Lateralized Learning to Solve Complex Problems

by

Abubakar Siddique

A thesis
submitted to the Victoria University of Wellington
in fulfilment of the
requirements for the degree of
Doctor of Philosophy
in Engineering.

Victoria University of Wellington
2021

Abstract

Artificial intelligence systems have become proficient at linking environmental features to targets to describe simple patterns in data. However, these systems can struggle with many real-world problems that entail hierarchical patterns within patterns, for example, in recognizing object ontologies where one object is made-up of other objects. Although it is possible to capture such complex structures by utilizing state-of-the-art deep networks, the knowledge is often stored in layers that do not take advantage of the potential benefits provided by reusing patterns within a layer of the system.

Biological nervous systems can learn knowledge from simple and small-scale problems and then apply it to resolve more complex and large-scale problems in similar and related domains. However, rudimentary attempts to apply this transfer learning in artificial intelligence systems have struggled. This may be due to the homogeneous nature of their knowledge representation. The current understanding of the learning mechanisms in the brains of human and non-human animals can be used as inspiration to improve learning in artificial agents. Research into lateral asymmetry of the brain shows that it enables modular learning at different levels of abstraction that facilitate transfer between tasks.

The proposed thesis is that an artificial intelligence system that enables lateralization and modular learning at different levels of abstraction has the ability to solve complex hierarchical problems that a similar homogeneous system can not. The comprehensive goal of this thesis is to accomplish lateralized learning, inspired by the principles of biological intelligence, in artificial intelligence systems. The objectives are to show that

lateralization and modular learning assist the novel systems to encapsulate the underlying knowledge patterns in the form of building blocks of knowledge. These building blocks of knowledge are to be tested on analyzable Boolean tasks as well as practical computer vision and navigation tasks. Academic contributions are related to the novel methods of the linking, transfer, and sharing of learned knowledge which are based on the analogous strategies of the brain.

This thesis proposes a general framework for lateralized artificial intelligence systems. The novel lateralized framework spans key aspects of knowledge perception, knowledge representation and utilization, and patterns of connectivity. It determines the essential functionality, critical methods, and associated parameters that are required to be incorporated into an artificial intelligence system to behave as a lateralized artificial intelligence system.

This thesis creates a novel evolutionary machine learning system, by adapting the lateralized framework, to obtain a proof-of-concept of the lateralized approach. Considering the same problem at different levels of abstraction enables the novel system to reframe a complex problem as a simple problem and efficiently resolve it. The results on analyzable Boolean tasks show that the problems that contain a natural hierarchy of patterns are solved to a scale that exceeds previous work (i.e. 18-bit hierarchical multiplexer problem), and reusing learned general patterns as constituents for future problems advances transfer learning (e.g. n-bit parity problem effectively becomes a sequence of 2-bit parity problems).

This thesis creates a novel lateralized artificial intelligence system, by adapting the lateralized framework, that shows robustness in a real-world domain that includes uncertainty, noise, and irrelevant and redundant data. The results of image classification tasks show that the lateralized system efficiently learns hierarchical distributions of knowledge, demonstrating performance that is similar to (or better than) other state-of-the-art deep systems as it reasons using multiple representations. Crucially,

the novel system outperformed all the state-of-the-art deep models for the classification (binary classes) of normal and adversarial images by $0.43\% - 2.56\%$ and $2.15\% - 25.84\%$, respectively. This thesis creates another novel multi-class lateralized system for computer vision problems to show that the lateralized approach can be scaled and not limited to learning classifier systems.

Both the Boolean and computer vision problems are single step problems in the spatial domain. However, most biological tasks, which exhibit heterogeneity, are temporal in nature. This thesis creates a novel frame-of-reference based artificial intelligence system, by adapting the lateralized framework, to address perceptual aliasing in multi-step decision making tasks. Considering aliased states at a constituent level enables the novel system to place them appropriately in holistic level policies. Consequently, the novel system transforms a non-Markov environment into a deterministic environment and efficiently resolves it. Experimental results show that the novel system effectively solves complex aliasing patterns in non-Markov environments that have been challenging to artificial agents. For example, the novel system utilizes only 6.5, 3.71, and 3.22 steps to resolve Maze10, Littman57, and Woods102, respectively.

