CBGT-Net: A Neuromimetic Architecture for Robust Classification of Streaming Data

Shreya Sharma

CMU-RI-TR-24-29

June 12

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Prof. Katia Sycara, *chair*
Prof. Steven Chase
Ini Oguntola

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.

To my parents.

Abstract

This research introduces CBGT-Net, a neural network model inspired by the cortico-basal ganglia-thalamic (CBGT) circuits in mammalian brains, which are crucial for critical thinking and decision-making. Unlike traditional neural network models that generate an output for each input or after a fixed sequence of inputs, CBGT-Net learns to produce an output once sufficient evidence for action is accumulated from a stream of observed data. For each observation, CBGT-Net generates a vector representing the amount of evidence for each potential decision, accumulates this evidence over time, and makes a decision when the accumulated evidence surpasses a predefined or dynamically learned threshold.

We evaluate the proposed model on various image classification tasks, where models must predict image categories based on a stream of partially informative visual inputs. Our results demonstrate that CBGT-Net offers improved accuracy and robustness compared to models trained to classify from a single image, as well as models utilizing an LSTM layer or a ViT-style transformer to classify from a fixed sequence of image inputs. Additionally, we introduce a novel dataset for classification based on sequential image data of urban city buildings. This dataset provides multi-view images of 3D building assets on fire, categorized into five stages of fire severity.

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my advisor, Prof. Katia Sycara, for her invaluable guidance, continuous support, and patience throughout my research. Her insight and expertise have been instrumental in shaping this thesis. Prof. Sycara's passion and commitment to research and her lab have been a tremendous source of inspiration, leaving me in awe of her dedication to her work. I aspire to attain even a fraction of the dedication and enjoyment she derives from her work. I look up to her as a role model and feel deeply grateful and fortunate to have had the opportunity to work with and learn from her over the past two years.

I am also grateful to the members of my thesis committee, Prof. Steven Chase and Ini Oguntola, for their constructive feedback and encouragement which have been critical in the development of this thesis.

I would also like to extend my heartfelt thanks to my mentor, Dr. Dana Hughes, who supported me at every step of this project from day one until the evening before my thesis presentation. Your constant guidance, great insights, and willingness to help as I navigated through the research project have been invaluable. I feel fortunate to have had a mentor who dedicated so much time to discussing bottlenecks and brainstorming ideas to keep the project on track. Your hands-on involvement and mentorship have taught me a great deal, especially in improving my presentation skills, which you consistently helped refine during our weekly meetings. I am deeply grateful for your support and guidance throughout this journey.

Special thanks to my colleagues in the Advanced Agent-Robotics Technology (AART) Lab. I would like to thank my collaborators, Seth and Venkat (Arjun), for making the project enjoyable. I truly appreciated our discussions and the chance to work on this project together. It has been a pleasure to work with you as a team. I am also grateful to Joe, Simon, Yaqi, and Woojun, who served as mentors to me. I would also like to give a shout out to my colleagues and friends—Sarthak, Zach, Srujan, Aryan, Silong, Nishant, Renos, Sofie, and Yi Sha. We've shared many memorable moments in the lab, at lab events, and during outings, which I will cherish forever. I feel fortunate to have been part of such a supportive and stimulating research environment, where insightful discussions with

all of you have made this journey of 2 years a truly enriching experience.

I couldn't have survived these 2 years living 8000 miles away from my family without the very important presence of my friends in my life. They have been my rock through the highs and lows, always there to lift my spirits in times of need and celebrate my successes with genuine pride. I will always cherish our coffee breaks, table tennis games, and random trips fondly. I am immensely grateful to have met these exceptional individuals who I proudly call my best buds. In alphabetical order, I want to acknowledge Anish Madan, Anubha Kabra, Dvij Kalaria, Leon Zamel, Poorvi Hebbar, Pranay Gupta, Pushkal Katara, Sarthak Bhagat, Shagun Uppal, Sriram Narayanan, and Tobias Schindler, who have been there for me throughout my two years pursuing my MSR degree at CMU. Special mention to Shagun Uppal for being an exceptional listener, always there to listen to me talk about anything and everything, and for keeping me sane with her unwavering support. I will always cherish our coffee catch-ups, long walks around campus, midnight snacking, and the deep (as well as completely stupid) conversations. Special thanks to Dvij for always being there for me and encouraging me to strive for a better version of myself. I feel incredibly fortunate to have you as a friend who understands me when no one else can and supports me unconditionally. I would also like to thank Gauri, my best friend who has been my unwavering pillar of support throughout my Master's journey. I am thankful for her constant motivation and inspiration, pushing me to strive for more in life and always being there for me like a fire brigade.

Last but certainly not least, I extend my heartfelt gratitude to my parents, who are the foundation of my journey and to whom I owe all my success. I am deeply thankful for the nurturing environment, unwavering support, encouragement, freedom, and staunch belief they have instilled in me. I am and will be forever thankful for the education and opportunities they provided, allowing me to spread my wings and live life to the fullest, all while making countless sacrifices themselves. I would also like to thank my brother, Aviral, for always being my closest companion and my source of energy and strength. Additionally, I am grateful to my grandparents and Kaka for their unconditional love, and blessings, and for imparting invaluable values that have shaped me into a better human being.

Contents

When this dissertation is viewed as a PDF, the page header is a link to this Table of Contents.

List of Figures

List of Tables

Chapter 1

Introduction

In the realm of neuroscience and machine learning, the pursuit of sophisticated neural architectures capable of navigating dynamic decision-making in ever-changing environments has been a focal point of research. Drawing inspiration from the intricate cortico-basal ganglia-thalamic (CBGT) circuits observed in mammalian brains, this thesis is focused on exploring and implementing a groundbreaking neural network model known as CBGT-Net. Building upon the foundational principles of decision thresholds and encoder learning elucidated in prior studies [1], this thesis extends the horizon to encompass the complexities of data streams, with a specific emphasis on image data classification.

The CBGT-Net model not only demonstrates remarkable adaptability to diverse encoding layers but also introduces a novel supervised training approach that places a premium on transparent deliberation in decision-making processes. By explicitly focusing on regulating the speed and precision of decision-making, this research endeavors to contribute significantly to the advancement of autonomous systems capable of adeptly handling a myriad of decision tasks across various domains.

Through a meticulous examination of the CBGT-Net model's performance in learning from low-information data streams, this thesis aims to shed light on the model's efficacy in discerning patterns and making informed decisions in scenarios where data availability may be limited or noisy. By delving into the nuances of decision thresholds and encoder learning within the CBGT-Net framework, this research seeks to pave the way for more robust and efficient decision-making mechanisms in neural networks, ultimately propelling the field towards enhanced autonomy and adaptability in intelligent systems.

In real-world scenarios, agents often operate under conditions of incomplete information, limited sensing capabilities, and inherently stochastic environments, resulting in observations that are incomplete and unreliable. Additionally, it is often preferable to delay making a decision rather than risk a poor one. In such cases, it is crucial to aggregate information before taking action. However, most conventional neural network algorithms tend to make predictions or take actions at every time step, even when the agent lacks confidence in its choice. This lack of caution can lead to critical mistakes, regardless of the agent's prior experience and acclimation to the environment. Inspired by theories of dynamic resolution of uncertainty during decision-making in biological brains, we propose a neuro-inspired module. This module gathers evidence for each possible decision, encodes uncertainty as a dynamic competition between actions, and acts only when it is sufficiently confident in the chosen action. The agent makes no decision by default; the burden of proof lies on the accumulation process to gather enough evidence in favor of a single decision to cross a threshold before action is taken.

Significant strides in deep learning have led to remarkable advancements across various domains such as image classification, natural language comprehension, and decision-making [17]. The success of such methods arises from multi-layered architectures capable of learning feature mappings at increasing levels of abstraction from large datasets. Despite the success of deep learning, models are trained in an end-to-end manner and generally produce an output for every provided input. Generated output may be incorrect—often with a high level of confidence—with minimal perturbation to the input [20], and traditional neural network models are not designed to consider when a single input is insufficient for inference purposes. For instance, traditional image classification models generate a category for a single image without the ability to consider additional viewpoints, while policies learned for control are designed to generate an action regardless of how complete or noisy the observation is.

In contrast, models of decision-making in primate brains have been developed where decisions are made based on the integration of noisy information over time [18]. In these models, evidence for a response is accumulated until a requisite amount is reached, explaining response accuracy and timing. Specifically, the cortico-basal ganglia-thalamic (CBGT) circuits in the brain have been shown to play a role in action selection [13, 19], including describing means of evidence accumulation for and response criteria of competing actions [16]. In essence, this circuitry deliberates over potential actions based on a stream of noisy or incomplete information from multiple cortical areas.

Inspired by the evidence accumulation aspect of primate decision-making, our research aims to develop and evaluate a neuromimetic model of the CBGT circuit in mammalian brains. We believe such a model would provide several desired features in autonomous decision-making—in addition to potentially improved model accuracy, the deliberation process of the model is transparent, allowing for better interpretability during human-autonomy collaborations. Building on our prior proof-of-concept work in this area [1], we present a CBGT-inspired neural network architecture[1] and evaluate its ability to learn to integrate noisy information, as well as determine the effect of varying evidence criterion, in complex domains (i.e., vision-based tasks). We demonstrate that the proposed model is able to perform classification tasks using a stream of incomplete information more accurately than models trained to classify based on a single observation, and also generally outperforms LSTM-based sequential models in terms of accuracy and data efficiency. Additionally, our model's performance is robust to decreasing information in observations, compared to the LSTM models. Finally, our model is designed to make decisions based on acquiring a sufficient amount of evidence, as opposed to a fixed amount of time, which is easily adjusted during deployment using a simple decision threshold level.

This paper is organized as follows: Section 2 describes relevant work related to our approach; Section 2.1 provides a brief description of the CBGT circuit in mammalian brains; Section 3 describes the architecture and training approach for our model; Section 4.1.1 and 4.2 describes our evaluation and results. Section 5 provides discussion and future work.

[1] Code available at https://github.com/ShreyaSharma99/CBGT-Net

Chapter 2

Related Work

In the field of neuroscience, computational models of basal ganglia circuitry have been used to explore aspects of decision making in dynamic environments and its role in reinforcement learning. For instance, competition between neural pathways in the basal ganglia has been proposed as a model of action uncertainty [4] and for describing exploration-exploitation tradeoffs in volatile environments [2]. While previous research has showcased the basal ganglia's involvement in different facets of decision-making, the existing models predominantly investigate the biological dimensions of decision-making, such as response time. In contrast, our focus lies in developing models tailored for machine learning tasks with inspiration drawn from neuroscience.

In the area of deep neural networks, confidence-aware learning aims to not only accurately perform some inference task (e.g., image classification), but to also assign a confidence score to each inference. In [15], a correctness ranking loss is utilized to ordinally rank training examples and produce a confidence score for classification tasks. In [14], training loss is augmented with a distance loss to encourage clustering of training examples in an embedding space; post-training, the distance of a novel data point to the nearest neighbor of the training data in the embedding space is leveraged as a confidence score. While such confidence scores are analogous to our usage of evidence, these approaches aim to generate confidence scores of a single prediction, while in our approach, the evidence encoder learns to produce a value akin to confidence when learning with a stream of data.

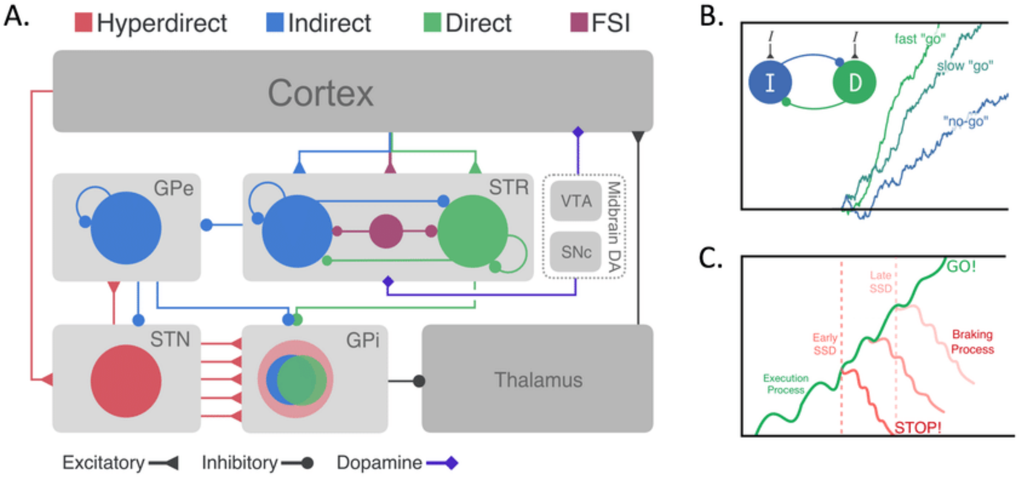


Figure 2.1: Cortico-Basal Ganglia Thalamus (CBGT) Circuits in Mammalian Brains

Our prior exploration into developing a CBGT-inspired network [1] demonstrates a proof-of-concept network capable of learning decision thresholds and very simple encoders; in this paper, we extend this effort to learn encoders for more complex data streams (e.g., images), demonstrate that our approach is agnostic to encoding layers, and utilize a more effective supervised training approach.

2.1 Cortico-Basal Ganglia Thalamic Circuit

The network architecture presented in this paper is inspired by CBGT circuits in mammalian brains, and the role they play in decision making and evidence accumulation [13, 16, 19], as shown in figure 2.1. Corticostriatal connections provide pathways for projection from the functional areas of the cortex—including the sensorimotor, associative, and limbic areas—to the striatum. For each potential action (i.e., motor neuron activation), two pathways exist in the basal ganglia that facilitate or suppress the action in the thalamus: Direct (“Go”) pathways inhibit the globus pallidus internus (GPi), which in turn causes disinhibition of the thalamus and facilitation of the action corresponding to the circuit; Indirect (“NoGo”) pathways inhibit the globus pallidus externus (GPe), which in turn disinhibits the GPi and suppresses the circuit’s action.

