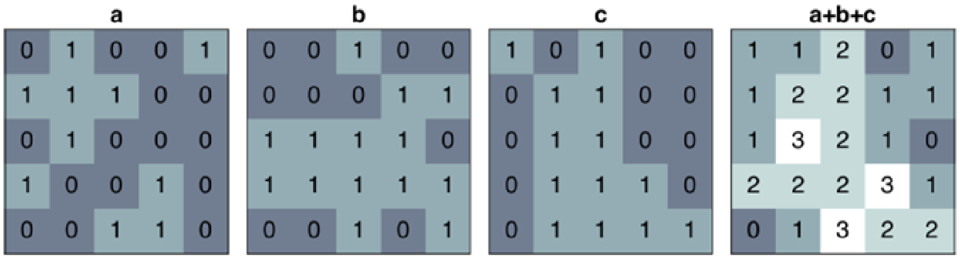
Table 3.1: Output data produced by Graphitti after the growth simulation.

Variable	**Shape**	**Description**
HDF5 spike file	Varies per neuron	One dataset per spiking neuron listing its spike times as time-step indices (i.e., the discrete times when that neuron fired).
Weights matrix	10,000 $\times$ maxEdges	Synaptic weights for each incoming edge; stored alongside `sourceIndex` and `destinationIndex` to define directed connectivity.

The Weights matrices form a sparse adjacency list that we append to the original GraphML file which has a list of neuron IDs and their (x,y) locations, producing a fully specified graph ready for GNN processing.

3.2 Previous Data Analysis

Throughout the 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. To understand burst initiation patterns within neuronal networks, various computational methods have previously been employed to analyze spike train data. Given the complexity and scale of neuronal data, reliable and interpretable identification methods are essential. This section reviews prior



a

0	1	0	0	1
1	1	1	0	0
0	1	0	0	0
1	0	0	1	0
0	0	1	1	0

b

0	0	1	0	0
0	0	0	1	1
1	1	1	1	0
1	1	1	1	1
0	0	1	0	1

c

1	0	1	0	0
0	1	1	0	0
0	1	1	0	0
0	1	1	1	0
0	1	1	1	1

a+b+c

1	1	2	0	1
1	2	2	1	1
1	3	2	1	0
2	2	2	3	1
0	1	3	2	2

Figure 3.2: Example of spike train binning (bin size = 3) . Image a, b and c represent three consecutive time steps in a 5 X 5 arrangement. The binary values in each location represents a single spike train value. The rightmost image shows the result of binned spike trains of a,b, and c. The color in each square corresponds to the spike count or spike rate of the neuron at that (x; y) location. The brighter colors indicate higher spike rates.[22]

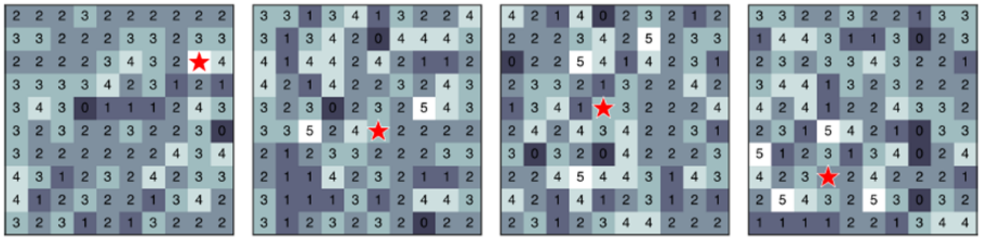
analytical approaches used for burst detection, characterization, and identification of origins from simulations, highlighting methodologies that guided our research.

3.2.1 Identification and Characterization of Bursts

A reliable method involved dividing the network spike data into short temporal bins (e.g., 10 ms bins, equivalent to 100 simulation time steps at 0.1 ms per step). Bursts were identified by applying thresholds to spike counts within these bins. As previously demonstrated in [23] and [40], bursts manifest as significantly elevated spike counts across the entire network in short time intervals. Consistent with physiological findings, a threshold of 50 spikes per bin (equivalent to 0.5 spikes/sec/neuron) effectively discriminated between normal activity and burst periods. These thresholds have been extensively validated and applied across multiple studies, providing a robust standard for identifying burst intervals.

3.2.2 Identifying Burst Origins

Burst initiation locations have attracted significant attention in previous analyses. Visualization methods revealed that bursts consistently originated from distinct localized regions, subsequently propagating across the network ([23], [40]). Accurately pinpointing these ini-



2	2	2	3	2	2	2	2	2	2
3	3	2	2	2	3	3	2	3	3
2	2	2	2	3	4	3	2	★	4
3	3	3	3	4	2	3	1	2	1
3	4	3	0	1	1	1	2	4	3
3	2	3	2	2	3	2	2	3	0
3	2	2	2	2	2	2	4	3	4
4	3	1	2	3	2	4	2	3	3
4	1	2	3	2	2	1	3	4	2
3	2	3	1	2	1	3	2	2	2

3	3	1	3	4	1	3	2	2	4
3	1	3	4	2	0	4	4	4	3
4	1	4	4	2	4	2	1	1	2
4	2	1	4	2	2	3	2	4	3
3	2	3	0	2	3	2	5	4	3
3	3	5	2	4	★	2	2	2	2
2	1	2	3	3	2	2	2	3	3
2	1	1	4	2	3	2	1	1	2
3	1	1	1	3	1	2	4	4	3
3	1	2	3	2	3	2	0	2	2

4	2	1	4	0	2	3	2	1	2
2	2	2	3	4	2	5	2	3	3
0	2	2	5	4	1	4	2	3	1
2	3	3	2	1	3	2	2	4	2
1	3	4	1	★	3	2	2	2	4
2	4	2	4	3	4	2	2	3	1
3	0	3	2	0	4	2	2	2	3
2	2	4	5	4	4	3	1	4	3
4	2	1	4	1	2	3	1	2	1
2	3	1	2	3	4	4	2	2	2

3	3	2	2	3	2	2	1	3	3
1	4	4	3	1	1	3	0	2	3
2	3	2	3	3	4	3	2	2	1
3	4	4	1	3	2	3	2	2	2
4	2	4	1	2	2	4	3	3	2
2	3	1	5	4	2	1	0	3	3
5	1	2	3	1	3	4	0	2	4
4	2	3	★	2	4	2	2	2	1
2	5	4	3	2	5	3	0	3	2
1	1	1	1	2	2	1	3	4	4

Figure 3.3: Examples of burst origin. It is determined by computing the centroid of the brightest pixels. For each image, numbers in each square represent pixel values and red pentagram marks the centroid of the brightest pixels (highest pixel value = 5). Images from left to right show the centroids of 1, 2, 3, and 4 brightest pixels, respectively. Taken from [23]

tiation sites is crucial for understanding underlying burst mechanisms and their biological implications.

[22] introduced a systematic approach brightest pixel selection to determine burst origin locations. Specifically, the centroid of the neuron(s) displaying the highest spike count within the initial time bin (first 10 ms) was computed, indicating the burst initiation site. To ensure accuracy, a threshold spike count of at least two spikes per neuron per bin was enforced, addressing potential artifacts due to binning effects [22]. This approach consistently pinpoints tight clusters of neurons as burst initiation sites, and the centroid of each cluster designates the single “burst origin” neuron.

In this thesis, we aim to apply a graph based deep learning approach to understand neural network dynamics preceding burst events by analyzing subgraphs centered around each burst origin detailed in the next section.

Following [16], we inspect the connectivity-radius history (Figure 3.6). We can see that connectivity stabilized after 2̃0 DIV. Since we have connectivity data recorded at the end of the simulation’s last epoch, bursts are only considered from the last quarter of simulation time, resulting in approximately 2600 burst events [16].

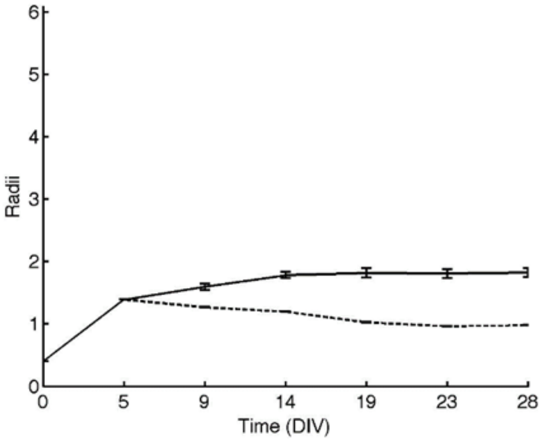


Figure 3.4: Evolution of neuronal cluster radii over time in vitro. It's seen that radii stabilized after $\tilde{2}0$ DIV.[16]

3.3 Data Pre-Processing

3.3.1 Burst Origin Subgraph Construction

To reduce the complexity and redundancy of using the entire 10,000 neuron network to train GNN models and to capture detailed local connectivity patterns surrounding only the burst origins, we decided to construct what we called burst origin subgraphs. Each burst origin subgraph is represented by:

- Nodes: Individual selected neurons around the respective origin.
- Edges: Synaptic connections between those neurons, defined using the known weight Matrix (source neuron index, destination neuron index, and synaptic weight).

Based on the connectivity analysis showing that synaptic connections typically extend 2-3 micrometers (corresponding to the neurite growth radius at network maturity), we determined that a 2-hop graph neighborhood would capture the most relevant local interactions. It is important to note that "2-hop" refers to graph distance (the number of synaptic connections), not spatial distance. As shown in Figure 3.5 a 2-hop neighborhood includes:

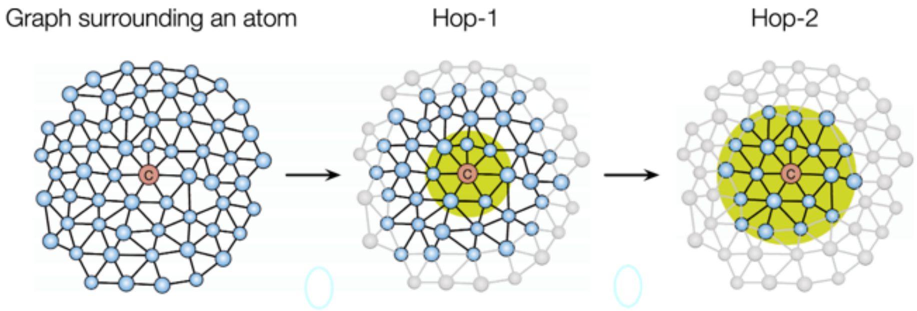


Figure 3.5: Illustration of expanding node neighborhood around a central neuron (C). The leftmost panel shows the full graph with all neighbors; the middle panel highlights the 1-hop neighborhood (directly connected nodes); and the rightmost panel shows the 2-hop neighborhood, capturing both direct and indirect interactions. Ignore the lighted gray parts of second and third graphs.[43]

- The origin neuron (0 hops)
- All neurons directly connected to the origin (1 hop)
- All neurons connected to those first neighbors (2 hops)

This graph-based approach captures neurons that are synaptically close, even if they are spatially distant.