A final contribution of this work is to obtain evidence of the benefits/costs of lateralization from artificial intelligence in order to inform cognitive neuroscience. Given that lateralization is ubiquitous in brains, evolutionary benefits can be assumed, at least in some domains. But that does not mean those benefits extend to all domains. The cognitive neuroscience research community has been struggling to determine the trade-off between the benefits and costs of lateralization. It has been hypothesized that lateralization has benefits that may counterbalance its costs. Lateralization has been associated with both poor and good performance. This thesis demonstrates the value of viable artificial systems for testing the costs and benefits of lateralization in biological systems.

Dedication

To all, especially my father

S.M. Hassan

, who spent their lives supporting and caring for deprived members of the community

Acknowledgments

All praise unto **ALLAH**, Lord of all the worlds, Who enabled me to learn a little more about His universe.

First and foremost, I would like to express my deep gratitude to my supervisors Prof. Will N. Browne and A/Prof. Gina M. Grimshaw for their guidance, enlightenment, and support. They helped me in various ways to shape my academic and leadership skills. They provided me unflinching encouragement that helped me to complete this thesis. I am very fortunate to have supervisors who spent many dedicated hours reviewing my drafts, even on weekends and holidays. I am indebted to my supervisors more than they know.

I offer my utmost gratitude to Prof. Will for his valuable bits of advice, critical analyses, and thought-provoking suggestions. He was available for a discussion at any time despite his busy schedule. Thanks Will! Needless to mention A/Prof. Gina, who had been a source of inspiration. Despite my engineering background, she helped me understand neuroscience concepts, especially at the start of my Ph.D. I am extremely grateful to Gina for the detailed feedback on my writing, especially w.r.t. multidisciplinary audience.

Special thanks to Science for Technological Innovation for awarding me the doctoral scholarship and to both my supervisors for supporting my application. I would also like to appreciate the support rendered by the Faculty of Science and the members of the Evolutionary Computation Research Group. I am indebted to the Victoria University of Wellington

Students' Association for awarding me the Gold Award and the School of Engineering and Computer Science for nominating me for the Landers Award. I am very thankful to the selection board for providing me the opportunity to serve the Vic Muslims Club and the cooperative behavior of the club members. I gratefully acknowledge the support for the Muslim community rendered by Victoria University, different organizations, society, government, and the honorable prime minister Jacinda Ardern.

Words are inadequate in expressing my gratitude for the blessings, unending support, and prayers of my beloved parents (S.M. Hassan and late Shamim Hassan) and parents-in-law (Col.(r) Naseer Ahmad and Tasneem Naseer). I acknowledge the sacrifice of my beloved father to sending me abroad, even though it was very hard for him due to emotional attachment. I owe my deepest gratitude to my brother (Ch. Kaleem) and my sisters for unconditional love, encouragement, and support. I am very grateful to have had a chance to share my Ph.D. journey with extraordinary friends and colleagues. I gratefully acknowledge the endless love, sacrifice, and support of my beloved wife and truest friend Muniba AB. I apologize to my kids (Zalayed, Omaiza, Zunairah, and Nabiha) for not giving them their deserved time.