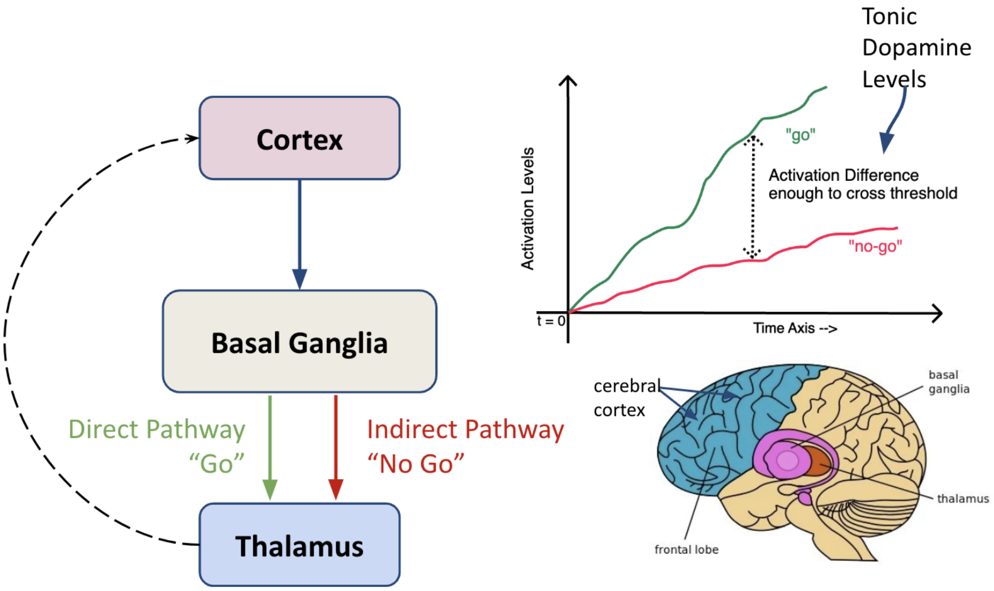


Figure 2.2: Simplified CBGT circuitry illustrates the interaction between brain sub-architectures: Cortex, Basal Ganglia, and Thalamus. The graph on the right shows that for the brain to decide to perform an action, the activation difference between the 'GO' and 'NoGo' pathways must surpass a threshold determined by dopamine levels in the Thalamus

In the context of the described structure, the information used for decision-making is generated in the cortex, which in turn increases or decreases the total activation of the "Go" and "NoGo" pathways for each action. Action selection for a given action is based on the relative activation of the "Go" and "NoGo" pathways—actions with a higher differential between the "Go" and "NoGo" pathways are more likely to be performed, see figure 2.2. In essence, the basal ganglia facilitate actions with a probability proportional to the activation difference between the "Go" and "NoGo" pathways. An action is performed once the activation difference for the action exceeds some criteria. Tonic dopamine levels in the brain increase the excitability of the "Go" pathways and decrease the excitability of the "NoGo" pathways, which influences the overall criteria for action selection, as well as reaction time.

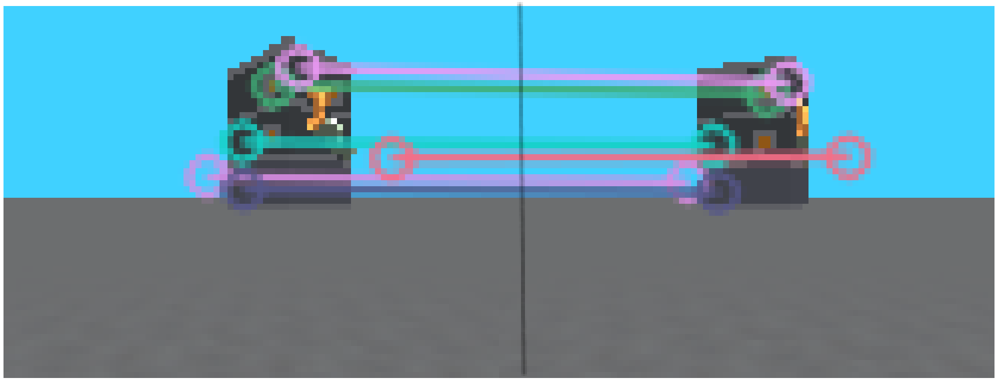


Figure 2.3: Feature Matching between two observations - left and right taken at separate times of the same building from different angles

2.2 Feature Matching in Loop-closure

Loop closure in Simultaneous Localization and Mapping (SLAM) is a crucial process wherein a robot identifies that it has returned to a previously visited location. This recognition allows the SLAM system to update and correct the map and the robot's trajectory by aligning the current sensor observations with earlier ones. As the robot navigates, small errors in its position estimates accumulate over time, leading to drift. By detecting loop closure, the SLAM system can correct these accumulated errors, ensuring the map remains accurate and consistent.

In practical terms, loop closure detection involves comparing current sensor data with stored data from past observations to find matches. Techniques like feature matching, scan matching, and visual recognition are commonly used for this purpose

Feature matching is a pivotal technique used in loop closure within SLAM to recognize previously visited locations. This method involves detecting, describing, and comparing distinctive features from sensor data (e.g., images or LIDAR scans) collected at different times to identify overlaps between current and past observations as shown in figure 2.3. Feature matching involves detecting and describing features in sensor data using algorithms like SIFT (Scale-Invariant Feature Transform), SURF (Speeded-Up Robust Features), or ORB (Oriented FAST and Rotated BRIEF), comparing these features to find correspondences, and using these correspondences to recognize previously visited locations.

2.3 Low Information Streaming Data

Low information streaming data refers to data streams that contain a relatively small amount of useful or relevant information relative to their volume. This type of data can pose challenges for processing, storage, and analysis because much of it might be redundant, noisy, or irrelevant to the tasks at hand. Noise and redundancy are the key problems in such a data. Much of the data can be repetitive or include noise, which does not contribute meaningful information. For instance, a temperature sensor in a stable environment might report the same or similar readings repeatedly, contributing little new information over time.

In practical applications, dealing with low information streaming data efficiently can lead to significant improvements in system performance and resource utilization. For example, in the context of Internet of Things (IoT) devices, effectively managing low information streaming data can extend battery life, reduce bandwidth usage, and improve the overall efficiency of data handling processes.

Another interesting example of low-information streaming data can be streams of partial images of a large environment arriving piece by piece. These images might be repetitive and overlapping, but we need to extract comprehensive information about the entire environment from these partially observed patches.

2.4 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are specialized neural networks designed to handle sequential data by maintaining a 'memory' of previous inputs through a hidden state that is updated at each time step. This architecture allows RNNs to process sequences of data such as time series, text, or image sequences, making them well-suited for tasks where context and order are crucial. A key feature of RNNs is their weight sharing across all time steps, which helps generalize patterns and reduces the number of parameters needed.

There are different types of RNNs, each designed to address specific challenges. Vanilla RNNs, the simplest form, often face issues like vanishing and exploding gradients, making them difficult to train on long sequences. Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs) are advanced variants that include mechanisms to better manage long-range dependencies. LSTMs use memory cells and gates to control information flow, while GRUs simplify this structure with two gates, making them more computationally efficient yet effective.

RNNs are widely applied in natural language processing (NLP) tasks such as language modeling, machine translation, and speech recognition, where understanding context over sequences is essential. They are also used for time series forecasting, like predicting stock prices or weather conditions, and in image and video processing tasks, such as captioning and event detection. Despite challenges like vanishing gradients, advancements like LSTMs, GRUs, and attention mechanisms have significantly improved RNNs' capability to handle complex, long-sequence data effectively.

Chapter 3

Approach

3.1 Network Architecture

In this section, we describe a neural network model, referred to as *CBGT-Net*, whose functionality aims to be analogous to the functionality of CBGT circuits in mammalian brains, as described in Section 2.1. Unlike traditional feed-forward networks, this model is designed to perform inference tasks based on a *stream* of observations, as opposed to a single input. In contrast to recurrent architectures, which maintain arbitrary latent embeddings as an internal state, the model maintains the total evidence accumulated over time in support of each possible decision. We note that the inference task described here differs from inference tasks performed by recurrent neural networks on *sequential* data: recurrent models generate a decision at a fixed point in time, or at the end of a sequence of known length; the inference task here requires the model to make a decision at an arbitrary point in time, in the presence of a (hypothetically) unending stream of data.

The model accepts as input a stream of observations at discrete time steps, denoted $\mathbf{o}_t$, from an environment (see Section 4.1.1). At each time step, the model produces a pair of outputs: an output vector, $\mathbf{y}_t$, corresponding to the inference task, and a binary decision variable, d_t, indicating if the model has accumulated sufficient evidence to make a decision. The output of the model should only be considered meaningful at the first time step that the decision variable indicates that the evidence criteria is satisfied, denoted as t_d. Thus, while the model generates a pair of outputs at each time step, only the decision at the first changepoint is considered meaningful.

For this paper, we explore the task of classifying observation streams, thus the output vector is interpreted as the probability distribution over possible categories. Figure 3.1 shows the basic structure of the model and the interaction of its core components—Evidence Encoder, Evidence Accumulator, and Decision Threshold Module—each of which is detailed below.

3.1.1 Evidence Encoder

Evidence Network learns a mapping from observations to evidence for or against each possible decision emulating the *"Direct & Indirect pathways"* of the CBGT Circuitry.

The *Evidence Encoder*, E_θ is a parameterized model responsible for mapping observations at each time step t, called $\mathbf{o}_t$, to an evidence vector, $\mathbf{e}_t$,

$$\mathbf{e}_t = E_\theta(\mathbf{o}_t) \tag{3.1}$$

where θ represents the parameters of the evidence encoder. The evidence encoder may be an arbitrary neural network model suitable for the modality of the observation data (e.g., convolutional neural network for images); we constrain its design to generate an output whose dimensionality and semantic interpretation are consistent with the available decisions. For example, for classification tasks, each decision category can have a single corresponding element in the evidence vector $\mathbf{e}_t$.

3.1.2 Evidence Accumulator

The *Evidence Accumulator* consists of a vector corresponding to the total evidence accumulated since the beginning of the input stream, $\mathbf{a_t}$,

$$\mathbf{a}_t = \mathbf{a}_{t-1} + \lambda \mathbf{e}_t \tag{3.2}$$

with a_0 assumed to be $\mathbf{0}$. The λ here is a weight parameter that can be determined heuristically or learned in different scenarios based on the context. This is discussed in detail in the section 4.1.3.

In addition, the accumulated evidence is mapped to the output vector using a suitable mapping function. For classification tasks, this simply involves calculating the *softmax* over the accumulated values

$$\mathbf{y}_t^{(i)} = \frac{exp(\mathbf{a}_t^{(i)})}{\sum_{j=1}^{K} exp(\mathbf{a}_t^{(j)})} \tag{3.3}$$

where (i) corresponds to the i^{th} element of a vector, and K is the number of decision categories.

The Accumulation Vector explicitly indicates the network's support for each decision; making deliberation transparent and mimicking the *"Go – NoGo Activation Difference"* of the CBGT Circuitry.

3.1.3 Decision Threshold

The *Decision Threshold* module is a component that is used to determine if the required evidence criterion has been satisfied and generates the decision variable, d_t. The Threshold Network inhibits decision until accumulated evidence exceeds the situation-dependent threshold mimicking the *"Tonic Dopamine Levels"* of the Thalamus in CBGT Circuitry. We explored two types of thresholding mechanisms -

Fixed Decision Threshold

This module is defined by a fixed threshold parameter, τ. For each time step, the decision variable is *true* if and only if at least one element in the evidence accumulator exceeds this threshold,

$$d_t = \begin{cases} true & \text{if } \exists i \text{ where } \mathbf{a}_t^{(i)} \geq \tau \\ false & \text{otherwise} \end{cases} \tag{3.4}$$

If the threshold is not exceeded, the model ingests additional data, allowing for additional evidence before making a choice.

At the initial instance when d_t becomes *true*, signifying the first time the threshold is crossed, the model makes a prediction by selecting the category associated with the highest value in the Evidence Accumulator's output vector y_t.

Dynamic Decision Threshold

Learning a Threshold Parameter. We start with the initial value of the dynamic threshold parameter D_v set to 0. The threshold loss is defined as:

$$T_v = \sigma(A_v - D_v) \tag{3.5}$$

where σ denotes the sigmoid function. A_v is the accumulation vector, D_v is the trainable threshold parameter.

The decision gate which determines whether a decision will be made at this time step 't' is determined by:

$$\text{Decision\_gate} = \text{if\_any}(T_v > 0.5) \tag{3.6}$$

Reward Formulation The reward formulation at any time t is given by:

$$\text{returns}_t = \text{rewards}_t + \gamma \cdot \text{returns}_{t+1} \tag{3.7}$$

where $rewards_t$ is defined as reward at any time instance t

$$rewards_t = \begin{cases} 30 & \text{if correct guess} \\ -30 & \text{if incorrect guess} \\ -30 & \text{if timeout} \\ 0 & \text{if no guess} \end{cases}$$

Threshold Loss Term. The threshold loss term is finally defined as:

$$-\sum (\log(T_v) \cdot \text{returns}_0) \tag{3.8}$$

where T_v is the threshold vector defined in equation 3.5 and $returns_0$ is cumulative reward from 0 to time $'t'$.

Rationale for Learning a Threshold

The rationale behind learning a threshold is based on the fact that different observation inputs may require different optimal threshold values. For instance, in image classification tasks, varying the patch size of observations can result in different convergence values for the threshold parameter. Smaller patches capture lesser details and could also introduce more noise thus, needing a higher threshold to make a confident prediction whereas larger patches might provide more contextual information and less noise thus potentially needing lower decision threshold. Thus, the optimal threshold value that maximizes accuracy can vary depending on these input characteristics.