Figure 3.6 'Left plot shows a Full 100×100 neural network layout showing the spatial distribution of all 10,000 neurons (blue: excitatory, red: inhibitory). We have chosen as an example a burst with the location of origin neuron 4637 (red star) and the corresponding 2-hop neighborhood extraction region. The right plot shows a detailed spatial view of the extracted 2-hop subgraph containing 111 nodes (108 excitatory circles, 2 inhibitory squares) and 1470 synaptic connections (gray lines) arranged in their actual culture coordinates.

We implement this using NetworkX's ego_graph(G, center_node, radius=2) function. As shown in Figure 3.6, this approach extracts irregular, non-circular subgraphs that follow the actual synaptic connectivity rather than imposing an artificial spatial boundary. This approach reduces the computational complexity from analyzing the entire 10,000-neuron

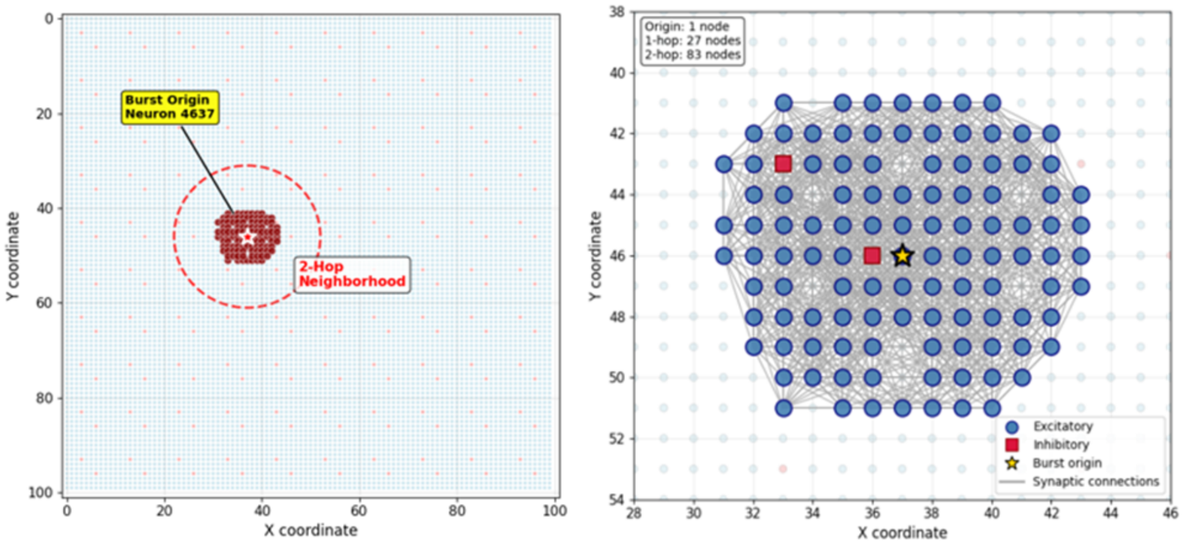


Figure 3.6: 2-hop neighborhood graph. Example of 2-hop burst origin subgraph construction using NetworkX ego_graph().

network while preserving the local connectivity patterns essential for understanding burst initiation dynamics. The dense interconnectivity visible in the spatial layout demonstrates the rich synaptic organization surrounding burst origins, providing the foundational data structure for subsequent GNN analysis.

3.3.2 Feature Extraction

After generating the burst origin subgraphs, we prepare these graphs for a binary graph classification task to classify bursts based on local connectivity and spike-timing features into 2 categories: Pre-burst and Non-burst.

Spike data for neurons were retrieved from the simulator's HDF5 spike file. Features were computed from two distinct time windows for each burst origin subgraph depicted in Figure 3.9:

· Pre-burst window: Immediately preceding burst onset.

· Non-burst window: Baseline period, temporally distant from any burst event. From

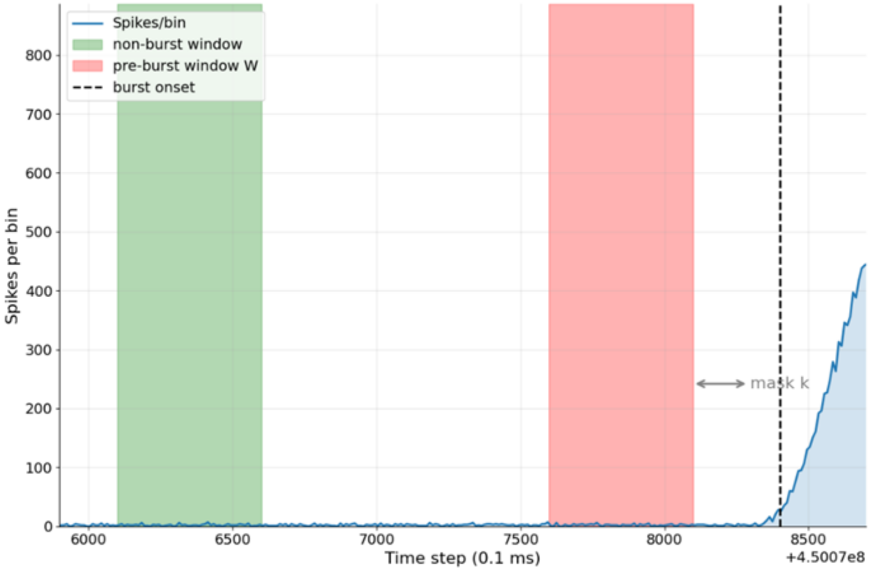


Figure 3.7: Spike-count time series for a single burst event with pre-burst and non-burst windows. The green shaded region marks the non-burst window (*'w'* timesteps) positioned *'gap'* timesteps before the red pre-burst window (*'w'* timesteps), which precedes the burst onset (black dashed line) by a mask interval *'k'*.

these spike windows, each neuron's activity is summarized into four quantitative node features widely recognized in computational neuroscience:

1. Mean Inter-spike Interval (ISI) *(ms)*: The average time between successive spikes.
2. Entropy of ISI *(bits)*: Shannon entropy of ISI, indicating spike variability.
3. Last Lag *(ms)*: Time from the neuron's last spike to the end of the analysis window.
4. Firing Rate *(Hz)*: Defined as the reciprocal of the mean inter-spike interval (ISI), firing rate quantifies how frequently a neuron fires over time.

These features collectively capture nuanced aspects of neuronal activity, including average firing rate, temporal variability (ISI entropy), recent spike timing (last lag), and rhythmicity (mean ISI). By quantifying both rate-based and temporal dynamics of each neuron's firing pattern, they provide a comprehensive encoding of neural behavior. Each neuron's feature vector comprising these temporal statistics is embedded into its corresponding node within the subgraph depicted by Figure 3.8. This results in a feature-rich graph representation

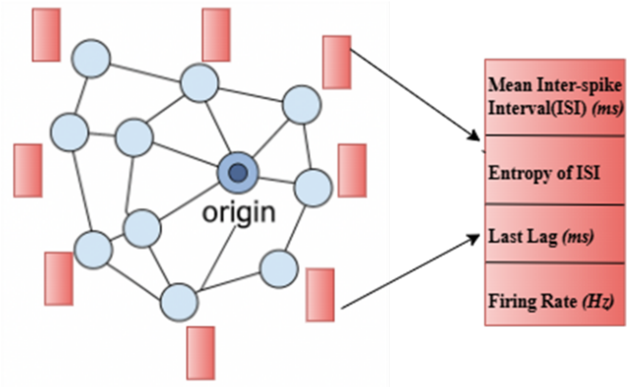


Figure 3.8: Illustration of a burst-origin-centered subgraph with node-wise temporal features. Each neuron is enriched with a 4-dimensional feature vector capturing mean inter-spike interval (ISI), ISI entropy, last lag, and firing rate, enabling the GNN to jointly model spatial and temporal dynamics.

where both structural connectivity and node-level activity patterns jointly inform the GNN model's learning process.

Two feature-enhanced subgraphs are thus generated per burst origin one capturing the neuronal dynamics immediately before the burst (label = 1, positive class), and one from a baseline non-burst period (label = 0, negative class). This method effectively doubles the dataset size from approximately 2600 to around 5200 subgraphs, while simultaneously ensuring class balance and enabling direct comparative analysis of neuronal behaviors preceding burst initiation versus baseline activity.

3.3.3 Scaling and cleaning

Before training the Graph Neural Network (GNN) model, all computed node features undergo normalization to ensure comparable scales and numerical stability. Specifically, we apply z-score normalization to each feature independently.

Additionally, the graph adjacency matrix normalization follows the method proposed by [19], using symmetric normalization which effectively balances the influence of neighboring nodes during convolutional aggregation, improving the stability and performance of GNN training [19].

In the next section, we describe the design and training of our Graph Neural Network (GNN) model for predicting burst vs. non-burst subgraphs. We cover the architectural choices, input/output formulation, training protocol (including train-validation-test splits with group shuffling), and baseline comparisons.

3.4 Graph neural Network and its Training

3.4.1 Model Architecture

For simplicity and clear interpretation, we employ a 3-layer Graph Convolutional Network (GCN) architecture introduced by [19] for our binary graph-classification task (pre-burst vs. non-burst). This model is built using the PyTorch Geometric library. As depicted in Figure 3.11, the proposed model architecture comprises three Graph Convolutional Network layers (GCNConv), each employing symmetric normalization of the adjacency matrix as described by [19]. A ReLU activation function is applied after the first two convolutional layers to introduce non-linearity. Following the final GCNConv layer, a global mean pooling layer aggregates all node-level representations into a fixed-size graph-level embedding, t-SNE plot of the embeddings shown in Figure 3.12. To prevent overfitting, a dropout rate of 50% is applied to the pooled embedding before passing it to the final linear classifier.

This two-layer setup allows the network to aggregate information from nodes' 2-hop neighborhoods. At each layer, node features (e.g., firing-rate statistics) are mixed with neighboring nodes' representations, enabling our model to learn both local spike-pattern cues and surrounding network structure.

3.4.2 Input / Output

The GCN is designed for graph-level binary classification, assigning labels to each burst origin graph element as either a pre-burst (**1**) or non-burst (**0**) graph.