Publications

- **Abubakar Siddique**, Will N. Browne, and Gina M. Grimshaw. "Lateralized Learning to Solve Complex Boolean Problems" In *IEEE Transactions on Cybernetics*, (IF = 11.448, successfully passed the second round) (cf. Chapter 5)
- **Abubakar Siddique**, Will N. Browne, and Gina M. Grimshaw. "Lateralized learning for robustness against adversarial attacks in a visual classification system" In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2020)*, ACM, pp. 395-403. (cf. Chapter 6)
- **Abubakar Siddique**, Will N. Browne, and Gina M. Grimshaw. "Frame-of-Reference based Learning: Overcoming Perceptual Aliasing in Multi-Step Path Planning Tasks" In *IEEE Transactions on Evolutionary Computation (TEVC)*, (IF = 11.554), (cf. Chapter 7)
  DOI: 10.1109/TEVC.2021.3102241
- **Abubakar Siddique**, Will N. Browne, and Gina M. Grimshaw. "Learning classifier systems: appreciating the lateralized approach" In *Genetic and Evolutionary Computation Conference Companion (IWLCS 2020)*, pp. 1807–1815. (cf. Chapter 8)
- **Abubakar Siddique**, Will N. Browne, and Gina M. Grimshaw. "Lateralized learning for robustness against adversarial attacks in a multi-class visual classification system" In *IEEE Transactions on Emerging Topics in Computational Intelligence*, (being prepared), (cf. Chapter 6)

- **Abubakar Siddique**, Will N. Browne, and Gina M. Grimshaw. "Lateralization in Agents' Decision Making: Evidence of Benefits/Costs from Artificial Intelligence" In *Symmetry, Special Issue "Cognitive and Neurophysiological Models of Brain Asymmetry"* (being prepared), (cf. Chapter 8)

Contents

1 Introduction **1**

1.1 Scope 1
1.2 Motivations 9
1.3 Thesis Statement 12
1.3.1 Research Questions 12
1.4 Goals 14
1.5 Major Contributions 17
1.6 Thesis Organization 19

2 Background **23**

2.1 Natural Cognitive Architecture 24
2.1.1 Hemispheric Lateralization 25
2.1.2 Semantic Knowledge 30
2.1.3 Frames of Reference 38
2.1.4 Benefits and Costs of Lateralization 44
2.2 Artificial Cognitive Architecture 46
2.2.1 Perception 47
2.2.2 Attention 48
2.2.3 Action Selection 48
2.2.4 Memory 49
2.2.5 Learning 49
2.3 Artificial Learning Methods 49
2.3.1 Neuroscience Inspired AI 51

2.3.2 Evolutionary Computation 54
2.4 Associated Techniques 62
2.4.1 Features 62
2.4.2 Adversarial Attacks 63
2.5 Chapter Summary 64

3 Benchmark Problems and Approaches 65
3.1 Benchmark Problems 66
3.1.1 Boolean Problems 66
3.1.2 Computer Vision Problems 70
3.1.3 Navigation Problems 71
3.2 Benchmark Approaches 74
3.2.1 Existing Lateralized AI Systems 75
3.2.2 Relevant Approaches for Boolean Problems 75
3.2.3 Relevant Approaches for Computer Vision Problems 78
3.2.4 Relevant Approaches for Navigation Problems 79
3.3 Chapter Summary 82

4 Lateralized Framework 85
4.1 Introduction 86
4.1.1 Chapter Objectives 89
4.1.2 Chapter Organisation 90
4.2 Lateralization in Vertebrate Brains 91
4.2.1 Representation and Processing 91
4.2.2 Coordination 92
4.2.3 Goal-driven Processing 93
4.3 Lateralized Framework 93
4.3.1 Features of a Lateralized AI System 94
4.3.2 Lateralized Architecture 96
4.4 Problem Domains 102
4.5 Discussion 104
4.6 Chapter Summary 104

5 Lateralization for Boolean Problems 107
5.1 Introduction . . . 108
5.1.1 Chapter Objectives . . . 110
5.1.2 Chapter Organisation . . . 111
5.2 Lateralized System . . . 112
5.2.1 Knowledge Identification . . . 113
5.2.2 Resolve Problem . . . 118
5.2.3 Learning Methodology . . . 120
5.3 Walk-Through of the Algorithms . . . 123
5.4 Experimental Design . . . 126
5.4.1 Problem Domains . . . 126
5.4.2 Learning Order . . . 129
5.4.3 Experimental Setup . . . 129
5.5 Experiments . . . 131
5.5.1 Multiplexer Problems . . . 131
5.5.2 Parity problems . . . 133
5.5.3 Hierarchical Problems . . . 139
5.6 Experimental Analysis . . . 142
5.6.1 Overhead of Irrelevant Sub-problems to LateralXCS . 142
5.6.2 Interpretation of Decisions . . . 144
5.7 Discussion . . . 149
5.8 Chapter Summary . . . 150