Trying different fixed threshold values result in varying optimal values for different scenarios. By "optimal," we mean achieving maximum accuracy in the minimum number of time steps or observations. This suggests that rather than trying an infinite number of potential threshold values, it is more viable to learn the decision threshold value dynamically. Thus, automatically learning the threshold eliminates the need for manual tuning, saving time and effort. And a dynamically learned threshold can generalize better across different scenarios, datasets, and conditions, providing robust performance.

However, it is important to note that learning a dynamic threshold introduces additional complexity to the training process. Compared to a fixed threshold scenario, training with a dynamic threshold typically requires more epochs to converge. This is because the model needs to simultaneously learn:

- The encoder parameters that map observations to evidence for different choices.
- The optimal threshold value that should be applied for decision-making.

Due to these challenges, this thesis does not present any successful results for CBGT-Net trained with a learnable threshold parameter. We suggest this area for exploration in future research. We believe that with careful tuning of losses, the evidence encoder and threshold parameters can be trained simultaneously.

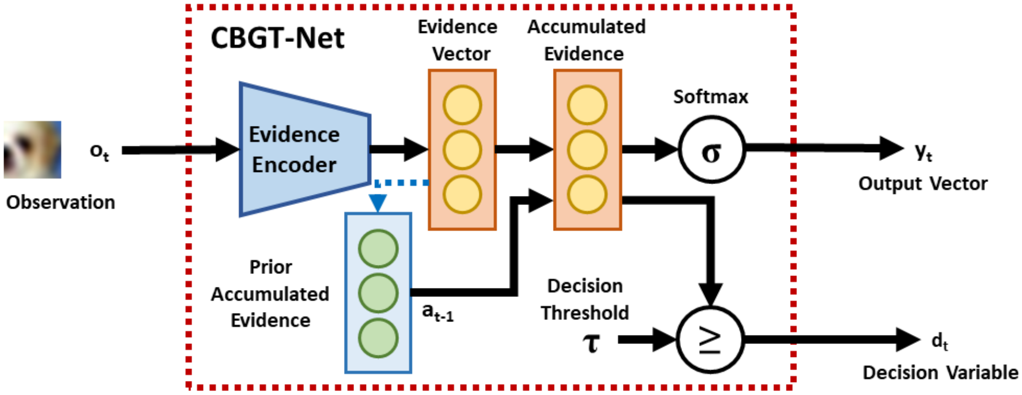


Figure 3.1: Main components of the CBGT-Net architecture.

Chapter 4

Experiments

4.1 Experimental Setup

4.1.1 Environments

To evaluate the described approach, we developed a set of environments to generate streams of information for use as input to the model, where individual streams are conditioned on a target category. Environments were constructed around publicly available datasets used for image classification.

We denote a single stream of information generated by the environment as an episode. At the beginning of a given episode, the environment selects an image at random from the dataset and its corresponding target category. At each time step, the environment extracts a square patch of pixels from the environment at a random location in the image. The extracted patch is zero-padded to produce an image with the same dimensionality as the original dataset, and in such a manner that the patch is centered on the image. This approach ensures that the qualitative amount of information present in a single observation in the episode can be controlled (through the size of the patch), and positional information regarding the observation is removed (through centering of the patch). The task of the model is to infer the target category of the selected image based on the stream of observations.

We constructed environments based on two image datasets:

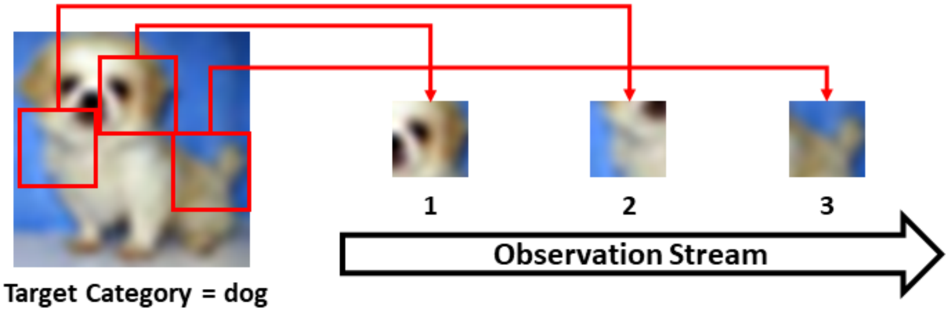


Figure 4.1: Example episode from CIFAR-10 environment: a sequence of three patches from an image in the “dog” category.

MNIST Environment

The MNIST dataset [10, 11] consists of images of handwritten digits, with ten target categories corresponding to each digit (i.e., 0 – 9). Each image is greyscale and 28x28 pixels in size and contains a single handwritten digit. The dataset contains a total of 60,000 training images and 10,000 test images. Using this dataset, we constructed environments which generated patches of size 5x5, 8x8, 10x10, 12x12, 16x16, 20x20.

CIFAR-10 Environment

The CIFAR-10 dataset [9] consists of 50,000 color images in training data and 10,000 color images for testing in 10 categories. Each image is 32x32 pixels in size. The ten image categories that the images belong to are *airplane*, *automobile*, *bird*, *cat*, *deer*, *dog*, *frog*, *horse*, *ship*, and *truck*. The environments which were constructed using this dataset generated patches of size 5x5, 8x8, 10x10, 12x12, 16x16, 20x20.

MiniWorld Environment

MiniWorld [5] is a minimalistic 3D environment simulator designed for reinforcement learning and robotics research. Miniworld is being maintained by the Farama Foundation [5]. It can simulate environments containing various objects such as buildings, roads, and rooms. The simulator is entirely written in Python and is easily modifiable and extendable.

Using the MiniWorld codebase, we generated a search and rescue environment. This environment features a cityscape with road networks and buildings, some of

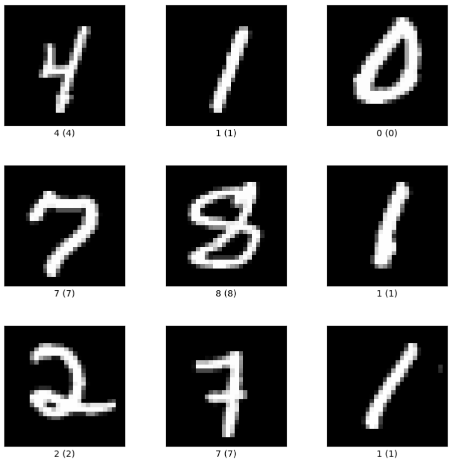


Figure 4.2: MNIST Dataset

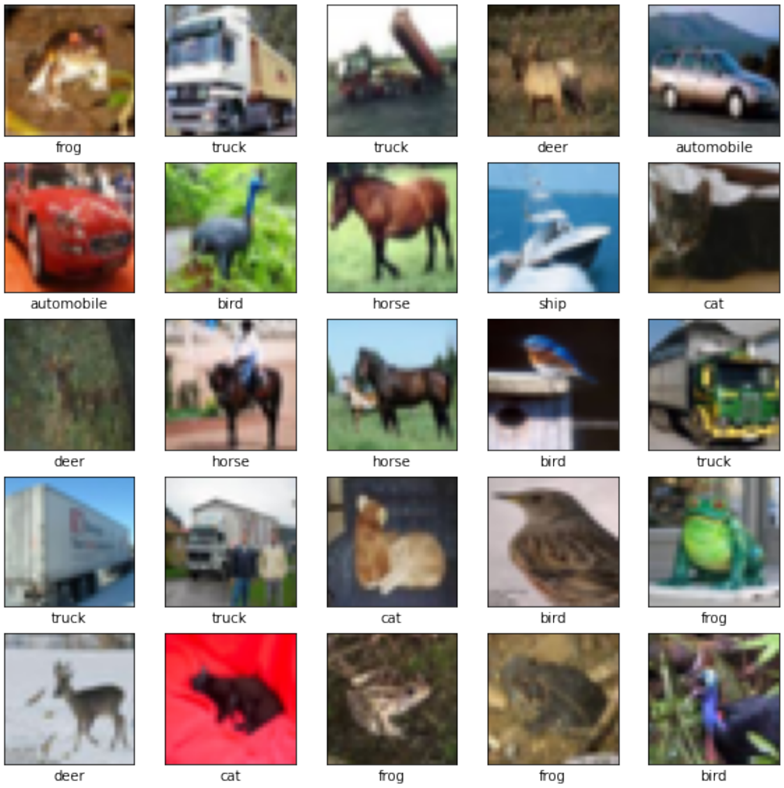


Figure 4.3: CIFAR10 Dataset

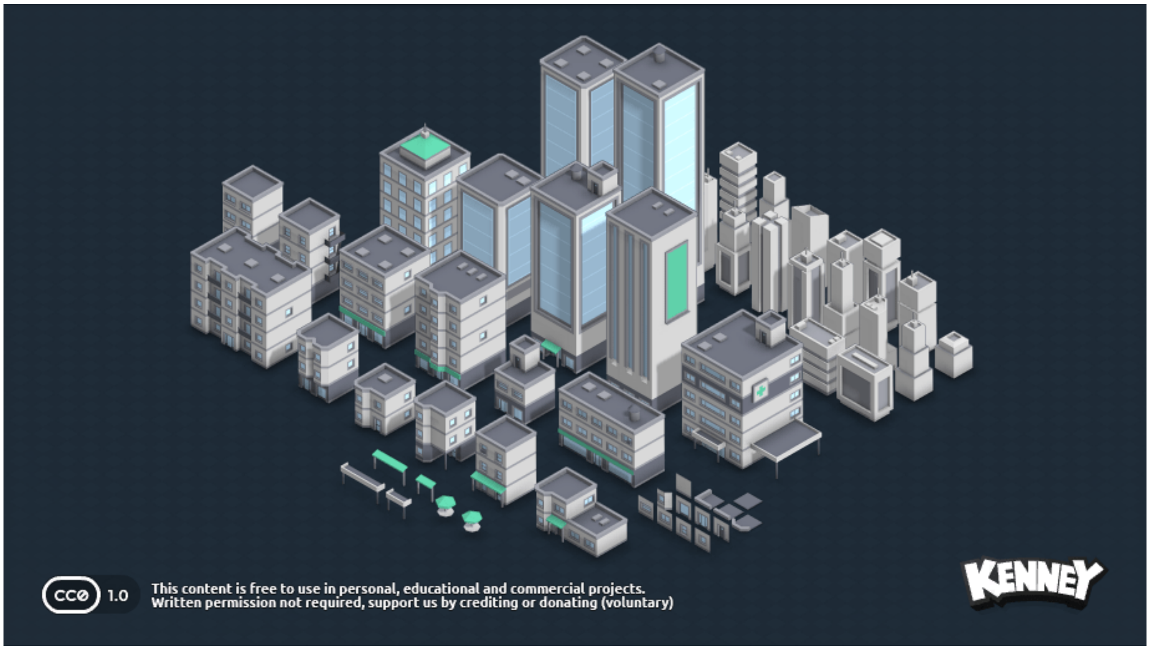

CC0 1.0
This content is free to use in personal, educational and commercial projects.
Written permission not required, support us by crediting or donating (voluntary)
KENNEY

Figure 4.4: City Kit (Commercial) 3D Building Asset by Kenney

which are at different levels of fire. An autonomous rover navigates these diverse maps along random paths to collect a dataset of buildings on fire.

- **Building Assets** - We modified the MiniWorld Environment to generate a search rescue simulation with various types of building assets imported from Kenney 3D City Kit [8] for buildings. The dataset has 35 different kinds of buildings in 3D mesh format with customizable colors and textures of the buildings.
- **Procedural Generation** - Procedural generation is a technique in computer graphics, game development, and content creation that uses algorithms to automatically generate data and content. This method leverages mathematical formulas, randomization, and predefined rules to create complex structures, landscapes, textures, and even entire game levels without the need for manual design.

  We utilized procedural generation techniques to create randomized search and rescue scenarios. This includes randomizing the color, texture, orientation, and position of buildings, as well as the path an agent takes to traverse the scene.

Additionally, the texture of the buildings can be modified.

- **Dataset Generation and Frames Scrapping** - After randomly generating different scenarios, we extract a dataset consisting of a training set and a test set. The training set includes 750 buildings, each with 29 views from different directions. Similarly, the test set contains 250 buildings, each with 29 views. We have defined five levels of fire: [no fire, stage 1, stage 2, stage 3, stage 4] based on the intensity, temperature and time as shown in figure 4.6. The dataset is balanced across all classes, with the training set having 152 buildings per label and the test set having 50 buildings per label. The stages of fire and their descriptions are detailed in Table 4.1.