The Input graph is represented as a PyG *data* object [6] which is a fundamental data

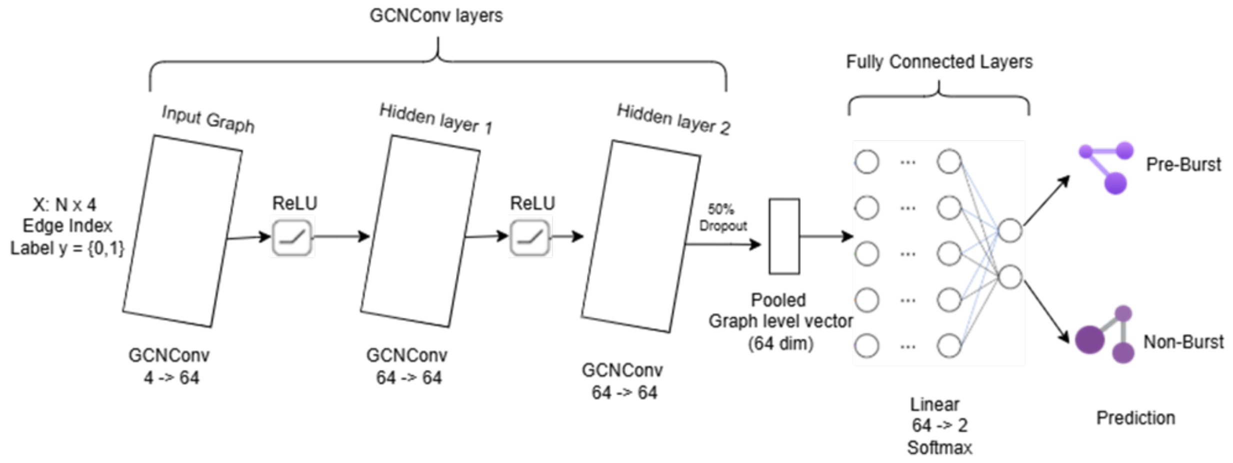


Figure 3.9: Detailed architecture of the GCN-based classification model. The input is represented as a PyTorch Geometric *Data* object containing node features $X \in \mathbb{R}^{N\times 4}$, edge indices (graph connectivity), and a binary label indicating burst class (0: Non-burst, 1: Pre-burst). The model consists of three GCNConv layers with symmetric normalization (Kipf & Welling), each with a hidden dimension of 64. ReLU activations follow the first two convolutional layers, and a 50% dropout is applied after the final convolution. A global mean pooling layer aggregates node-level features into a fixed-size graph embedding, which is then passed through a fully connected linear layer (64 → 2) followed by a softmax classifier to predict whether the input graph corresponds to a pre-burst or non-burst neural pattern

structure used to represent a single graph and all its associated properties (e.g., node features, edge connections, labels) in a compact and efficient way. It contains X (node-feature matrix of shape [N, 4], where N is the number of nodes in the subgraph and 4 = {mean_isi, entropy, last_lag, rate}), edge index (sparse representation of the functional subgraph's edges) and a y label (1 for a pre-burst window, 0 for a non-burst window).

The Output is a two-class prediction per graph indicating whether that subgraph occurred immediately before a burst (1) or during non-burst period (0).

This formulation makes our task a graph-level classification problem, where each input subgraph representing the local structure around a burst origin is mapped to a fixed-length embedding using global mean pooling. These embeddings are learned through stacked GCN layers and encapsulate both node feature information and structural connectivity. To qualitatively assess how well the learned embeddings separate the two classes (pre-burst and non-burst), we project them into two dimensions using t-SNE [3]. As shown in Figure 3.10, the embeddings exhibit clear clustering behavior, with Class 0 (non-burst) and Class 1 (pre-burst) forming distinguishable, though overlapping, regions in the embedded space. This separation suggests that the model can capture meaningful latent representations that aid in classification. The embeddings for all the graphs are collected and visualized in a t-SNE plot in Figure 3.10.

3.4.3 Data Splitting and Model Training

Since each burst origin is represented by two graphs (pre-burst and non-burst), grouping ensures that all graphs from the same burst data sample either appear wholly in the train, validation, or test sets. This avoids having correlated examples (from the same burst) in both train and evaluation sets, which could artificially inflate performance.

To prevent this information leakage between sets, we split the data using the burst identifier (`burst_id`) rather than individual graphs. For this, we use the `GroupShuffleSplit` function from Scikit-learn [31]. The splitting is performed in two stages:

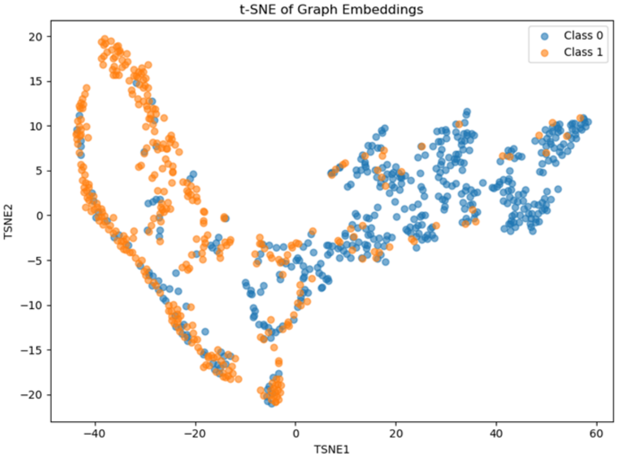


Figure 3.10: t-SNE visualization of graph-level embeddings produced by the GCN model. Each point represents a pooled embedding of a subgraph, colored by its ground-truth label (class 0: non-burst, class 1: pre-burst). The clear separation indicates effective class discrimination in the learned embedding space.

1. **Train-Validation/Test split:** 15% of unique `burst_ids` are held out for testing.

2. **Train/Validation split:** 15% of the remaining `burst_ids` are held out for validation.

We train our Graph Neural Network (GNN) model using the cross-entropy loss function, a standard choice for binary classification tasks. Optimization is performed using the Adam optimizer [18], with a learning rate of $l \times 10^{-2}$.

To ensure efficient convergence and dynamically adapt the learning rate, we employ a `ReduceLROnPlateau` scheduler. This scheduler monitors the validation loss and halves the learning rate if no improvement is observed for four consecutive epochs. This strategy enables the model to fine-tune weights during later training stages without overshooting the minima. Such an approach is particularly useful in graph learning, where model performance may plateau due to overfitting or local minima [27].

To further mitigate overfitting, we adopt an early stopping strategy: training is terminated if the validation loss does not improve for ten successive epochs, with a maximum training cap of 100 epochs.

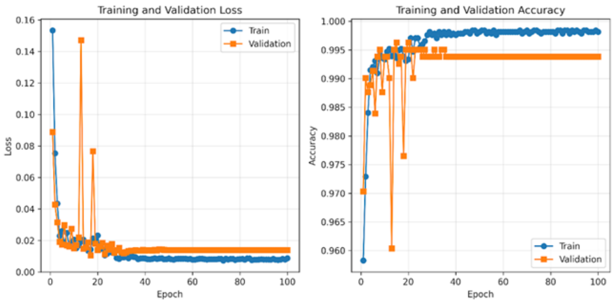


Figure 3.11: Training and validation curves of the GCN model. It shows loss over epochs and accuracy comparison. The model achieves early convergence, high accuracy, and stable generalization, as indicated by the close alignment of training and validation metrics.

Training is carried out using PyTorch Geometric's `DataLoader`, where graphs are batched to facilitate mini-batch stochastic gradient descent. We use a batch size of 32, with `shuffle=True` during training to ensure varied mini-batches across epochs, and `shuffle=False` for validation and testing to maintain consistency.

During each epoch, we record the average training loss and training accuracy. On the validation set, we compute a suite of metrics including loss, accuracy, precision, recall, and F_1-score, offering a comprehensive view of model performance beyond simple accuracy. However, since our dataset is balanced, we report the F_1-score for comparison with prior work.

We also performed extensive model training with varying window, gap, and mask sizes, shown in the results section. Results show that window size of 5 and Mask size of 0 performs the best. Hence, we have chosen that combination for our further analysis.

Figure 3.11 illustrates the training and validation loss curves over epochs. The loss curve (left) shows rapid convergence within the first 20 epochs, with training and validation loss stabilizing afterward, indicating good generalization and the effectiveness of early stopping. The accuracy curve (middle) demonstrates near-perfect classification performance after the initial phase, with validation accuracy plateauing slightly below training accuracy, suggesting minor overfitting. The F1-score curve (right) mirrors this trend, achieving consistently high scores above 0.99, highlighting the model's strong ability to balance precision and recall across both classes.

3.4.4 Ablation Study and Baseline Model Comparison: Validating the Role of Graph Topology

The ablation study was employed to systematically check the contributions of different components of our model and data representation. Without such validation, it remains possible that accuracy is driven by spurious correlations, node-level firing features, or dataset-specific artifacts. By selectively removing or altering components—such as graph connectivity, feature sets, or message-passing mechanisms—an ablation study allows us to test whether performance declines in a manner consistent with our hypotheses.

This methodology serves two purposes: (1) it verifies that graph topology and message-passing are essential for the model's success, rather than incidental; and (2) it provides a principled way to compare against simpler baselines, ensuring that the added complexity of GCNs is justified. In doing so, the ablation study strengthens the scientific validity of our findings by demonstrating that the discovered patterns reflect meaningful network structure, not accidental statistical cues. The results are displayed in Table 3.2.

Variant 1: Label Shuffle Test

This test guards against the most fundamental form of data leakage—information inadvertently encoded in features that directly reveals the label. By randomly permuting labels while keeping all other data unchanged, we test whether the model can still achieve above-chance performance. We randomly shuffled the binary labels (pre-burst vs. non-burst) across all training samples while maintaining the original feature-graph pairs.

Result: Accuracy dropped to 48.9%. This confirms that no label leakage exists in our pipeline. The model cannot predict shuffled labels, validating that our features do not inadvertently encode the target variable.

Variant 2: Node ID Shuffle Test

GNNs can potentially memorize specific node positions rather than learning generalizable connectivity patterns. For instance, if burst origins always occur at node index 0, the model might simply learn “node 0 predicts positive class” rather than understanding topological patterns.

To test this, we randomly permuted node indices within each graph while preserving the graph structure (i.e., edges are remapped to maintain connectivity).

Result: Performance remained at 99.6%, identical to the baseline. This demonstrates that our model genuinely learns from connectivity patterns and features—not from memorizing specific neuron IDs or positions. The model’s invariance to node permutation confirms proper graph-based learning.

Variant 3: Constant Features Test

This ablation isolates the contribution of topology by removing all node feature information. If the model maintains high performance with constant features, it suggests that graph structure alone drives predictions. All node features were replaced with a constant value (1.0), eliminating any temporal dynamics information.

Result: Accuracy collapsed to 50.0%. Node features are essential for burst classification. This confirms that topology alone, without temporal spike information, cannot distinguish pre-burst from non-burst states.

Variant 4: No Edges Test

This ablation establishes the upper bound of what features alone can achieve when processed through the same GCN architecture but without any connectivity information. All edges were removed, leaving isolated nodes that pass through the GCN without message passing.

Result: Performance dropped to 95.7% (a 3.9% decrease). This represents the ceiling for feature-only prediction using our architecture. The 3.9% gap between this and full GCN performance quantifies the value added by graph convolutions and message passing.

Variant 5: MLP Baseline Test

This variant provides a fair comparison against a traditional machine learning approach that cannot leverage graph structure. The MLP has comparable capacity but lacks inductive bias for processing graphs. Node features were mean-pooled to create graph-level features, then classified using a 3-layer MLP with hidden dimensions matching the GCN.

Result: Achieved 97.6% accuracy (2.0% below GCN). This indicates that features are highly discriminative, but topology still offers performance gains.

Table 3.2: Ablation study and baseline model results.

Variant	**F_1 Score (± SD)**
Label shuffle	0.657 ± 0.000
Node ID shuffle	0.996 ± 0.002
Constant features	0.667 ± 0.000
No edges	0.955 ± 0.012
MLP baseline	0.976 ± 0.004
GCN (full model)	0.996 ± 0.001

Our ablation study reveals a nuanced picture:

1. **Features are powerful:** Our temporal features (ISI statistics, firing rate) are highly discriminative, enabling even simple models to achieve >95% accuracy.

2. **Topology provides crucial refinement:** Graph structure consistently improves performance across all tests.

3. **The model learns correctly:** Lack of sensitivity to node permutation and failure on

shuffled labels confirms the model learns meaningful patterns rather than exploiting artifacts.

The 2–3% performance gains from incorporating topology may appear modest but represent significant improvements in the high-accuracy regime. By extracting the connectivity information after every simulation epoch and during the exact burst timing—which will be explored in future work—we could further improve model performance.

These results validate our hypothesis that local connectivity patterns contain information complementary to temporal features for predicting neural burst events, justifying the use of graph neural networks for this task.

3.5 Interpretability for Burst Initiation Pattern

Understanding not just what a model predicts, but why, is essential where trust, interpretability, and mechanistic insight are critical. In this chapter, we present our work using Interpretability to understand the burst initiation patterns captured by our Graph Neural Network (GNN) model. This aids in identifying neuron behaviors that contributed significantly to a subgraph being classified as pre-burst.

Several interpretability approaches for GNNs were considered in this work, each offering distinct strengths and trade-offs. The primary method adopted was GNNExplainer [44], which learns a compact subgraph and node-feature mask that maximizes mutual information with the model's prediction. This framework is both model-agnostic and non-parametric, directly producing edge and feature importance scores. Its ability to yield instance-specific explanations makes it particularly well suited for our problem, where the goal is to uncover minimal connectivity patterns that drive burst initiation rather than providing only global feature rankings.

We also considered PGExplainer[29], a parameterized technique that trains an auxiliary neural model to generate edge masks across multiple instances. While promising for generalization across graphs, PGExplainer requires additional training overhead and careful hyperparameter tuning, which makes it less practical in our setting where the focus is on interpreting individual burst initiation events. Similarly, GraphLIME[14] was explored as a surrogate-based method that fits LIME-style linear models to local graph neighborhoods. Although effective for producing feature-level interpretability, its reliance on linear approximations limits its ability to capture the complex, nonlinear interactions characteristic of cortical network dynamics.

For these reasons, GNNExplainer was selected as the primary explanation technique. Its direct compatibility with GCN outputs, ability to identify instance-level causal subgraphs, and proven success made it the most suitable tool for extracting biologically meaningful insights from our models. Each burst event receives a tailored explanation, enabling pattern discovery across individual burst instances rather than population-averaged explanations. GNNExplainer also avoids the need for retraining or surrogate models, making the pipeline simple and interpretable. Prior work demonstrates that GNNExplainer outperforms other explainers while remaining computationally efficient [44].

3.5.1 GNNExplainer Integration and Evaluation

We employed GNNExplainer [44] as the primary interpretability framework to identify minimal subgraphs and node features that preserve the GNN's predictions. The explainer optimizes learnable masks over edges and node features, balancing fidelity to the original model with interpretability. In our setup, the input included the node feature matrix $x \in \mathbb{R}^{N\times 4}$ (encoding mean inter-spike interval, entropy, last lag, and firing rate), the graph structure in COO format, batch tensors for pooling, and the target label (0: non-burst, 1: pre-burst). The outputs consisted of a node mask, which highlights the importance of the four spike features, and an edge mask, which assigns probabilities to edges after sigmoid transformation. Each explanation was generated at the graph-instance level, enabling analysis of individual pre-burst events while also facilitating discovery of consistent motifs across multiple instances.

To quantitatively assess explanation quality, we adopted two standard metrics: **fidelity**, which measures how well the GNN's output is preserved when retaining only the most important nodes and edges, and **sparsity**, which measures the conciseness of the explanation in terms of the fraction of edges retained. A fidelity-to-sparsity trade-off score was used to capture the balance between predictive faithfulness and interpretability. We performed a hyperparameter sweep varying the number of epochs (20–200), learning rates (0.001–0.05), and mask thresholds (0.01–0.2).

The results were consistent across settings: edge fidelity averaged 0.461 and node fidelity 0.502, while edge sparsity ranged from 0.033 to 0.699, reflecting sensitivity to the mask threshold. Explanations also showed strong consistency across runs, with an average Jaccard similarity of 0.73 ± 0.12, confirming that the explainer reliably identified recurring precursor subgraphs.

Despite achieving excellent model accuracy (99.6%), the fidelity scores were consistently moderate (0.45–0.50). This observation indicates that no single subset of nodes or edges—or minimal critical subgraph—preserves the model's prediction. Rather than being a limitation of GNNExplainer, these moderate scores provide important evidence that burst classification is inherently distributed across the network.

If prediction relied on specific critical connections, we would observe high fidelity when those edges are retained. Instead, the consistent ∼50% fidelity suggests that roughly half of the network information is needed for accurate prediction, indicating a distributed mechanism rather than a localized one.

This aligns with neuroscientific principles: network-level behaviors such as burst initiation often emerge from population dynamics rather than isolated neuronal circuits. These results validate our hypothesis that local connectivity patterns complement temporal features for classifying neural burst events, justifying the use of graph neural networks for this task.

Hence, the moderate fidelity scores—initially concerning—ultimately provided the first evidence for our key biological insight: **burst initiation cannot be localized to specific critical connections because it emerges from network-wide pattern integration**.

This realization fundamentally shaped our subsequent analytical approach. We imple-

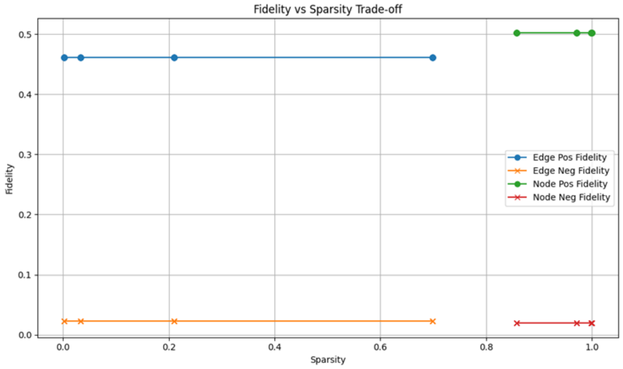


Figure 3.12: Fidelity versus sparsity trade-off for node and edge explanations. **Fidelity+** (Edge/Node Pos) evaluates how much prediction accuracy drops when key components are removed, while **Fidelity** (Edge/Node Neg) evaluates how accurately the model predicts using only the explanation subgraph. High Edge and Node Pos fidelity at varying sparsity levels confirms the model's reliance on compact, informative substructures for decision making. The low Fidelity values indicate that the masked subgraphs alone may be insufficient for faithful standalone predictions.

mented a full explanation pipeline over the entire burst dataset, generating explanations for each graph instance using GNNExplainer with tuned hyperparameters. For every subgraph instance, the resulting explanations were saved for downstream spatial and statistical analysis.

GNNExplainer Saliency Interpretation: The output from GNNExplainer includes saliency scores for both nodes and edges. These scores represent *relative importance* within each graph instance—indicating how crucial each node or edge was to that specific prediction—not absolute importance across all graphs. This distinction is essential: the saliency scale is instance-specific and continuous in the range $[0, 1]$ (after sigmoid transformation). Therefore, applying a global threshold (e.g., saliency > 0.5) across instances is unreliable due to variation in score distribution.

To address this, we adopted a **ranking-based approach** over absolute thresholding. Instead of using a fixed cutoff like "saliency > 0.5," we selected the top 20% of nodes and edges with the highest saliency values in each graph. This method ensures that:

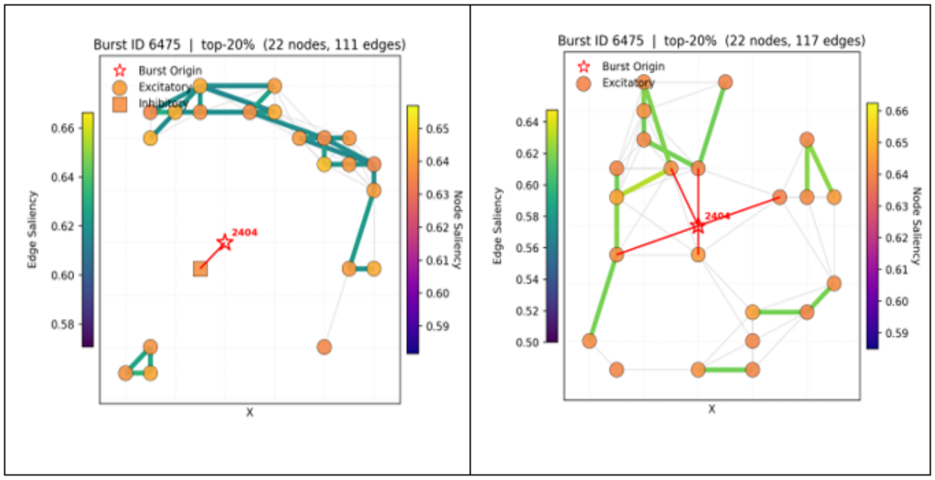


Figure 3.13: Top 20% salient nodes visualized in its spatial layout. Red star denotes the Burst Origin while left shows Pre-burst graph and right Non-burst graph

- Each explanation focuses on the most salient parts of the graph.
- Explanation sizes remain consistent across instances.
- It is robust to scale variation and avoids hard thresholding artifacts.

This approach aligns with best practices in explainable AI for graphs and is analogous to saliency visualization in image models, where the top pixels are highlighted instead of using a hard intensity threshold [44].

We then plotted the spatial layout of each graph using its original 2D neuron coordinates (x, y), overlaying the top 20% most important nodes and edges. This provided a biologically interpretable visual explanation of how the model predicted burst vs. non-burst for a given subgraph.

Figure 3.13 shows such an explanation for **Burst ID 6475** in both its pre-burst and non-burst states, highlighting the spatially distributed nature of model decision-making.

When analyzed all the pre-burst explanation figures as a movie, we observed 2 patterns being repeated in most of the graphs. The subsequent analysis will discuss more about that.

3.6 Two Pattern Discovery in Pre-Burst Graphs

3.6.1 Spatiotemporal Pattern Discovery in Pre-Burst Graphs

Through systematic analysis of GNNExplainer outputs across 2,689 burst events, we observed that the GNN model predicts burst initiation based on distributed neural circuit patterns, rather than localized activity solely at the burst origin neuron.

As an initial step toward motif discovery, two prominent patterns emerged consistently. In the following section, we outline our methodology for identifying and characterizing these recurring spatiotemporal motifs in pre-burst activity graphs, which we refer to as:

1. **Pattern A (Local Hub):** A small, densely connected cluster of salient neurons centered near the burst origin(Figure 3.14).

2. **Pattern B (Remote Ring):** A spatially dispersed ring-like structure of salient neurons positioned away from the burst origin (Figure 3.15).

Graphs that did not exhibit strong evidence of either motif were conservatively labeled as **Pattern C (Unclassified)**. This unsupervised classification was driven by both quantitative graph metrics and visual inspection of salient node distributions.

To systematically classify the spatial saliency patterns observed in our visual analysis, we developed a set of quantitative decision rules based on spatial distribution and saliency characteristics of each graph's top salient nodes shown in Algorithm 1.

Classification Criteria

This categorization resulted in Pattern A (Local Hub) which has 686 graphs (25.5%), Pattern B (Remote Ring) 324 graphs (12.1%) and Pattern C (Unclassified) 1,679 graphs (62.4%). The substantial proportion of unclassified patterns (62.4%) suggests the existence of additional organizational motifs or hybrid mechanisms that warrant future investigation.