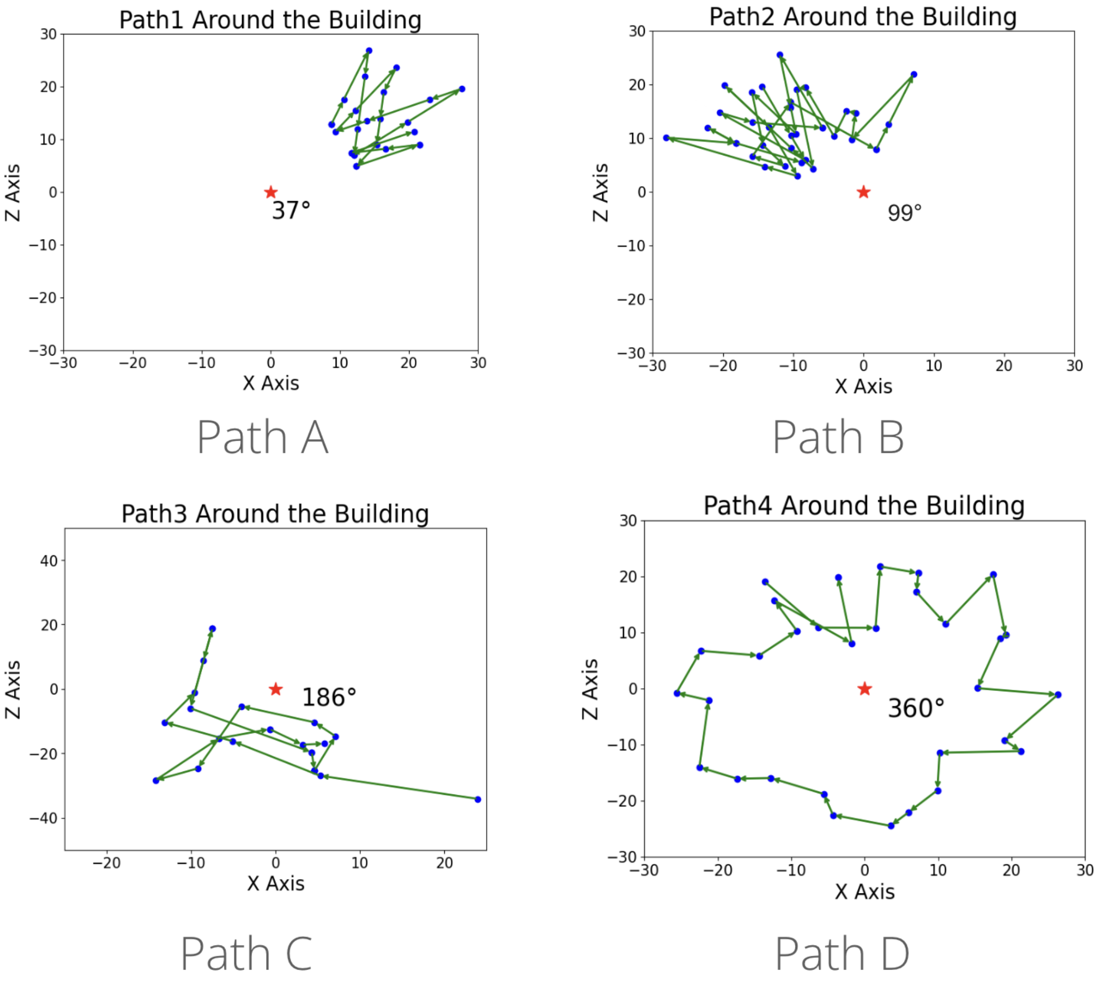
Label	**Window Color**	**Flame**	**Smoke**	**Wall Texture**
No Fire	Any	False	False	Normal
Stage 1 Fire	Light orange	False	True	Normal
Stage 2 Fire	Orange	True	True	Grayish
Stage 3 Fire	Dark Orange	True	True	Blackish
Stage 4 Fire	Black	False	True	Dark Black

Table 4.1: Fire Stages and Their Characteristics

4.1.2 Evidence Encoders

For evaluation, we utilized existing network architectures as evidence encoders in the CBGT-Net. For experiments involving the MNIST Environment, we utilized Lenet-5 [10] as the evidence encoding network. Lenet-5 (figure 4.12) is a convolutional neural network consisting of seven total layers—two convolutional layers interleaved with two subsampling layers, followed by two fully connected layers and a softmax classification layer. Lenet-5 is an appropriate and effective choice for the MNIST digit dataset due to its relatively simple architecture designed by Yann LeCun et al. for digit recognition tasks, particularly handwritten digits, making it highly relevant to the MNIST dataset. It serves as a good baseline model. Its performance on MNIST can be used as a benchmark to compare with more complex models, providing a reference point for improvements.

For experiments involving the CIFAR-10 and MiniWorld Environment, we utilized a ResNet style (figure 4.13), residual architecture [6]. The model consists of an initial



Path A

Path B

Path C

Path D

Figure 4.5: Different Paths of the Autonomous Agent around the Building marked as Star. The blue dots represent the camera location in the x-z plane from where the image of the building is captured as the agent moves around.

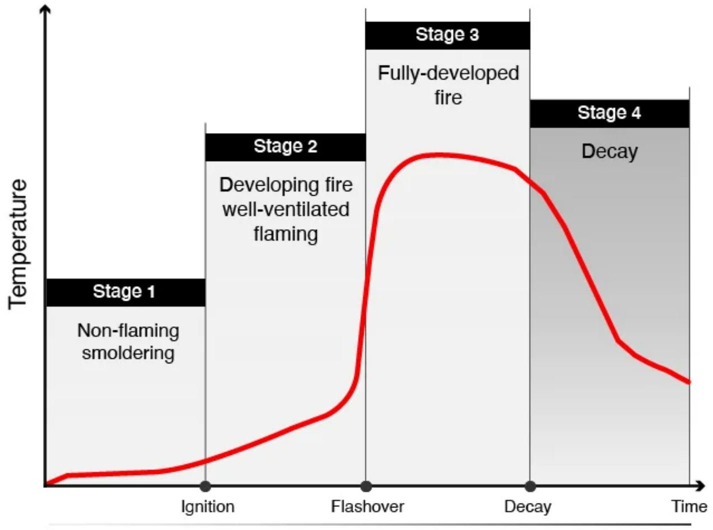
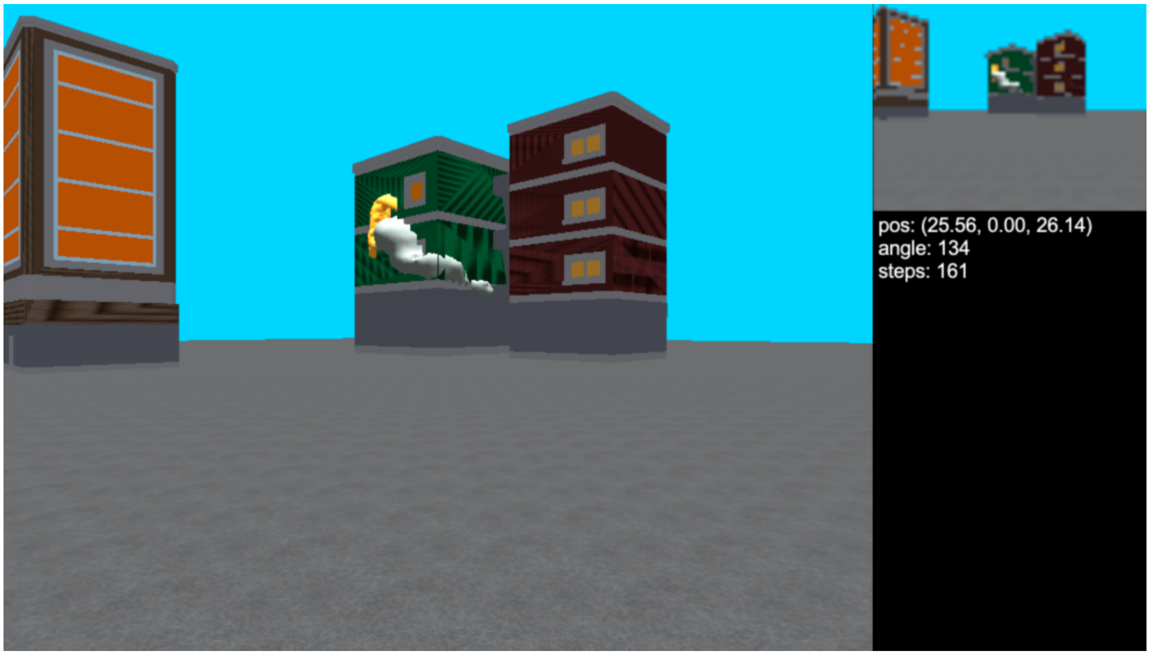


Figure 4.6: 4 Stages of Fire

convolutional layer and batch norm layer, followed by six "blocks" of two convolutional layers, followed by a fully connected layer, an average pooling layer, and a softmax classification layer. Each block in the model is designed to maintain the size of the generated feature map and includes a shortcut connection from the input of the block to the output so that each block learns to compute a residual, rather than general, mapping from input to output. Additionally, the network downsamples the size of the feature map after every pair of blocks. ResNet-18 utilizes residual blocks with shortcut connections. The residual connections help mitigate the vanishing gradient problem, making it easier to train deeper networks and improving convergence. ResNet18 is known for its high performance on various image classification tasks, including CIFAR-10. It consistently achieves high accuracy, making it a reliable choice for this dataset.

We experimented with different activation functions—softmax, sigmoid, and tanh—for the final linear layer, which maps hidden features from the encoder architectures to the output evidence vector, representing the number of classes. However, the softmax activation consistently demonstrated the best performance. Consequently, we opted to employ softmax activation on top of the evidence output. This decision is logical because it reflects a scaled confidence level for each class based on the evidence provided by a new observation. In this context, if an observation offers



```
pos: (25.56, 0.00, 26.14)
angle: 134
steps: 161
```

Figure 4.7: Sample City View

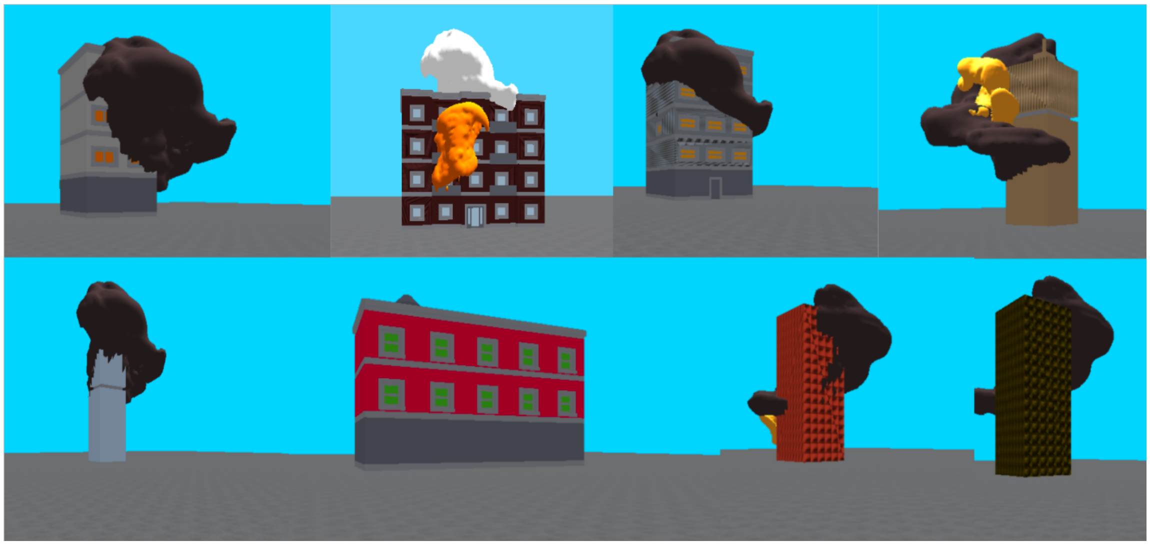


Figure 4.8: Buildings on Fire

Figure 4.9: MiniWorld: Sample City Views with Buildings on Fire

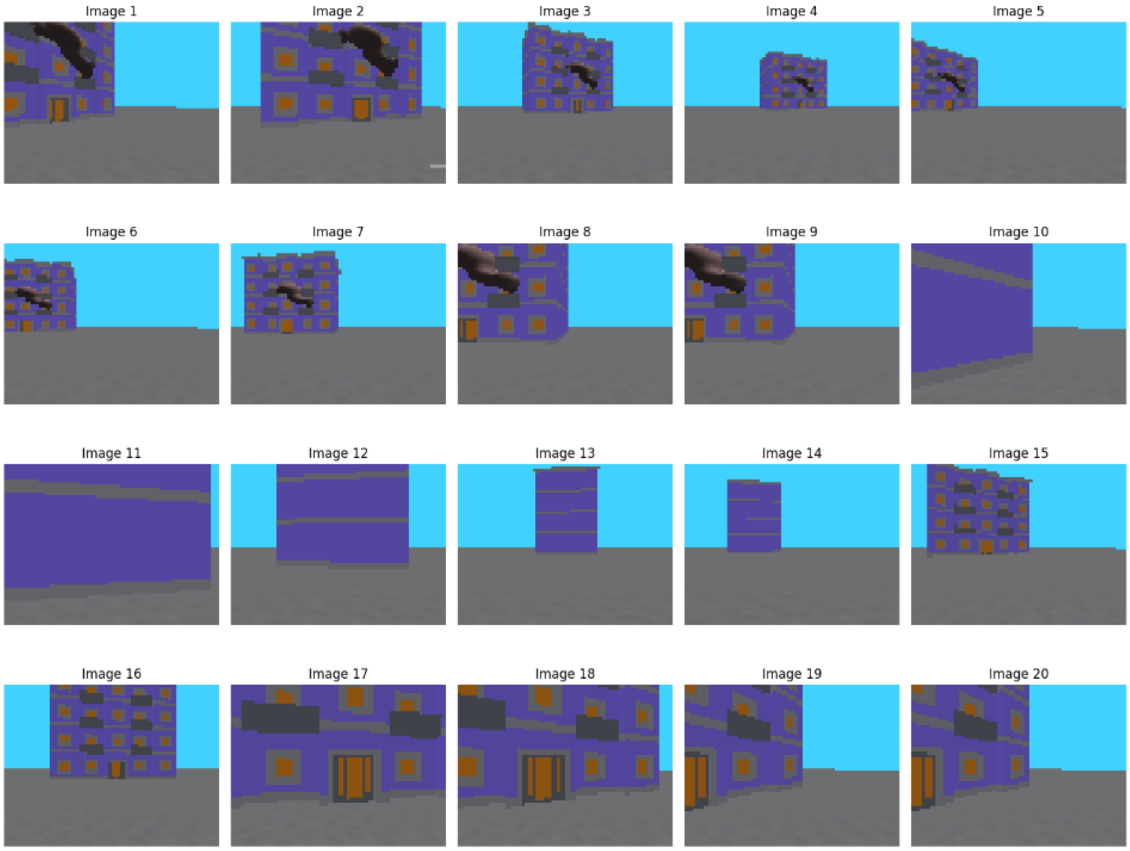


Figure 4.10: MiniWorld Dataset Example 1

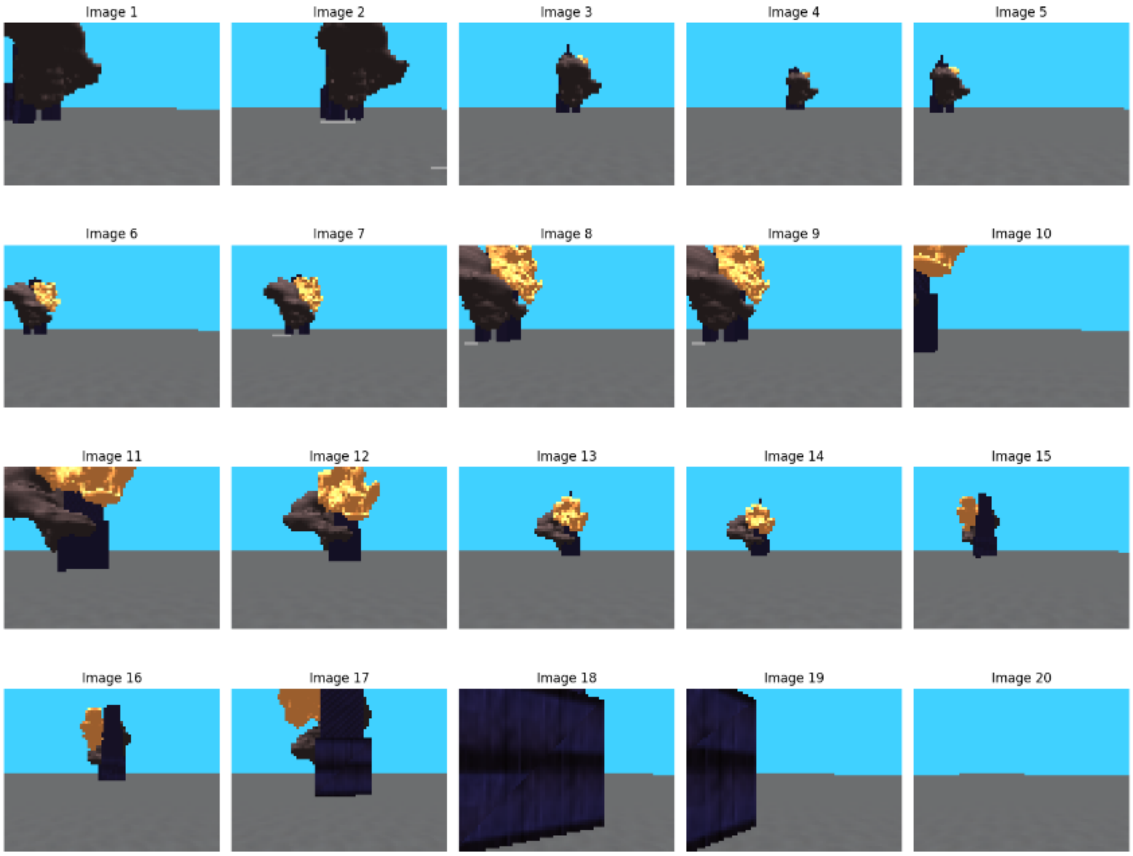


Figure 4.11: MiniWorld Dataset Example 2

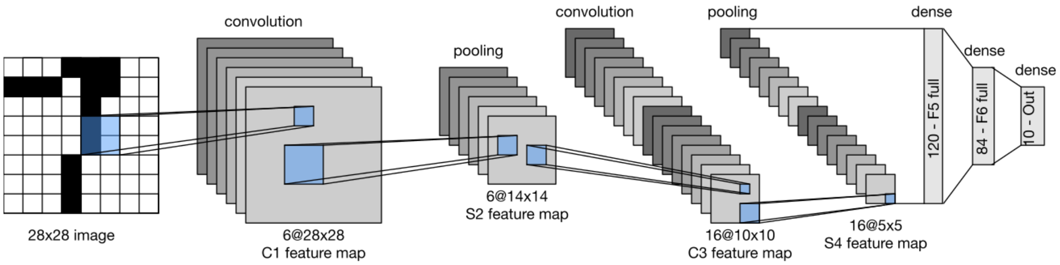


Figure 4.12: LeNet5 Architecture

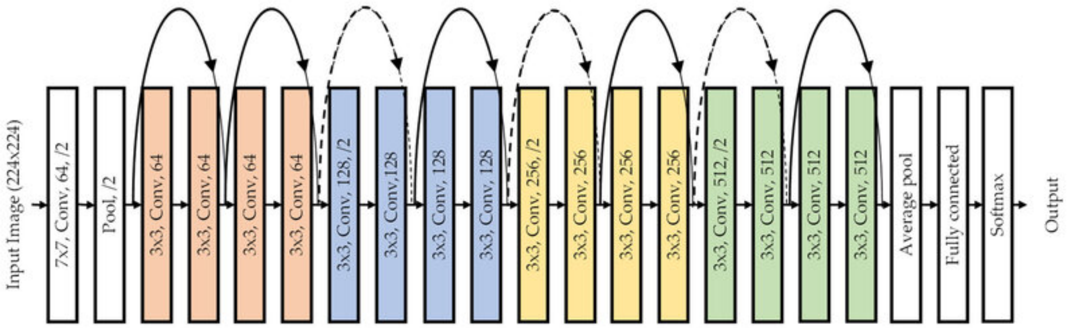


Figure 4.13: ResNet18 Architecture

no information regarding a specific class, it will not contribute to its accumulation, ensuring a sensible interpretation of the output.

4.1.3 Context-aware Accumulator

The Accumulator Module in the CBGT-Net architecture is responsible for the evidence accumulation process, where the evidence collected so far is combined with newly available evidence at any time t. This process is called context-aware evidence accumulation because the new evidence is accumulated based on the previously seen evidence.

In the MNIST and CIFAR10 environments, the sampled patches are considered equally important for classification and are thus weighted equally. Given how the patches are generated—by selecting a pixel value (x, y) in the image and extracting a patch of size $(patch\_sz, patch_{)}$ with (x, y) as the top left corner—it is less likely to encounter highly overlapping patches. However, if the CBGT-Net is presented with the same observation patch multiple times, it will accumulate the evidence in the same manner each time, without recognizing that it has seen these patches before. This means that seeing the same input multiple times could cause the model to cross the threshold without encountering any new information, which is a logical limitation of the equally weighted, context-unaware accumulator.

While this method of accumulation might be effective for simpler environments like MNIST and CIFAR10, it poses challenges in the MiniWorld environment. Here, images of buildings are captured by a rover traversing a path around the building.

It is likely that several consecutively captured images will have identical features. When accumulating evidence, these images will count the evidence from the same views multiple times, potentially causing a false threshold crossing. To address this issue, we propose multiple methods of context-aware accumulation.

- **Simple Accumulator**- This is the standard accumulation process, where each new evidence vector is added to the accumulation vector with a weight of 1. This method disregards any context information or potential overlap during accumulation. The experiments in the MNIST and CIFAR10 environments were conducted using this simple accumulator approach.
- **Camera-pose based Weighting** - This context-aware accumulation approach is deployed in the MiniWorld environment. In this setup, each view of a building includes the camera's position information. Therefore, when a new observation is made, it consists of both an image and the camera's pose information relative to the center of the building. The evidence encoder encodes the building image into an evidence vector, but unlike simple accumulation, this vector is not directly added to the accumulator vector. Instead, a heuristic is used to weigh the new evidence based on all previously seen observations.

  In this heuristic, we discretize the 2D space around the building's center into 25 non-overlapping sections called p as 5x5 vector. The details of these sections can be seen in the accompanying image 4.14. The space is divided into 5 sectors of 72 degrees each, and each sector is further divided into 5 segments based on the distance from the center. We track where the image observations have been captured so far. When a new image is received, if it is captured from a camera pose that has already been seen in the previously observed images, it is weighted as 0. However, if it is captured from a new segment, we weigh it as shown below.

  Let P^t be the pose accumulator at time t, represented as a 5x5 matrix initialized with all zeros, which stores a 1 at (i, j) if an image has been seen from camera pose location $p_{i,j}$. Thus, if at time t, the camera's position lies in $p_{i,j}$, where i is the sector and j is the segment within the sector that has not been covered so far, then

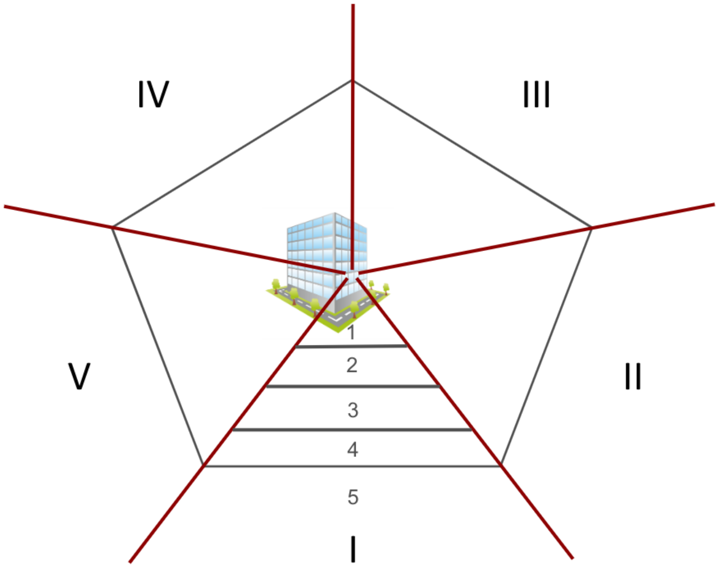


Figure 4.14: Discretisation of 2D Space around the Building

$$w_t = \begin{cases} 0 & \text{if } P^t_{i,j} = 1 \\ 1 - \frac{1}{5}\sum_{k=0}^{4} P^t_{i,k} & \text{otherwise} \end{cases}$$

- **SIFT Features-based Accumulation** - In this method, we compute the number of matching keypoints from SIFT features in pairs of images and then discount the weight for a new image based on the number of matching keypoints with the images observed so far. If $m_{i,j}$ if the matching key points between images seen at time t = i and t = j then we define the weight for an image seen at time t as follows -

$$w_t = 1 - \max_{0 \leq i \leq t-1} \left(\frac{m_{i,t}}{m_{t,t}} \right)$$

SIFT (Scale-Invariant Feature Transform) [12] is a widely-used feature extraction method in computer vision and image processing. It identifies and describes local features in images, making it highly effective for tasks such as object recognition, image stitching, and 3D reconstruction.

4.1.4 Baselines

For each experiment, we compare our approach with multiple baselines. As with the CBGT-Net, all baseline models are trained to minimize the cross-entropy loss given in Equation 4.1.

Single Patch Evidence Encoder

For each experiment, we train the evidence encoder used in the CBGT-Net to predict the target category of an image from a single patch. This baseline serves as a benchmark for evaluating the encoder's capability to independently classify individual data patches. Furthermore, it helps us assess how effectively the model's accuracy improves when evidence is accumulated from multiple observed patches.

LSTM Model

For each experiment, we train a model in which the output of the evidence encoder is connected to a Long Short Term Memory (LSTM) layer [7]. The LSTM layer has ten memory cells and is provided with a sequence of evidence encoder outputs from observations from the environment. Models were trained on sequences of varying length, in order to compare model performance with the CBGT-Net's decision times at each decision threshold.The model outputs the predicted category at the final time step.

Vision Transformer Model

The Vision Transformer (ViT) is a powerful and versatile model for image classification that leverages the strengths of the Transformer architecture. By treating images as sequences of patches, it effectively uses self-attention mechanisms to capture both local and global dependencies, achieving state-of-the-art performance on various benchmarks.

For more technical details, refer to the original ViT paper: ”An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale” by Alexey Dosovitskiy et al [3].

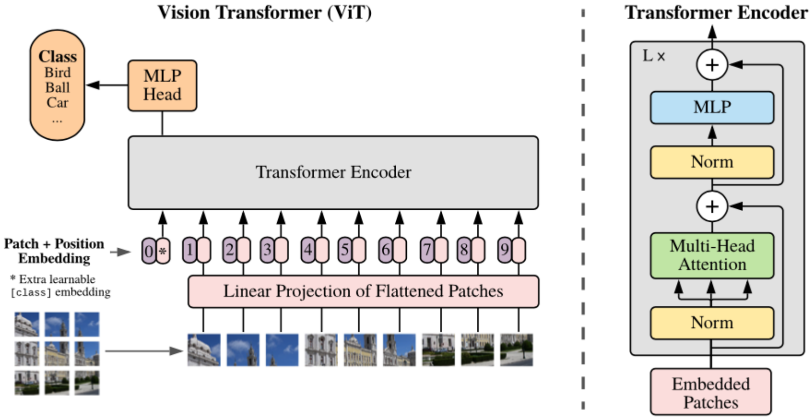


Figure 4.15: Vision Transformer Architecture [3]

ViT as shown in figure 4.15 has demonstrated superior performance on image classification benchmarks like ImageNet, often surpassing traditional convolutional neural networks (CNNs) when pre-trained on large datasets. Both ViT and CBGT-Net process images in patches, but ViT's configuration requires specifying image and patch sizes during training, enforcing a fixed number of patches. This makes it challenging to handle online input streams of variable lengths, similar to the limitations faced by LSTM-based models, which also require fixed-length input sequences during training.

Pre-trained ViT Model details. For the baseline comparison of CBGT-Net on the MiniWorld dataset, we are utilizing Google's 'vit_base_patch16_224_in21k', a pre-trained Vision Transformer (ViT) model from the Hugging Face's timm (short for ”PyTorch Image Models”) library. This model has been pre-trained on the ImageNet-21k dataset, which consists of 14 million images across 21,843 classes, at a resolution of 224x224. In the ViT model, images are processed as sequences of fixed-size patches (16x16 pixels) that are linearly embedded. This pre-training allows the model to learn a robust internal representation of images, which can be leveraged for various downstream tasks. For our experiments, we fine-tune the classification head of the ViT model on the MiniWorld dataset to classify the image sequence as patches of a larger image, thereby utilizing the pre-trained features for effective classification.