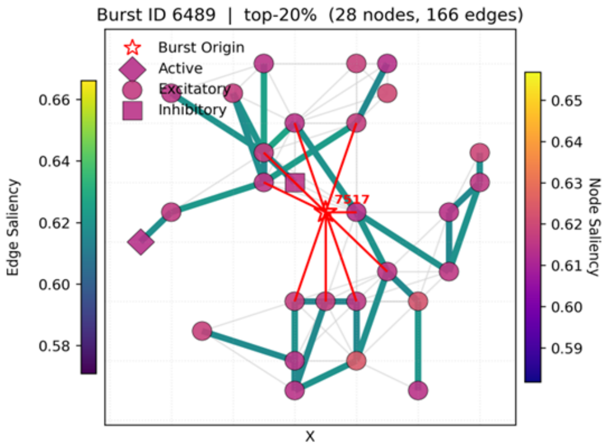


Figure 3.14: Pattern A: Local hub pattern. High-saliency neurons forming hub-like structures around origin. Medium - high scores (0.60 – 0.62 range). Hub-and-spoke architecture with central integration points.

Algorithm 1 Pattern Classification Based on Saliency and Proximity

```
Let originInTop $\leftarrow$ True if OriginNode $\in$ TopSalientNodes
Let percentWithin1Hop $\leftarrow$ % of TopSalientNodes within 1 hop of origin
Let percentAt2Hops $\leftarrow$ % of TopSalientNodes at exactly 2 hops
Let percentInRange $\leftarrow$ % with saliency $\in [0.60, 0.62]$
Let percentAboveThresh $\leftarrow$ % with saliency $> 0.63$
if (originInTop or percentWithin1Hop $\geq 33$) and percentInRange $\geq 70$ then
    return Pattern A: Local Hub
else if percentWithin1Hop $< 10$ and percentAt2Hops $\geq 80$ and percentAboveThresh $\geq 80$ then
    return Pattern B: Remote Ring
else
    return Pattern C: Mixed / Unclassified
end if
```

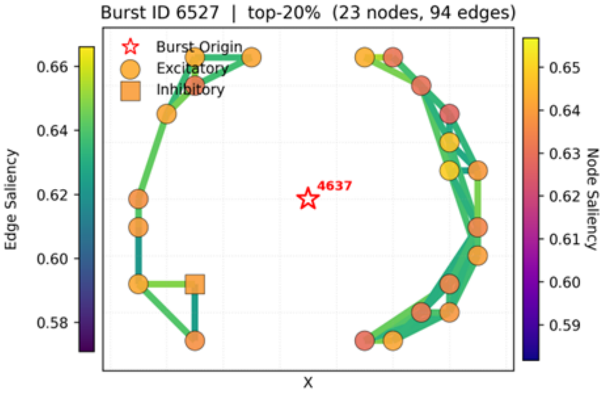


Figure 3.15: Pattern B: Remote ring pattern. High-saliency neurons forming ring-like structures exactly 2 hops from origin. High scores (greater than 0.63 range) and distributed regulatory circuit surrounding initiation site.

Graph-Level Saliency Metrics

For each detected pre-burst graph, we computed the following graph-level metrics to capture how saliency is distributed in space and how intense that activity is.

1. **Distance-Saliency Index (DSI):** Reflects the interplay between spatial dispersion and saliency magnitude. A higher DSI implies more spatially dispersed salient nodes; lower DSI suggests localization.

$$\text{DSI} = \frac{\sum_{v_i \in S} \text{Saliency}(v_i) \cdot \text{Distance}(v_i, v_{\text{origin}})}{\sum_{v_i \in S} \text{Saliency}(v_i)} \tag{3.1}$$

where:

- S is the set of top salient nodes

- Saliency(v_i) is the saliency score of node v_i
- Distance(v_i, v_{origin}) is the shortest path length (in hops) from node v_i to the burst origin v_{origin}

2. **Mean Hop Distance:** The average shortest-path distance (in network hops) between all pairs of top salient nodes. Lower values imply tighter clustering; higher values suggest spatial dispersion.

3. **Mean Saliency:** The average saliency value across the top salient nodes. This captures the overall intensity of important node activity in the graph.

4. **Saliency Entropy:** The Shannon entropy of the saliency distribution among the top salient nodes. This quantifies how evenly saliency is spread:

$$H = -\sum_i p_i \log(p_i) \quad \text{where} \quad p_i = \frac{\text{Saliency}(v_i)}{\sum_j \text{Saliency}(v_j)} \tag{3.2}$$

   A low entropy indicates dominance by a few nodes; high entropy indicates uniform distribution.

5. **Mean Betweenness Centrality:** The average betweenness centrality of the top salient nodes, representing their strategic position within the graph. Higher values suggest that salient nodes act as hubs or connectors.

Upon visualizing the distributions of these saliency-based graph metrics across all pre-burst graphs, we observed clear bimodal or clustered patterns in several metrics. This supports the presence of two dominant spatial motifs of burst initiation. These patterns and their implications are further elaborated in next chapter.

Chapter 4
RESULTS

4.1 Model Performance

Our Graph Convolutional Network (GCN) achieved robust performance in distinguishing pre-burst from non-burst subgraphs, with a final F_1-score of **99.6% ± 0.1%**. The model demonstrated excellent generalization with minimal overfitting, as evidenced by the convergence of training and validation loss and accuracy curves (see Figure 3.11).

The optimal window and mask sizes were identified through systematic hyperparameter optimization across a grid of:

- Window sizes: [5, 10, 20, 50]
- Mask sizes: [0, 2, 5, 10]

Results are shown as a heatmap in Figure 4.1.

Insights from Ablation Study

Our comprehensive ablation study validated that the model learns *meaningful graph-topology patterns* rather than exploiting data artifacts (see Table 3.2). Key observations include:

- **No data leakage:** Label shuffling drops performance to chance (48.9%).
- **Topology-invariant learning:** Node ID shuffling maintains performance (99.6%).
- **Feature necessity:** Constant node features collapse performance to chance (50.0%).
- **Graph structure value:** Removing edges costs 3.9% accuracy.

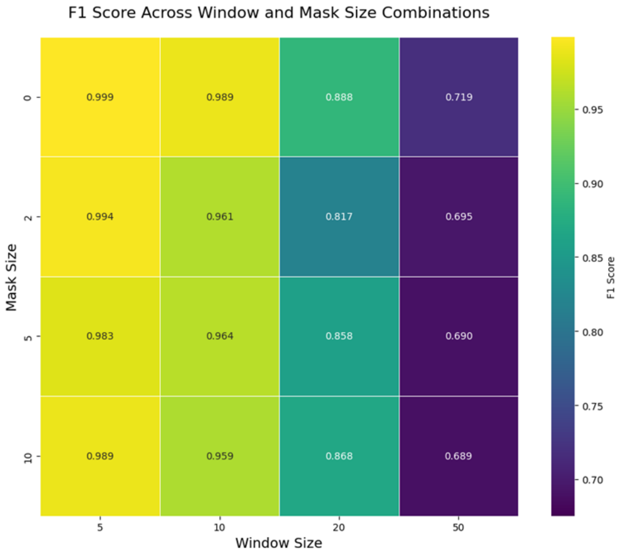


Figure 4.1: Hyperparameter testing for different window and Mask sizes. Results are shown for different window sizes from 5 to 50 spikes and mask size from 0 to 10 spikes. Each color in the square corresponds to the F1 score for the window and mask combination.

4.2 GNNExplainer-Based Interpretability Analysis

The explainability analysis revealed that the GCN's performance emerges from **distributed information integration** rather than dependence on specific critical circuits. While the GNNExplainer model demonstrated relatively low fidelity, it consistently revealed sparse sets of salient nodes and edges, many of which displayed repetitive organizational motifs across different bursts—supporting the core hypothesis of this thesis.

Evidence for Distributed Mechanism

1. **No critical subgraph identified:** Fidelity scores plateau regardless of sparsity threshold, suggesting no minimal set of nodes/edges dominates the decision.

2. **Spatial pattern organization:** Two motifs (Local Hub and Remote Ring) emerged from saliency analysis, found consistently across burst events.

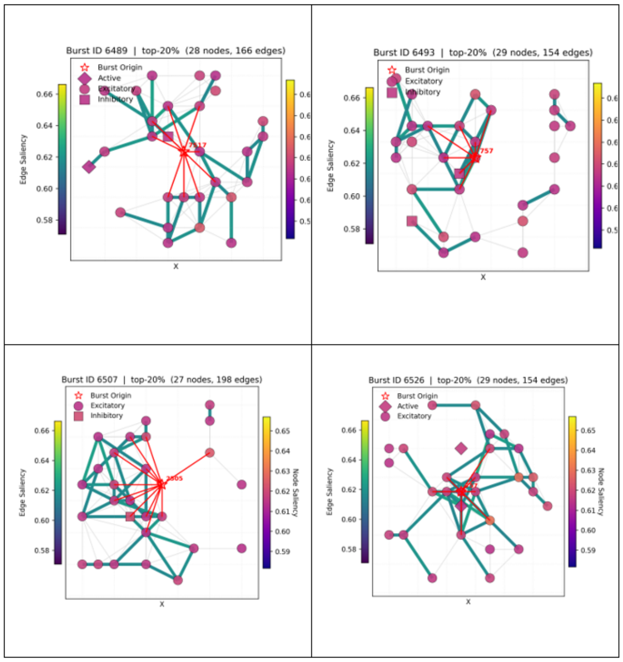


Figure 4.2: Pattern A Local Hub. Four examples picked at random from Pattern A category.

3. **Distance-dependent importance:** Saliency scores correlated positively with hop distance from the origin, indicating broader network influence.

While GNNExplainer struggled to assign high fidelity explanations, it successfully provided consistent per-graph sparse saliency maps. These explanations formed the basis of the spatial pattern discovery and reveal that salient activity is not concentrated around a single node or edge but rather distributed across structurally meaningful subgraphs. These findings support our central claim—that neural burst prediction arises from distributed, topology-dependent information patterns that span across 2-hop neighborhoods, rather than from a fixed set of trigger neurons. Redirecting from edge/node-level fidelity to spatial pattern analysis revealed **two spatial patterns** that predicted burst initiation shown in Figure 4.2 and Figure 4.3.