Patchification for ViT Baseline. This ViT baseline uses a sequence of 9 sequential views, starting from a randomly chosen initial view, and stacks them into a 3x3 grid. By padding and reshaping the original 60x80x3 images, we create a 224x224x3 input image. We then fine-tune the pre-trained ViT model to classify these input sequences of 9 images combined into one.

ViT Baseline Training Details. The ViT model is fine-tuned for 200 epochs with a batch size of 256, utilizing Cross Entropy Loss. Only the final linear layer (MLP Head), which maps the hidden state to the output feature space, is fine-tuned. We employ the Adam optimizer alongside the StepLR learning rate scheduler.

4.1.5 Training Details

During training, at the beginning of each episode, the environment (MNIST or CIFAR10) chooses an image and its corresponding label (the ground truth for prediction) from the training data at random. Now for every step within the episode, the environment randomly samples a patch of size (patch_size, patch_size, 3) from the image and pad it with 0 padding to generate an image of size same as the original images in the dataset and output it as o_t i.e. observation at time t which is then fed to the CBGT_Net.

In the MiniWorld environment, each episode begins with the environment rendering a random building in one of five possible stages of fire. The autonomous rover starts capturing images of the building from a random side and then moves around it, taking pictures from various angles and distances. Consequently, at every step within the episode, these images—captured from different camera positions on the rover—constitute o_t, the observation at time t, which is then fed into the CBGT_Net.

For training purposes, we utilize the cross-entropy between the output vector of the CBGT-Net and the target category at the decision time, t_d, as the objective function to minimize,

$$\mathcal{L}_{CE} = -log(\mathbf{y}_{t_d}^{(T)}) \tag{4.1}$$

where T is the index of the target category to be classified.

The model was trained using stochastic gradient descent with a learning rate of 1e-3 and batch size of 2048 for MNIST and 256 for CIFAR10 as it was difficult to fit a large batch size for an architecture as big as ResNet18 (with approximately 11M parameters) compared to LeNet5. For the Miniworld Environment a batch size of only 64 was used since the images input was 60x80x3 which is larger than the small patch images (from 5x5x3 to a maximum of 20x20x3) in MNIST and CIFAR10 scenarios.

The Adam optimizer was employed to minimize the cross-entropy loss. The maximum number of steps permitted within an episode was determined according to the selected threshold value, T_d. Specifically, it was configured to be 10 times the threshold value (T_d) plus 1. This setting ensures that the number of steps allowed exceeds the count required for a random model to surpass the threshold and make a decision. The rationale behind this choice lies in the fact that with each step, the evidence being accumulated follows a softmax output, leading to a monotonically increasing accumulator vector. Consequently, there exists mathematical assurance of crossing a threshold before reaching $10 * T_d + 1$ steps within an episode.

4.1.6 Evaluation Measures

To evaluate our models, we calculate the accuracy, average decision time, and the number of training episodes required. Accuracy measures the percentage of correct predictions the model makes when tested with a batch of episodes. Average decision time, on the other hand, quantifies the average number of steps taken before the model reaches a decision. In simpler terms, average decision time measures how much input data the model needs to see before making a confident prediction i.e. before it crosses the predefined threshold.

For each model and environment, we calculated the number of training examples required for the model to converge. During training, validation accuracy was calculated after every two training epochs (i.e., after training with 1,024 episodes). We performed exponential smoothing on the validation accuracy, with a smoothing factor of $\alpha = 0.995$. The normalized root mean standard deviation (NRMSD, i.e., standard deviation normalized to the mean) of the validation accuracy was calculated at each step using a window over the previous 100 steps. Training is considered converged when the NRMSD is below an empirically determined threshold of 0.0015.

4.2 Results

In this section we will delve into a comprehensive analysis of the performance and efficacy of the CBGT-Net model in handling complex decision-making tasks, particularly focusing on image classification from data streams. By evaluating the model's accuracy, decision times, and training efficiency across various experimental setups, this section aims to provide insights into the model's robustness and adaptability in real-world scenarios. Through a detailed examination of the model's performance metrics and comparison with existing LSTM-based sequential models, and transformer based baseline this section elucidates the CBGT-Net's superiority in terms of accuracy, data efficiency, and resilience to diminishing information in observations. Furthermore, the section highlights the model's ability to make decisions based on accumulating evidence, as opposed to fixed time intervals, showcasing its flexibility and adaptability in dynamic decision-making environments.

Figures 4.16 and 4.17 compare the performance of the CBGT-Net and baseline models for MNIST and CIFAR-10 Environments, respectively. Each figure compares the inference accuracy of the models based on the amount of information in each observation (i.e., patch size), and the number of observations made. For the CBGT-Net results, markers indicate the average decision time for decision thresholds in the range of 1 to 5; results for LSTM models were extracted from models trained with a sequence length comparable to the CBGT-Net decision times. Additionally, the accuracy of evidence encoders trained to categorize a single patch is provided for each case.

Figure 4.18 compares the performance of the CBGT-Net and baseline models for the MiniWorld Environment. Figure 4.18 compares the inference accuracy of the models based on the number of observations made. It is anticipated that as the decision time, representing the number of time steps and consequently the observations the model processed before reaching a decision, increases, there will be a corresponding rise in accuracy. This is clearly observed by the monotonic increasing nature of most of the plots. For the CBGT-Net results, markers indicate the average decision time for decision thresholds in the range of 1 to 5; results for LSTM and ViT models were extracted from models trained with a sequence length comparable to the CBGT-Net decision times i.e nearest whole number above corresponding CBGT-Net average decision times. Additionally, the accuracy of evidence encoders trained to categorize a single image of the building is provided for each case denoted by a red horizontal line. This baseline would be independent of decision time as we do not feed in a sequence but a single image to predict the stage of fire here similar to the single patch baseline in MNIST and CIFAR10 environments.

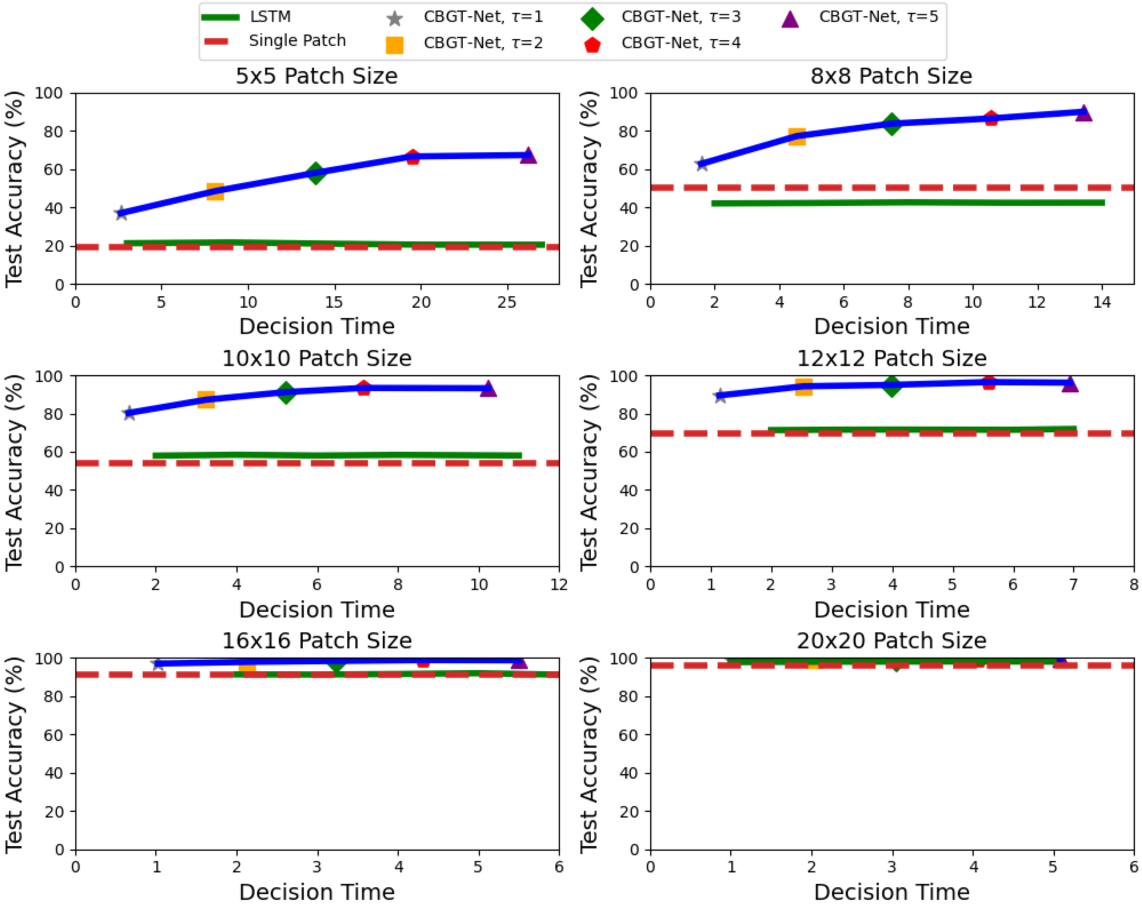


Figure 4.16: Inference accuracy of the CBGT-Net and baselines as a function of decision time for the MNIST Environment. Markers on the CBGT-Net results indicate the *average* decision time for models with a given threshold value. LSTM models were trained with sequence lengths corresponding to the nearest value above corresponding CBGT-Net decision times.

In general, for MNIST and CIFAR10 environments, the CBGT-Net outperforms both the LSTM and single patch baselines across decision times, with the exception that the LSTM models outperform the CBGT-Net models on the CIFAR-10 Environ-

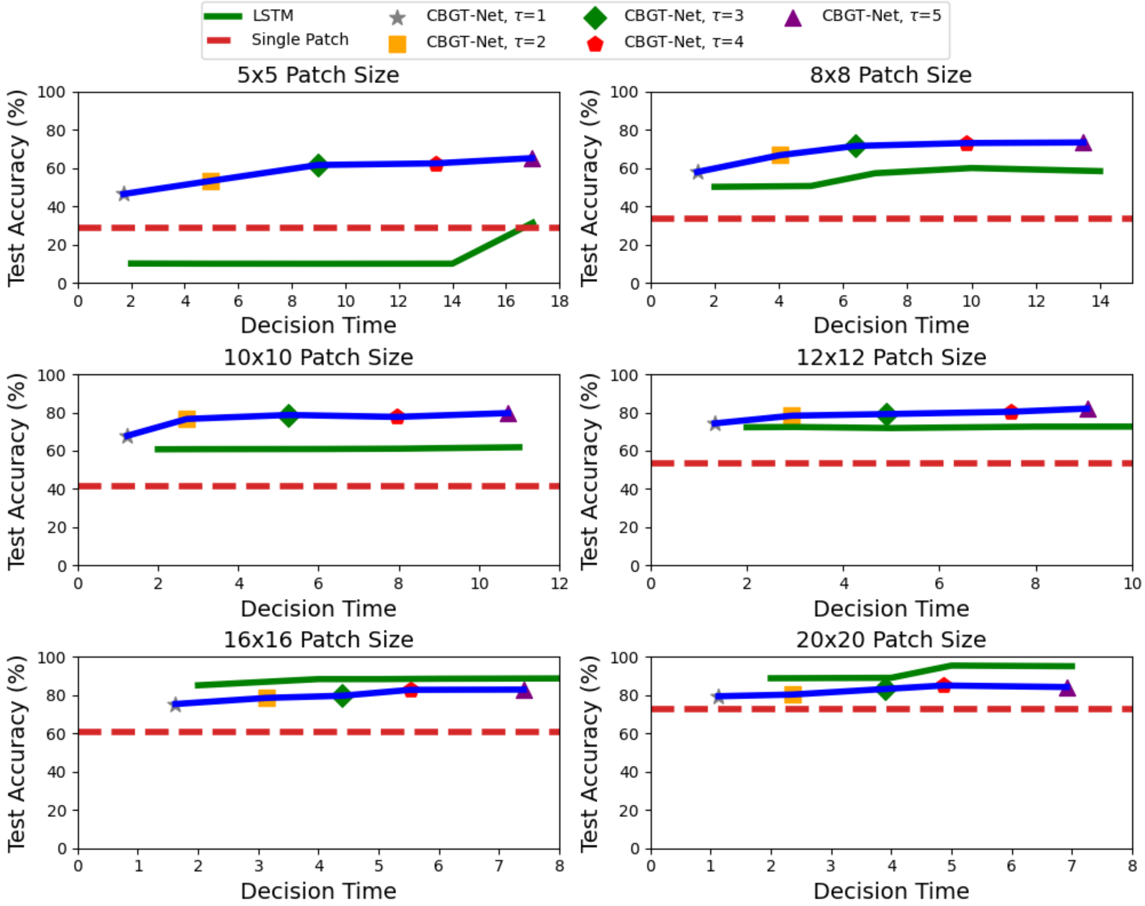


Figure 4.17: Inference accuracy of the CBGT-Net and baselines as a function of decision time for the CIFAR-10 Environment. Markers on the CBGT-Net results indicate the *average* decision time for models with a given threshold value. LSTM models were trained with sequence lengths corresponding to the nearest value above corresponding CBGT-Net decision times.

ments with 16x16 and 20x20 patch sizes. For the MNIST Environments, the LSTM models have roughly the same accuracy as the single patch models, demonstrating that this model was unable to learn to leverage the multiple observations to improve performance for this environment. The CBGT-Net, on the other hand, not only demonstrates an improvement in performance as sequence length increases, indicating that the model benefits from additional evidences to a certain extent, but also shows significant robustness when each observation's patch size decreases.

For the CIFAR-10 Environments, the LSTM models demonstrate the ability

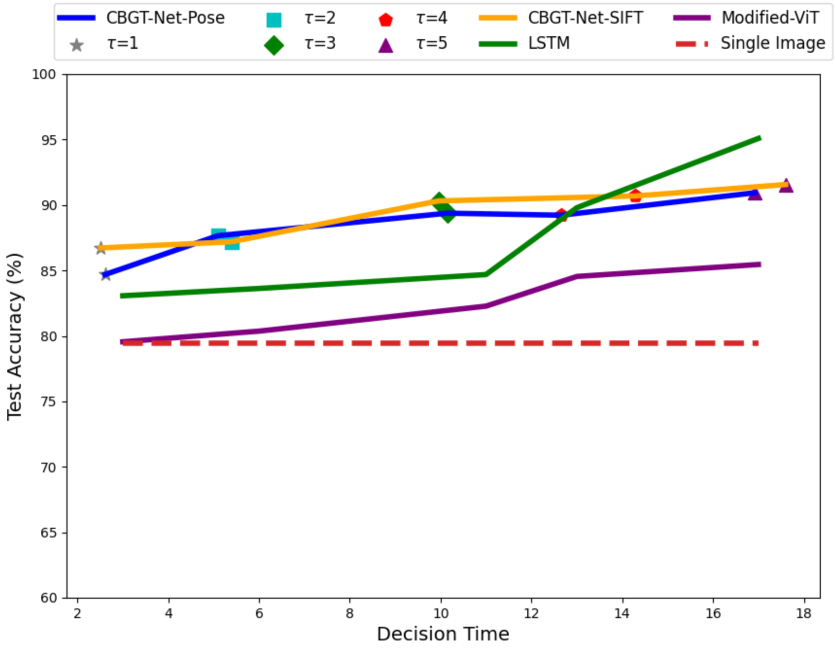


Figure 4.18: Inference accuracy of the CBGT-Net and baselines as a function of decision time for the MiniWorld Environment. Markers on the CBGT-Net results indicate the *average* decision time for models with a given threshold value (indicated in the label). LSTM and ViT models were trained with sequence lengths corresponding to the nearest value above corresponding CBGT-Net decision times.

to outperform the single patch baseline, demonstrating its ability to improve its performance with multiple observations (with the notable exception of the 5x5 patch size environment). For these environments, the CBGT-Net shows improvement over the single patch models similar to the MNIST environments; the performance margin between the CBGT-Net and LSTM models for environments using smaller patch sizes also demonstrates the CBGT-Net's improved robustness to reduced information in each observation when compared to both the LSTM and single patch baselines.

Figure 4.21 shows the average decision time for the CBGT-Net for different decision thresholds and patch sizes for the MNIST Environments and CIFAR-10 Environments. As can be seen, the required decision time increases as either the decision threshold increases or the patch size decreases. For larger patch sizes in the MNIST Environments (i.e., 16x16 and 20x20), the decision time is roughly equivalent to the decision threshold—this relationship indicates that the generated evidence vector is, on average, producing a maximum value (i.e., 1) for the target category, and that a single patch at these sizes is likely sufficient for categorization purposes.

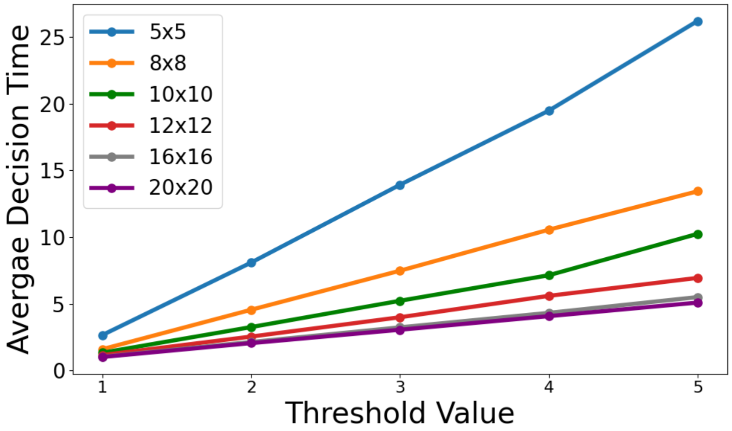


Figure 4.19: MNIST Environments

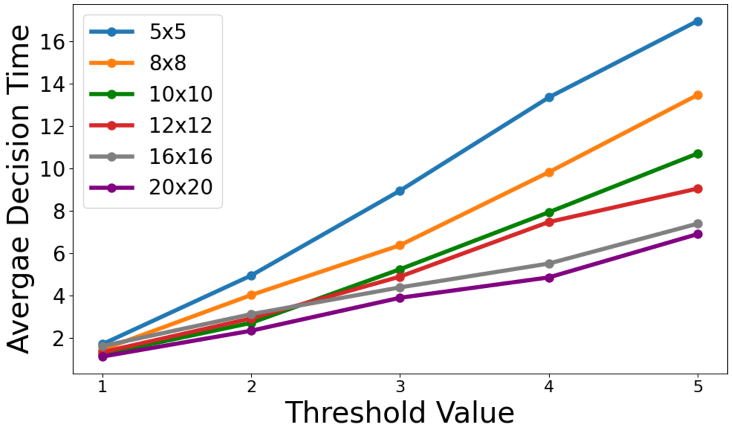


Figure 4.20: CIFAR10 Environments

Figure 4.21: Average Decision Time taken by CBGT_Net trained at different threshold values (τ) on MNIST (Fig. a) and CIFAR10 (Fig. b) Environments for different patch size observations

For the MiniWorld Environment, as shown in Figure 4.18, we compare the performance of five notable models: the Single Image Encoder, LSTM, ViT, and two variations of CBGT-Net—CBGT-Net-Pose (which uses camera pose information for context-aware accumulation) and CBGT-Net-SIFT (which uses SIFT features in images for context-aware accumulation). It is evident that all four models outperform the Single Image classifier, reinforcing that individual images lack sufficient information for accurate class identification of a building's fire stage.

Moreover, CBGT-Net models surpass the ViT baseline across all thresholds. This can be attributed to the misalignment of pre-trained ViT models [3] for the specific task. ViT models are originally trained for single-image classification by dividing the image into patches. However, we modified the input to provide a sequence of nine images as patches forming a single image processed through the transformer architecture. This modification made it difficult to leverage the pre-trained ViT effectively for the new task, even after fine-tuning.

Figure 4.18 also illustrates that CBGT-Net models outperform the LSTM model at lower thresholds, which are scenarios requiring decisions based on less information. The LSTM model only surpasses CBGT-Net at a decision time of around 16, highlighting CBGT-Net's robustness in situations with insufficient information. While LSTM performs better for longer input sequences due to its architecture's suitability for extended contexts, CBGT-Net is more reliable for making decisions with partial and incomplete information, as demonstrated in lower threshold scenarios.

Another interesting result is presented in Figure 4.22, which illustrates the behavior of decision time as the fixed threshold value increases. As expected, higher thresholds require CBGT-Net models to accumulate more evidence in favor of any class category to cross the threshold and make a decision. Consequently, the decision time, i.e., the average time taken before a decision is made, increases.

We also evaluated the performance of different models in the MiniWorld Environment across various paths, as shown in Figure 4.5. These paths are significant because they encompass different angles around the building. For example, Path 1 covers the building from all angles, providing a 360-degree view through the sequence of images collected along this path. In contrast, other paths cover only partial angles of the building, such as 99, 37, or 186 degrees.

Paths that cover only a partial segment of the building tend to provide the models with just the partial information being repeated. Consequently, even if observations are collected repeatedly by moving back and forth along these paths, the total information collected will still remain insufficient for correct decision-making. Therefore, it is intriguing to compare how the better-performing baselines—LSTM and CBGT-Net models—perform with data collected from these different paths.

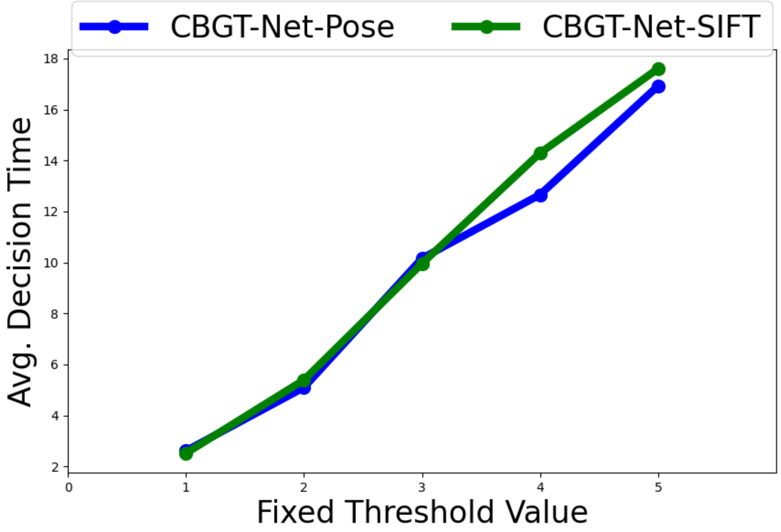


Figure 4.22: Average Decision Time taken by two versions of CBGT_Net - one using camera pose and other using Sift features for context-aware accumulation, trained at different threshold values (τ) on MiniWorld Environment

Additionally, it will be interesting to compare the practical performance of the two context-aware accumulation methods within the CBGT-Net models: one based on camera pose information (CBGT-Net-Pose) and the other based on SIFT features (CBGT-Net-SIFT).

Figure 4.23 illustrates the performance of CBGT-Net-Pose, CBGT-Net-SIFT, and the LSTM model for datasets collected along paths covering approximately 37, 99, 186, and 360-degree angles around the building. It is reasonable to expect that viewing the building from all directions would provide the most information for determining the fire stage. However, viewing only 37 degrees of the building's 360-degree view can be very uninformative for decision-making.

As shown in Figure 4.23, CBGT-Net models consistently outperform the LSTM baseline when observations are collected from a narrow view of the building's side. For higher thresholds (threshold > 3), CBGT-Net models often refrain from making a prediction because the accumulated evidence from these limited views never surpasses the threshold, also shown in plot 4.24. This behavior is desirable in high-threshold scenarios where precision is prioritized over decision speed. The LSTM model performs poorly with paths covering only 37 degrees and even 99 degrees for long sequence inputs. In contrast, CBGT-Net models demonstrate significant robustness and resilience to low-information data sequences. With a lower threshold, CBGT-Net models achieve decent performance across all path variations. For higher thresholds, they either refrain from predicting (as in case 1) or make high-accuracy predictions, as shown in graphs 2, 3, and 4 in 4.23.

Furthermore, among the CBGT-Net models, the one utilizing SIFT feature-based context-aware accumulation slightly outperforms the camera-pose-based method. This discrepancy can be attributed to how CBGT-Net-Pose treats observations when the camera location changes significantly. CBGT-Net-Pose considers such observations as entirely new, weighting them as substantial evidence. However, it's plausible that the relevant information—the section of the building visible in this view—remains the same as captured from the previous location. This aspect is addressed in the alternative context-aware accumulation method that employs SIFT features. It measures the similarity between two observations captured at different times. If the features match significantly, it downplays the evidence; conversely, if they are notably different, it accentuates the evidence. This approach offers a more reliable method for context-aware evidence accumulation, serving as a better check to avoid counting repeating information multiple times during accumulation, which could potentially mislead the model's prediction.

Figure 4.29, illustrates the performance of various CBGT-Net models trained at different threshold values, plotted against the angle covered by the path. The greater the coverage, the more information gained, resulting in better model accuracy. It is noteworthy that for a threshold of 1, the model performance plateaus after 99 degrees, as the low threshold allows decisions to be made within 2-3 observations. Therefore, the amount of information provided—whether from around the building or just one side—does not significantly impact performance. However, as the threshold value increases, the performance variation with the amount of information collected becomes more pronounced.

It is also intriguing to observe how CBGT-Net models with context-aware accumulation refrain from making decisions when presented with partial and incomplete information, such as paths covering only 37 degrees out of the 360 degrees around the building. Refer to Figure 4.24, where Paths A, B, C, and D subtend angles of 37°, 99°, 16°, and 360° around the building.

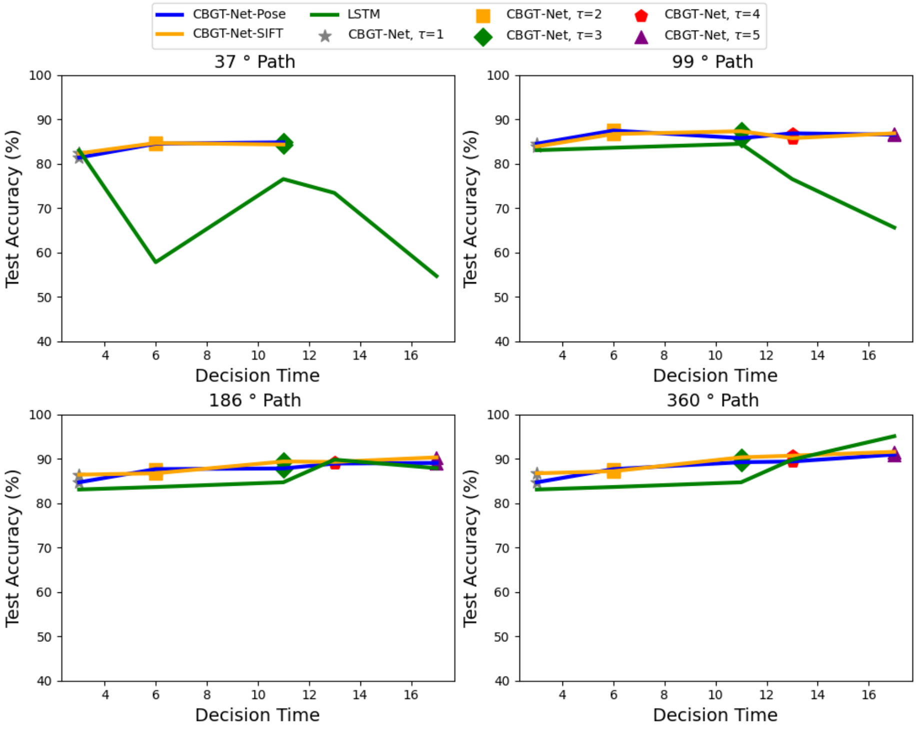


Figure 4.23: Path-wise inference accuracy of the CBGT-Net (Pose and SIFT) and LSTM baseline as a function of decision time for the MiniWorld Environment

For Path A, CBGT-Net with Pose and SIFT variants never make a decision because they discard duplicate information and fail to accumulate enough evidence to cross the threshold, especially when the threshold is set as high as 5 in this plot. Conversely, the single CBGT-Net with additive accumulation is misled by repetitive information and predicts an outcome in every scenario. As the path angles increase, such as in Path B and Path C, context-aware CBGT-Net models begin to make decisions in some scenarios. This highlights how context-aware accumulation prevents the CBGT-Net model from prematurely deciding in situations where it should refrain due to incomplete information.