The four example patterns in Figures 4.4–4.5 illustrate representative **non-burst** subgraph instances drawn from Pattern A and Pattern B categories. Visually these instances

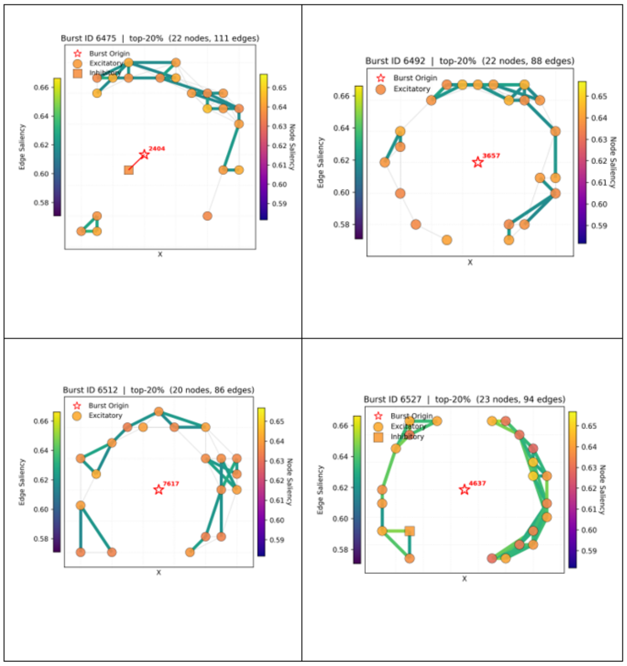


Figure 4.3: Pre-Burst Pattern B Remote Ring. Four examples picked at random from Pattern B category.

lack a consistent, repeatable motif: node placements, edge densities, and the distribution of node feature values (color scale) vary substantially from one panel to the next, and there is no single local arrangement of high-importance nodes or heavy edges that recurs across the examples. Instead we observe heterogeneous topologies — some subgraphs are sparse, others moderately connected; some show one or two locally dense clusters while others are more dispersed — and the temporal/feature signatures attached to nodes do not line up to form obvious pre-burst precursors. This heterogeneity suggests that the non-burst class is dominated by background activity and transient, event-independent fluctuations rather than by stable structural precursors.

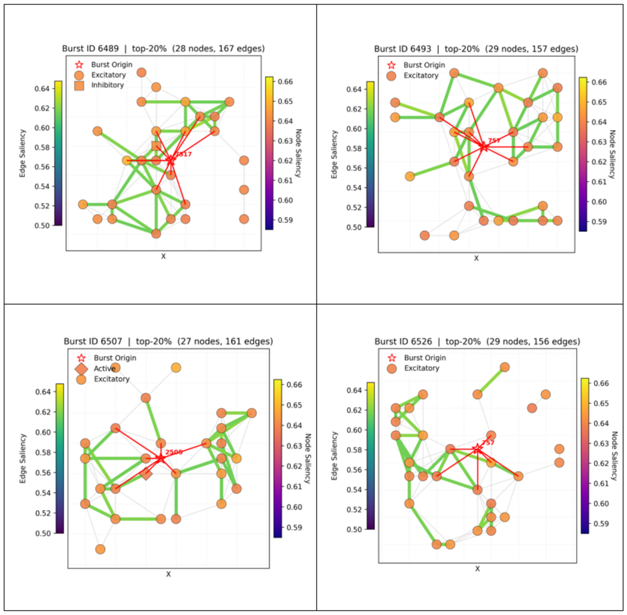


Figure 4.4: Non-Burst Pattern A instances. Four Non-Burst examples picked at random from Pattern A category.

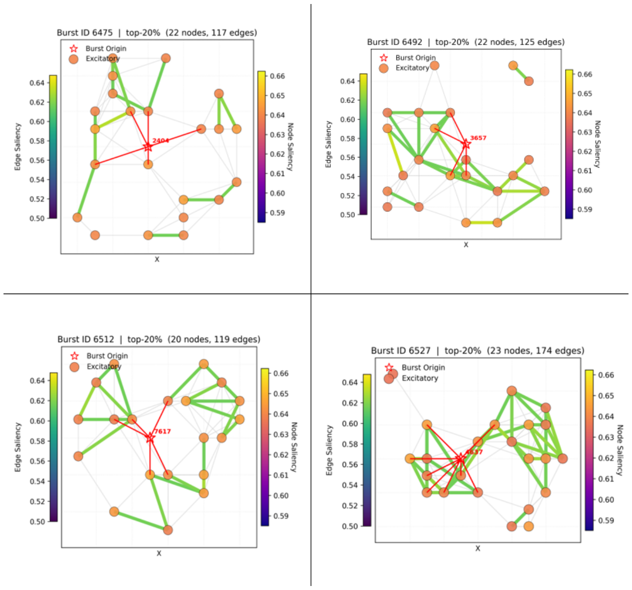


Figure 4.5: Non-Burst Pattern B instances. Four Non-Burst examples picked at random from Pattern B category.

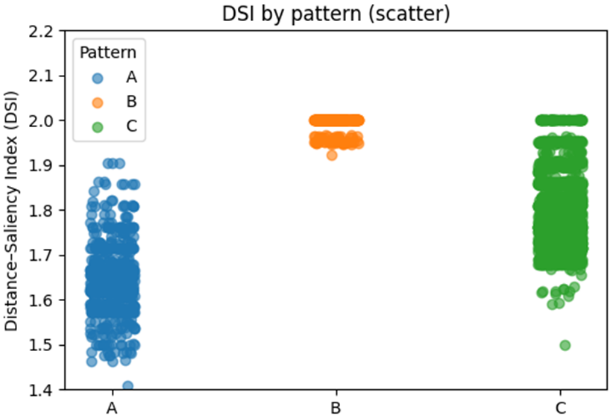


Figure 4.6: Distance-Saliency Index (DSI) by Pattern. Pattern B exhibits the highest median DSI (∼2.0), indicating dispersed saliency across distant nodes. Pattern A shows lower DSI (∼1.6), confirming its localized hub structure. Pattern C occupies an intermediate range with diverse spatial saliency distributions.

4.3 Pattern-Specific Metric Analysis

To further validate the spatial motifs discovered via explainability analysis, we systematically compared graph-level saliency metrics across the three identified pattern classes: Pattern A (Local Hub), Pattern B (Remote Ring), and Pattern C (Unclassified).

4.3.1 Distance-Saliency Index (DSI)

As can be seen from Figure 4.6, Pattern B graphs had markedly higher DSI values than Pattern A. The median DSI for Pattern B was close to 2.0, compared to about 1.6 for Pattern A. This difference was statistically significant ($p < 0.001$). Pattern B's high DSI reflects the combination of its broad spatial spread and strong saliency magnitude — a signature of the remote ring configuration. Pattern C graphs showed intermediate DSI values (median 1.7). Pairwise comparisons indicated that Pattern B's DSI was significantly greater than both Pattern A and Pattern C ($p < 0.001$ for B vs A; $p < 0.001$ for B vs C).

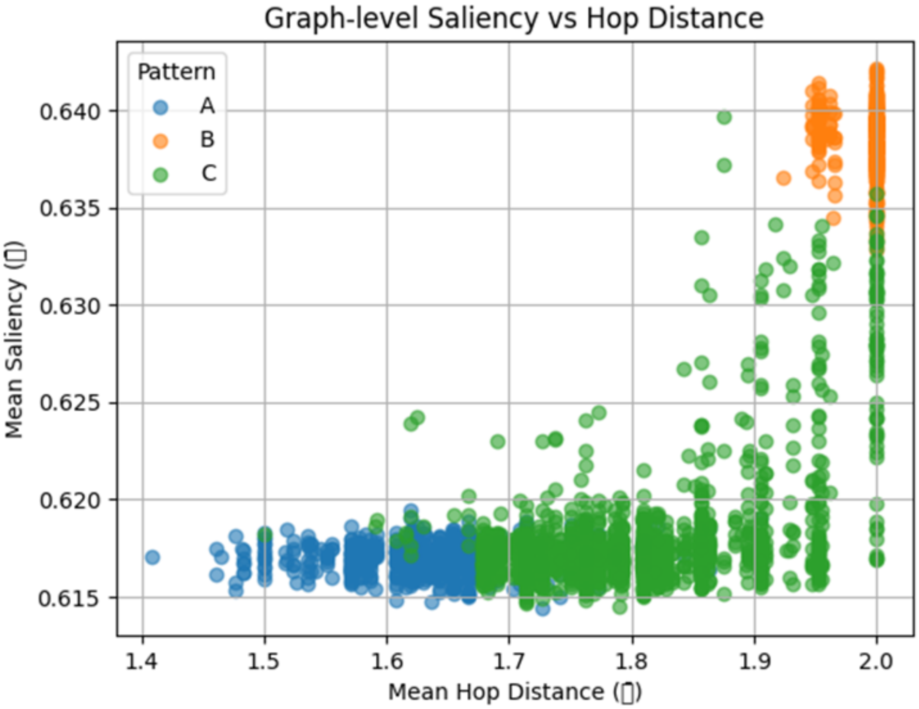


Figure 4.7: Graph-Level Saliency vs. Hop Distance. Pattern A shows low mean hop distance (∼1.5), indicating local clusters of salient nodes. Pattern B has high hop distance (∼1.95), indicating spatially remote saliency patterns. Pattern C is intermediate with greater variance.

4.3.2 Mean Hop Distance Between Salient Nodes

Pattern A and Pattern B were most clearly separated by the mean hop distance between salient nodes(See Figure 4.7). Pattern A graphs exhibited a small mean hop distance (median 1.5 hops), indicating that the key predictive nodes form a tight-knit local network. Pattern B graphs, in contrast, showed a much larger mean hop distance (median 1.95 hops), consistent with salient nodes scattered across remote parts of the network. This gap was highly significant ($p < 0.001$, Mann-Whitney U, Pattern B vs A). Unclassified graphs fell in between (median 1.7 hops) but with high variability. Statistical testing showed Pattern B also had a significantly higher mean hop distance than Pattern C ($p < 0.001$), while Pattern A had a significantly lower hop distance than Pattern C ($p < 0.01$). These findings quantitatively confirm that the local hub pattern involves a more compact spatial footprint of salient activity, whereas the remote ring pattern involves widely separated salient nodes. We observed a moderate difference in the average saliency strength of the key nodes between patterns. Pattern B graphs tended to have a higher mean saliency per node (roughly 0.63–0.64 in normalized units) compared to Pattern A graphs (around 0.615–0.620). In practical terms, this means that in Pattern B events, each salient node was, on average, slightly more strongly "activated" or predictive of the burst. This difference, while not as large as those for DSI or hop distance, was statistically significant ($p < 0.001$ for Pattern B vs A). Pattern C's mean saliency was around 0.620 on average, which was not significantly different from Pattern A (no significant difference, $p > 0.1$ for A vs C), but was lower than Pattern B ($p < 0.01$ for B vs C). Thus, Pattern B stands out as having both distant spatial configuration and slightly stronger individual node saliency, whereas Pattern A's clustered pattern is achieved with somewhat lower per-node saliency.

4.3.3 Saliency Entropy

Saliency entropy distinguished the patterns clearly (See Figure 4.8): An interesting difference emerged in how evenly distributed the saliency was among the top nodes of each pattern. Pattern A (local hub) graphs generally showed a higher saliency entropy, meaning the saliency was more evenly shared among multiple nodes in the local cluster. In many Pattern A cases, a cluster of nodes collectively contributed, yielding a moderately broad saliency distribution (entropy often in the range of 0.3–0.6 in our normalized units). Pattern B (remote ring) graphs, in contrast, typically exhibited lower saliency entropy. This indicates that Pattern B often had one or two nodes carrying a large portion of the saliency weight, with the remaining salient nodes contributing less. In fact, several Pattern B instances were characterized by one strongly salient node (or a few) driving the burst prediction with others playing a minor role, resulting in very low entropy values (sometimes near 0, indicating a highly skewed saliency distribution). A quantitative comparison found significantly lower entropy in Pattern B than in Pattern A ($p < 0.001$, B vs A), which aligns with the qualitative "hub" vs "distributed

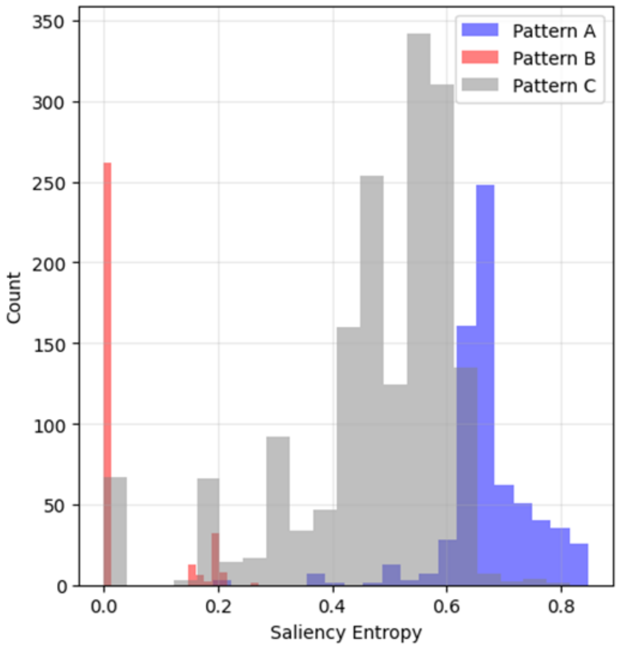


Figure 4.8: Saliency Entropy Distribution Across Patterns. Pattern A shows moderately high entropy, reflecting a more even saliency spread across clustered nodes. Pattern B has low entropy, often dominated by one or two highly salient nodes. Pattern C spans a wide range.

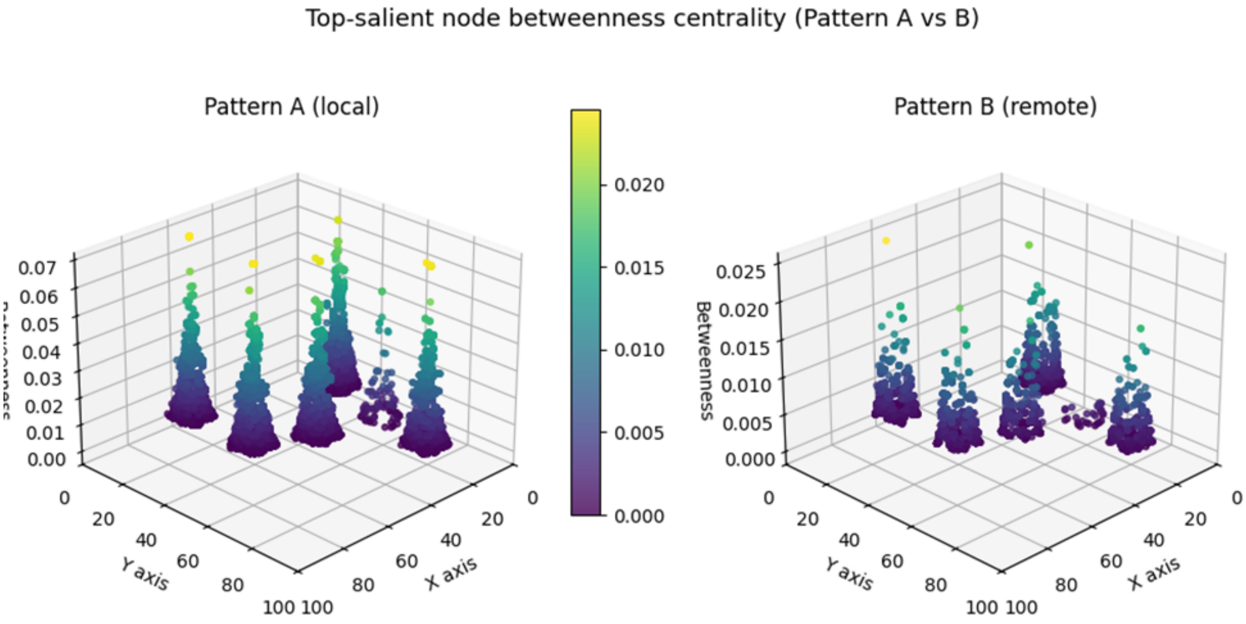


Figure 4.9: 3D Visualization of Betweenness Centrality in Top-Salient Nodes. Pattern A graphs feature high-centrality hub nodes, while Pattern B graphs show uniformly low betweenness across dispersed salient nodes.

ring" distinction: the local hub pattern's saliency is spread among the cluster, whereas the remote ring often has a single critical node (perhaps bridging a gap in the ring) that dominates. Pattern C graphs showed a wide range of entropy values (some approaching Pattern A levels, others as low as Pattern B), reflecting their heterogeneous nature. Overall, Pattern A had the highest entropy on average, Pattern B the lowest, and this difference was one of the more pronounced distinctions between the two patterns.

4.3.4 Betweenness Centrality of Salient Nodes

We also compared the network betweenness centrality of salient nodes between patterns to gauge their topological roles (Figure 4.9). Pattern A's defining "local hub" aspect suggested that one of the salient nodes might serve as a hub or connector within the network. Indeed, we found that Pattern A salient nodes had significantly higher betweenness centrality on average than those in Pattern B. The median mean betweenness for Pattern A was roughly 0.01 (in normalized units), compared to about 0.003–0.005 for Pattern B – a substantial difference (p < 0.001, B vs A). In practical terms, this means that the key nodes in Pattern A graphs often occupy central bridging positions in the network's connectivity (for example, a hub node that links several nearby nodes and channels the flow of activity within that local cluster). By contrast, Pattern B salient nodes generally had low betweenness centrality, consistent with the notion that in the remote ring configuration, the salient nodes lie on the periphery or in distinct modules without one node acting as a major intermediary. Pattern C again showed intermediate behavior: some unclassified graphs had moderately high-betweenness salient nodes (like Pattern A), while others did not, resulting in a broad distribution of mean betweenness (median 0.008). Our statistical tests indicated that Pattern B's mean betweenness was significantly lower than both Pattern A and Pattern C ($p < 0.001$ for B vs A; $p < 0.01$ for B vs C), whereas Pattern A's betweenness was slightly higher than Pattern C on average (though the A vs C difference was smaller, $p < 0.05$). These findings reinforce the idea that Pattern A is characterized by the presence of a local hub node, whereas Pattern B lacks any high-centrality hub among its salient nodes. This observation was visually supported by 3D network plots Figure showing the spatial layout of top-salient nodes colored by betweenness centrality: Pattern A graphs often featured one brightly colored node (high betweenness) at the center of a cluster of salient nodes, while Pattern B graphs showed uniformly low betweenness colors for all their distant salient node.

4.3.5 Excitatory/Inhibitory Composition of Salient Nodes

Finally, we examined whether the two patterns differed in the composition of salient nodes by cell type (excitatory vs. inhibitory neurons, See Figure 4.10). This was done as a supplemental characterization step, recognizing that the saliency method identifies important nodes regardless of type. A bar chart comparison of the E/I composition of top-salient nodes in Pattern A vs Pattern B did not reveal a large difference between the patterns. In both Pattern A and Pattern B graphs, most salient nodes were excitatory neurons ($> 97\%$ ex-

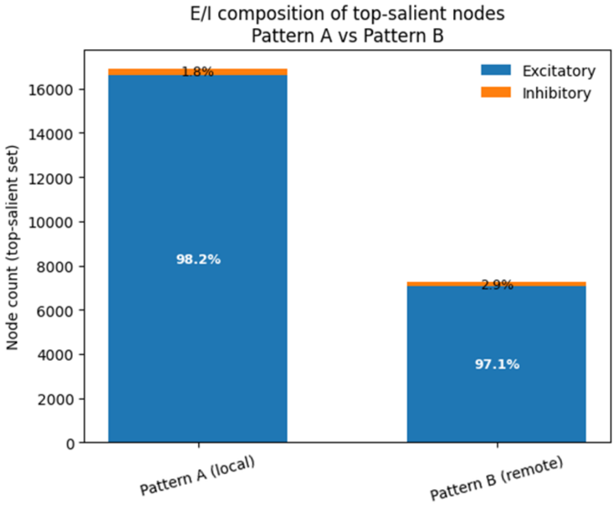


Figure 4.10: E/I Composition of Top-Salient Nodes (Pattern A vs Pattern B). The majority of salient nodes in both patterns are excitatory neurons (>97%), with a slightly higher proportion of inhibitory neurons in Pattern B (2.9%) compared to Pattern A (1.8%).

citatory in both cases), with only a small fraction being inhibitory (approximately 1.8% in Pattern A vs 2.9% in Pattern B). Thus, both patterns' salient-node sets are dominated by excitatory units, and any slight increase in inhibitory participation in Pattern B was not substantial in absolute terms.

To summarize, our combination of quantitative metric analysis and statistical validation revealed two robust spatial patterns of salient neural activity preceding burst initiation. The first, Pattern A (Local Hub Amplification), is characterized by clusters of salient nodes concentrated near the burst origin, exhibiting high betweenness centrality, moderate entropy, and low DSI. In contrast, Pattern B (Remote Ring Mechanism) is defined by salient nodes located at greater spatial distances from the origin, marked by high DSI, low entropy, and low centrality. Pattern C graphs showed intermediate or mixed characteristics and may contain additional motifs requiring further study. These findings challenge the origin-centric view of burst initiation and suggest that burst events are preceded by distributed network-wide preparatory states.

Chapter 5

DISCUSSION AND FUTURE WORK

This thesis presented a graph-centered framework for understanding burst initiation in large-scale simulated cortical networks. By leveraging Graph Neural Networks (GNNs), we demonstrated that both neural activity statistics and the underlying synaptic topology can be jointly modeled to classify and explain pre-burst dynamics with high accuracy. Unlike traditional approaches that either isolate temporal features or rely solely on structural metrics, our work highlights the power of graph-based representations in capturing the distributed interactions that give rise to emergent neural behaviors.

A key contribution of this study lies in the integration of a post-hoc interpretability. Through systematic analysis of explanations across 2,689 burst events, we showed that model decisions are driven not by activity localized at the burst origin, but by distributed precursor patterns spanning the network. This discovery challenges prevailing views of burst initiation as a primarily local phenomenon and instead emphasizes the role of wider circuit motifs in tipping the system into synchrony. Two recurring motifs—Local Hub Amplification and the Remote Ring Mechanism—were identified as robust signatures of pre-burst activity, providing a new lens through which to interpret neural population dynamics.

It is important to acknowledge the contrast between real neural data and the simulated data used in this work. Multi-electrode array (MEA) recordings, while widely used in experimental neuroscience, often suffer from noise, limited spatial sampling, and incomplete coverage of the neural population. These limitations make it difficult to reconstruct the full connectivity and activity patterns underlying burst initiation. In contrast, our simulated networks provide a complete, noise-free ground truth, allowing precise integration of activity features with synaptic topology.