Figure 4.25 compares the accuracy of CBGT-Net models across different paths. The left plot shows results for models trained with a fixed threshold of 1, while the right plot shows results for models trained with a fixed threshold of 5. These plots clearly demonstrate that incorporating historical observation information into the accumulation process enhances the model's prediction accuracy. The CBGT-Net with simple additive accumulation performs worse than those with Pose and SIFT information. Among these, CBGT-Net-SIFT outperforms CBGT-Net-Pose, as SIFT feature matching is a more effective metric for identifying information overlap between images compared to using camera pose location to determine the extent of overlap between observations.

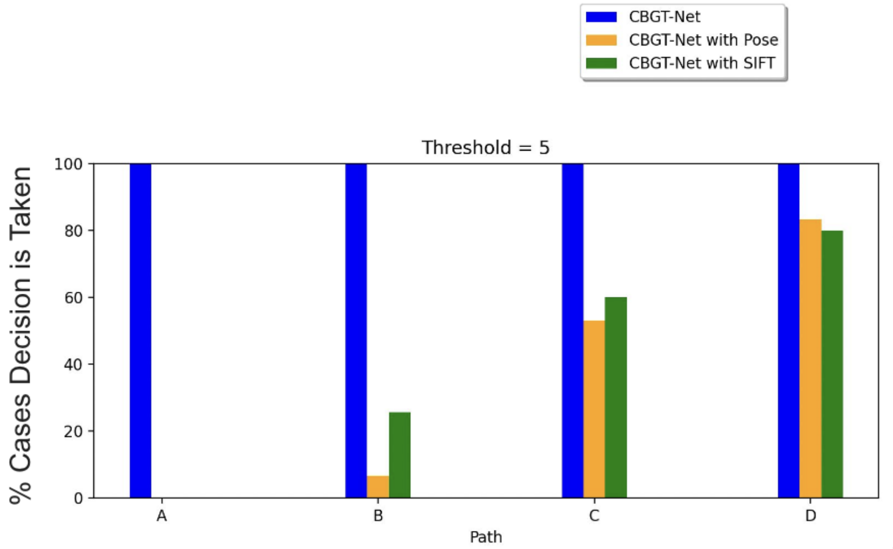


Figure 4.24: Comparing various versions of CBGT-Net across different paths in terms of the percentage of cases where a decision was made

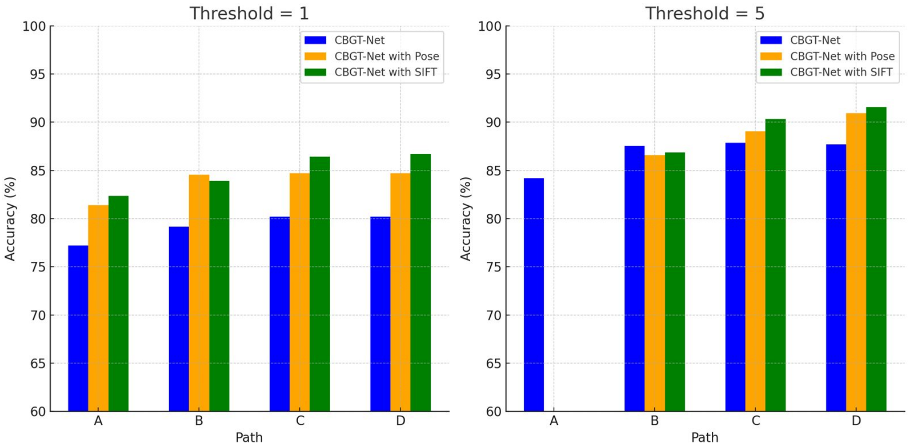


Figure 4.25: Comparing the accuracy of different versions of CBGT-Net on data streams collected along 4 different paths

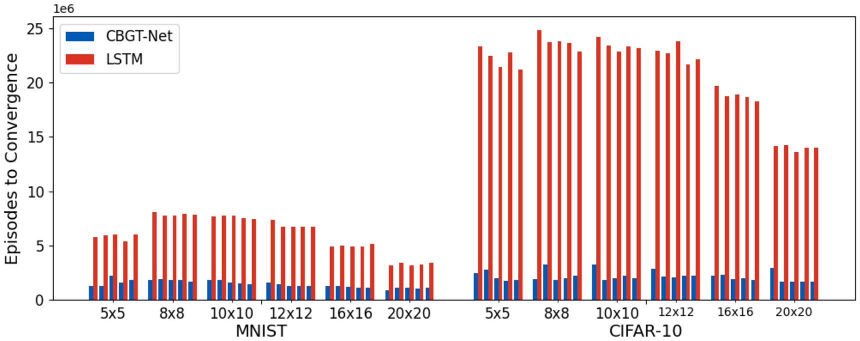


Figure 4.26: Number of training episodes required for convergence of CBGT-Net and LSTM models for each environment. For each group, the decision threshold and corresponding LSTM episode lengths increase from left to right.

For MNIST, CIFAR10, and MiniWorld environments, the LSTM baseline adds 950 additional trainable parameters to the model. Despite this, the CBGT-Net exhibits improved performance and robustness.

Figure 4.26 shows the number of episodes needed for convergence for training the CBGT-Net and LSTM models for each environment. In all cases, the CBGT-Net required fewer training episodes than the LSTM model. On average, the CBGT-Net required 75.4% fewer training episodes than the LSTM model for the MNIST environments, and 89.4% fewer episodes for the CIFAR-10 environments. In conclusion, the CBGT-Net consistently outperforms the LSTM model in terms of training efficiency across environments.

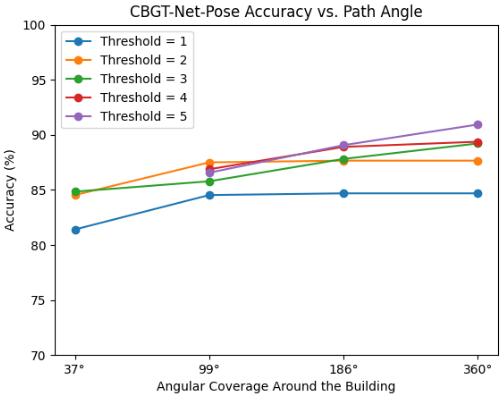


Figure 4.27: CBGT-Net-Pose model accuracy for different paths

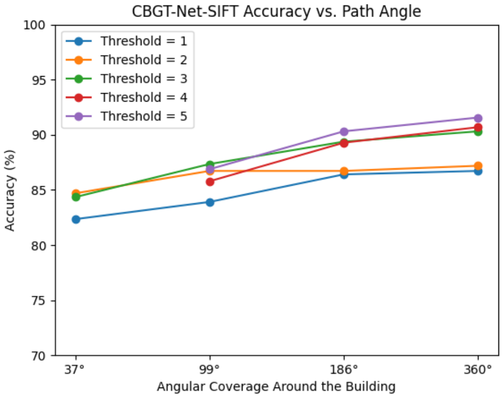


Figure 4.28: CBGT-Net-SIFT model accuracy for different paths

Figure 4.29: CBGT-Net model performance shown across various paths at different fixed threshold values

Chapter 5

Discussion

This paper introduces a neural network architecture based on cortico-basal ganglia-thalamic circuits found in mammalian brains and demonstrates its effectiveness in learning inference tasks from streams of low-information data. We demonstrated that the model can learn to categorize images based on a stream of small patches extracted from the image or a sequence of images of a large object in the scene, as well as specify when it should decide based on the amount of supporting evidence observed, as opposed to a fixed number of observations. The model generally outperforms similar models that use LSTMs for recurrent connections and transformer architectures like ViT and is especially robust to decreasing information presented in each observation.

In addition to improvements in performance and robustness to low-information observations, the evidence accumulation component provides for transparent deliberation, which we believe offers potential benefits in human-autonomy collaborations. Specifically, each element in the evidence accumulator corresponds to the model's current preference towards a desired decision, and the margin between accumulator values and decision threshold indicates how imminent a decision may be, as well as the presence of potential alternative decisions that have high levels of accumulated evidence.

There are several promising directions for the future development of the proposed model, with potential extensions to individual components opening up multiple avenues for exploration.

For example, the accumulator component could be enhanced by incorporating non-linear dynamics, such as attention-style context-aware weighting of evidence. This approach, particularly if inspired by biologically motivated dynamics, could introduce a sophisticated mechanism for prioritizing relevant evidence. Moreover, the incorporation of a memory buffer could facilitate the accumulation of evidence over longer time horizons. This could involve selectively aggregating evidence from a vast pool of data collected during an ongoing sequence.

In the current model formulation, we stipulated that the dimensionality of the evidence encoder must align with the number of decision categories. However, future research could investigate alternative representations of evidence and adapt the accumulator accordingly. This flexibility could lead to more robust and adaptable decision-making processes.

The transparency provided by the evidence accumulation aspect of the model offers insights into its deliberation process. This transparency presents an opportunity to delve into human understanding, interaction, and potential intervention with the model. Exploring these aspects could shed light on how humans perceive and interact with complex decision-making systems.

Finally, there is potential for applying the model as part of a policy for sequential decision-making tasks. This approach would empower agents to learn to perform actions based on accumulated evidence, rather than reacting solely to individual observations. Such an application could have far-reaching implications for various domains, including autonomous systems and decision support systems.

Bibliography

[1] Akshat Agarwal, Abhinau Kumar V, Kyle Dunovan, Erik Peterson, and Timothy Verstynen. Better safe than sorry: Evidence accumulation allows for safe reinforcement learning. *arXiv preprint arXiv:1809:09147*, 2018. 1, 2

[2] Krista Bond, Kyle Dunovan, Alexis Porter, Jonathan E Rubin, and Timothy Verstynen. Dynamic decision policy reconfiguration under outcome uncertainty. *Elife*, 10:e65540, 2021. 2

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=YicbFdNTTy`. (document), 4.1.4, 4.15, 4.2

[4] Kyle Dunovan and Timothy Verstynen. Believer-skeptic meets actor-critic: Rethinking the role of basal ganglia pathways during decision-making and reinforcement learning. *Frontiers in Neuroscience*, 10, 2016. ISSN 1662-453X. doi: 10.3389/fnins.2016.00106. URL `https://www.frontiersin.org/articles/10.3389/fnins.2016.00106`. 2

[5] Farama Foundation. Miniworld: A minimalistic 3d environment simulator for reinforcement learning robotics research. `https://miniworld.farama.org/`, 2023. Accessed: 2023-02-14. 4.1.1

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4.1.2

[7] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 4.1.4

[8] Kenney. City kit commercial. `https://kenney.nl/assets/city-kit-commercial`, 2024. Accessed: 2024-06-13. 4.1.1

[9] Alex Krizhevsky. Learning multiple layers of features from tiny images. pages 32–33, 2009. URL `https://www.cs.toronto.edu/~kriz/`

learning-features-2009-TR.pdf. 4.1.1

[10] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324, 1998. 4.1.1, 4.1.2

[11] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010. 4.1.1

[12] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. doi: 10.1023/B:VISI.0000029664.99615.94. 4.1.3

[13] T. V. Maia and M. J. Frank. From reinforcement learning models of the basal ganglia to the pathophysiology of psychiatric and neurological disorders. *Nature Neuroscience*, 14(2):154–162, February 2011. 1, 2.1

[14] Amit Mandelbaum and Daphna Weinshall. Distance-based confidence score for neural network classifiers. *arXiv preprint arXiv:1709.09844*, 2017. 2

[15] Jooyoung Moon, Jihyo Kim, Younghak Shin, and Sangheum Hwang. Confidence-aware learning for deep neural networks. In *international conference on machine learning*, pages 7034–7044. PMLR, 2020. 2

[16] M Andrea Pisauro, Elsa Fouragnan, Chris Retzler, and Marios G Philiastides. Neural correlates of evidence accumulation during value-based decisions revealed via simultaneous eeg-fmri. *Nature communications*, 8(1):1–9, 2017. 1, 2.1

[17] Samira Pouyanfar, Saad Sadiq, Yilin Yan, Haiman Tian, Yudong Tao, Maria Presa Reyes, Mei-Ling Shyu, Shu-Ching Chen, and Sundaraja S Iyengar. A survey on deep learning: Algorithms, techniques, and applications. *ACM Computing Surveys (CSUR)*, 51(5):1–36, 2018. 1

[18] Philip L Smith and Roger Ratcliff. Psychology and neurobiology of simple decisions. *Trends in neurosciences*, 27(3):161–168, 2004. 1

[19] Andrea Stocco, Christian Lebiere, and John R Anderson. Conditional routing of information to the cortex: A model of the basal ganglia's role in cognitive coordination. *Psychological review*, 117(2):541, 2010. 1, 2.1

[20] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, 2019. 1