While our study offers novel insights into the connectivity and activity patterns in pre-burst subgraphs, several limitations must be acknowledged—each of which opens pathways for future research:

First, in our current approach, we extract two subgraphs per burst event (pre-burst and non-burst) using a fixed underlying connectivity structure, differing only in node-level temporal features. However, the network's topology naturally evolves over time. Future work could incorporate dynamic graph construction by reconstructing the connectivity at each timestep (or temporal bin) within the window. This would enable the model to learn from both structural and temporal changes, potentially uncovering transient motifs that more accurately signal burst initiation.

Second, our experiments were conducted under a fixed set of growth parameters (e.g., synaptic growth factor ε and the proportion of excitatory neurons). To assess the generalizability of our findings, future studies should apply the proposed pipeline to simulations with varying connectivity growth regimes and excitatory/inhibitory cell ratios. This could reveal whether the same predictive motifs persist or whether alternative "burst-leader" patterns emerge under different developmental conditions.

Third, we utilized a standard Graph Convolutional Network (GCN) architecture in combination with GNNExplainer. While effective, this pairing may not fully capture higher-order structural dependencies or directional signaling. Future work should explore advanced GNN architectures—such as Graph Attention Networks (GAT), EdgeConv, or spectral-based models—which are better suited to modeling edge importance and complex non-linear interactions. Similarly, integrating more powerful explainability tools like PGExplainer, SubgraphX, or Counterfactual GNNs could yield more informative and interpretable explanations.

Finally, our analysis used a fixed temporal window size and mask parameter to define the context for classification and explanation. Exploring a range of window and mask sizes may reveal how temporal context influences both model predictions and the saliency of precursor patterns, potentially leading to more robust and biologically relevant insights.

In conclusion, this thesis demonstrates that graph-based learning combined with interpretability offers a powerful paradigm for uncovering the hidden drivers of emergent neural dynamics. The insights gained here not only deepen our understanding of burst initiation but also point toward a broader role for explainable GNNs in neuroscience, where prediction and mechanistic explanation must go hand in hand.

BIBLIOGRAPHY

[1] Vanessa Arndorfer. Network behavior analysis of spike timing dependent plasticity (stdp) in simulated neural networks. Master's thesis, University of Washington, 2025.

[2] Matthew N. Bernstein. Graph convolutional neural networks. Blog post, 2023. https://mbernste.github.io/posts/gcn/, accessed 2025-08-05.

[3] T. Tony Cai and Rong Ma. Theoretical foundations of t-SNE for visualizing high-dimensional clustered data. *Journal of Machine Learning Research*, 23(301):1–54, 2022.

[4] R. Chen, J. Wen, X. Li, et al. Rich dynamics and functional organization on topographically patterned pdms substrates reveal distinct burst initiation points. *Bioelectrochemistry*, 150:107957, 2022.

[5] Emily Cotterill and Stephen J. Eglen. Burst detection methods. https://arxiv.org/abs/1802.01287, 2018. Preprint; reviewed burst detection techniques in spike trains.

[6] DataCamp. A comprehensive introduction to graph neural networks (gnns). https://www.datacamp.com/tutorial/comprehensive-introduction-graph-neural-networks-gnns-tutorial, 2023. Accessed: August 5, 2025.

[7] Mallory Dazza, Stephane Métens, Pascal Monceau, and Samuel Bottani. A novel methodology to describe neuronal networks activity reveals spatiotemporal recruitment dynamics of synchronous bursting states. *Journal of Computational Neuroscience*, 49(4):375–394, 2021.

[8] Arnaud Delorme, Scott Makeig, Michèle Fabre-Thorpe, and Terrence J. Sejnowski. From single-trial eeg to brain area dynamics. *Neurocomputing*, 44–46:1057–1064, 2002.

[9] Arnaud Delorme, Jason Palmer, Julie Onton, Robert Oostenveld, and Scott Makeig. Independent eeg sources are dipolar. *PLoS ONE*, 7(2):e30135, 2012.

[10] Wassim M. Haddad, Qing Hui, and James M. Bailey. Human brain networks: Spiking neuron models, multistability, synchronization, thermodynamics, maximum entropy production, and anesthetic cascade mechanisms. *Entropy*, 16(7):3939–4003, 2014.

[11] Wassim M. Haddad, Qing Hui, and James M. Bailey. Human brain networks: Spiking neuron models, multistability, synchronization, thermodynamics, maximum entropy production, and anesthetic cascade mechanisms. *Entropy*, 16(7):3939–4003, 2014.

[12] Michael I. Ham, Luis M. A. Bettencourt, Floyd D. McDaniel, and Guenter W. Gross. Spontaneous coordinated activity in cultured networks: Analysis of multiple ignition sites, primary circuits, and burst phase delay distributions. *Journal of Computational Neuroscience*, 24(3):346–357, 2008.

[13] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems*, 30, 2017.

[14] Qikun Huang, Makoto Yamada, Yu Tian, Dawei Yin Singh, and Yi Chang. Graphlime: Local interpretable model explanations for graph neural networks. In *IEEE International Conference on Big Data (Big Data)*, 2020.

[15] James J. Jun, Nicholas A. Steinmetz, Joshua H. Siegle, Daniel J. Denman, Mircea Bauza, C. Barbarits, and et al. Fully integrated silicon probes for high-density recording of neural activity. *Nature*, 551(7679):232–236, 2017.

[16] Fumitaka Kawasaki and Michael Stiber. A simple model of cortical culture growth: Burst property dependence on network composition and activity. *Biological Cybernetics*, 108(4):423–443, 2014.

[17] Taehoon Kim, Dexiong Chen, Philipp Hornauer, Vishalini Emmenegger, Julian Bartram, Silvia Ronchi, Andreas Hierlemann, Manuel Schröter, and Damian Roqueiro.

Predicting in vitro single-neuron firing rates upon pharmacological perturbation using graph neural networks. *Frontiers in Neuroinformatics*, 16, 2022.

[18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[19] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, 2017. arXiv:1609.02907.

[20] Toki Kobayashi, Kenta Shimba, Taiyo Narumi, Takahiro Asahina, Kiyoshi Kotani, Yasuhiko Jimbo, et al. Revealing single-neuron and network-activity interaction by combining high-density microelectrode array and optogenetics. *Nature Communications*, 15(1):9547, 2024.

[21] J. Köhler, C. Müller, M. Schumacher, et al. Astrocytic regulation of synchronous bursting in cortical cultures. *eNeuro*, 8(2):ENEURO.0214–20.2021, 2021.

[22] Jewel Yun-Hsuan Lee. Machine learning of spatiotemporal bursting behavior in developing neural networks. Master's thesis, University of Washington, 2018.

[23] Jewel Yun-Hsuan Lee, Michael Stiber, and Dong Si. Machine learning of spatiotemporal bursting behavior in developing neural networks. In *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 348–351. IEEE, 2018.

[24] Michael S. Lewicki. A review of methods for spike sorting: the detection and classification of neural action potentials. *Network: Computation in Neural Systems*, 9(4):R53–R78, 1998.

[25] Y. Li and M. Nguyen. Extracellular matrix feedback shifts spiking networks from asynchronous to quasi-synchronous bursting. *PLOS Computational Biology*, 20(4):e1012356, 2024.

[26] Jin Liu, Lei Wang, Tao Zhang, Xue Jia, Wen-Wei Shao, Ning Hu, Ji Shi, Xing Fan, Chao Chen, Yong Wang, Lei Chen, Guan-Ji Qiao, and Xiao-Hong Li. Learning populations with hubs govern the initiation and propagation of spontaneous bursts in cultured networks of rat cortical neurons in vitro. *Frontiers in Neuroscience*, 16:854199, 2022.

[27] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[28] Fabien Lotte, Laurent Bougrain, Andrzej Cichocki, Maïa Clerc, Marco Congedo, Alain Rakotomamonjy, and Florian Yger. A review of classification algorithms for eeg-based brain-computer interfaces: a 10 year update. *Journal of Neural Engineering*, 15(3):031005, jun 2018. Epub 2018 Feb 28. PMID: 29488902.

[29] Dexin Luo, Wenbing Cheng, Dongkuan Xu, Dongxiao Yu, and Hongyuan Zha. Parameterizable explainer for graph neural network. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[30] E. Maeda, H. P. Robinson, and A. Kawana. The mechanisms of generation and propagation of synchronized bursting in developing networks of cortical neurons. *Journal of Neuroscience*, 15(10):6834–6845, 1995.

[31] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in python, 2011.

[32] Phillip E. Pope, Soheil Kolouri, Mohammad Rostami, Charles E. Martin, and Heiko Hoffmann. Explainability methods for graph convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10772–10781, 2019.

[33] S. Pradeepan, A. Gershoni, O. Cohen, et al. Unveiling mechanisms behind fragmented network bursts in patient-derived excitatory cortical neuron cultures. *Neurobiology of Disease*, 184:106144, 2024.

[34] PyG Team. torch_geometric.data.data — pytorch geometric documentation. https://pytorch-geometric.readthedocs.io/en/latest/generated/torch$_geometric.data.Data.html$, 2024. $2025-07-31$.

[35] S. Raghu et al. Spike classification using machine learning approaches, 2020. Preprint; exact source (DOI / arXiv ID / URL) needed to complete entry.

[36] T. H. Reijmers, R. Wehrens, and L. M. C. Buydens. The influence of different structure representations on the clustering of an rna nucleotides data set. *Journal of Chemical Information and Computer Sciences*, 41(5):1388–1394, 2001.

[37] scikit-learn developers. sklearn.model_selection.groupshufflesplit. https://scikit-learn.org/stable/modules/generated/sklearn.model$_selection.GroupShuffleSplit.html$, 2024.*Accessed* $2025-07-31$.

[38] Danial Sharifrazi, Nouman Javed, Javad Hassannataj Joloudari, Roohallah Alizadehsani, Prasad N. Paradkar, Ru-San Tan, U. Rajendra Acharya, and Asim Bhatti. Functional classification of spiking signal data using artificial intelligence techniques: A review. *arXiv preprint arXiv:2409.17516*, 2024.

[39] S. Singh. Graph analysis for simulated neural networks with stdp. Master's thesis, University of Washington, 2021.

[40] Smriti Singh. Understanding localized burst trigger patterns in developing neural networks using deep learning. Master's thesis, University of Washington, 2020.

[41] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, 2018.

[42] Paul Y. Wang, Sandalika Sapra, Vivek Kurien George, and Gabriel A. Silva. Generalizable machine learning in neuroscience using graph neural networks. *Frontiers in Artificial Intelligence*, 4:618372, 2021.

[43] Qi Wang and Longfei Zhang. Inverse design of glass structure with deep graph neural networks. *Nature Communications*, 12(1):3508, 2021.

[44] Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[45] Hao Yuan and Jiliang Yu. Explainability in graph neural networks: A taxonomic survey. *ACM Computing Surveys (CSUR)*, 55(3):1–37, 2022.

[46] Yun Zhao, Elmer Guzman, Morgane Audouard, Zhuowei Cheng, Paul K. Hansma, Kenneth S. Kosik, and Linda Petzold. A deep learning framework for classification of *in vitro* multi-electrode array recordings. *arXiv preprint arXiv:1906.02241*, 2019. Demonstrates CNNs outperform logistic regression on genotype classification from MEA recordings.