6 Lateralization for CV Problems 153
6.1 Introduction . . . 154
6.1.1 Chapter Objectives . . . 156
6.1.2 Chapter Organisation . . . 157
6.2 Binary-class Lateralized System . . . 158
6.2.1 Lateralized System . . . 158
6.2.2 Experimental Design . . . 165
6.2.3 Experiments . . . 167

6.2.4 Experimental Analysis 169
6.3 Multi-class Lateralized System 173
6.3.1 Lateralized System 173
6.3.2 Experimental Design 180
6.3.3 Experiments 181
6.4 Discussion 185
6.5 Chapter Summary 187

7 Lateralization for Navigation Problems 189
7.1 Introduction 190
7.1.1 Chapter Objectives 195
7.1.2 Chapter Organisation 196
7.2 Frame-of-Reference based System 196
7.2.1 Code-paths 196
7.2.2 Policies 198
7.2.3 Adjacent States Map 201
7.2.4 Aliasing Identification and Disambiguation 201
7.2.5 Predict Aliased Version 206
7.2.6 Overall Strategy 208
7.3 Experimental Design 212
7.3.1 Experimental Setup 212
7.4 Experiments 213
7.5 Experimental Analysis 215
7.6 Discussion 222
7.7 Chapter Summary 223

8 Discussions 225
8.1 Lateralized Systems 226
8.2 Relevant Approaches 231
8.2.1 Ensemble Systems 231
8.2.2 Deep Learning 232
8.2.3 Granular Computing 234

8.3 Evidence of Benefits/Costs from AI 236
8.3.1 Lateralization in AI 238
8.4 Chapter Summary 240

9 Conclusions and Future Work **243**
9.1 Achieved Objectives 244
9.2 Main Conclusions 247
9.2.1 Lateralized Framework 247
9.2.2 Proof-of-Concept of the Lateralized Approach 250
9.2.3 Robustness of the Lateralized Approach 251
9.2.4 Effectiveness of the Lateralized Approach 253
9.3 Future Work 254
9.3.1 Lateralized AI Systems for Neuroscience 254
9.3.2 Lateralized Ensemble Systems 255
9.3.3 Lateralized Deep Learning 256
9.3.4 Lateralized Granular Computing 257

1

Introduction

Biological nervous systems can learn knowledge from simple and small-scale problems and then apply it to resolve more complex and large-scale problems in similar and related domains. However, rudimentary attempts to apply this transfer learning in artificial intelligence systems have struggled. This may be due to the homogeneous nature of their knowledge representation. Research into lateral asymmetry of the brain shows that it enables modular learning at different levels of abstraction that facilitate transfer between tasks. This thesis will draw inspiration from the architecture of biological intelligence to create machine learning systems. This chapter introduces the main concepts and research questions that will be addressed in this thesis.

1.1 Scope

The behavior exhibited by machines, based on the principles of natural intelligence, is called artificial intelligence (AI). Recently, AI has been boosted

by the revolution in science and technology that yields both computers with high processing power and a large amount of data. Consequently, a broad range of AI-based systems has been developed that are playing critical roles in many aspects of everyday life ranging from self-driving cars to security and surveillance [1, 2, 3, 4, 5, 6, 7]. An artificially intelligent agent is a system (or a machine) that gets input from the environment, analyses it, and takes appropriate action under some criteria, e.g. to maximize its current or future rewards. Moreover, a system falls under the AI domain if it has the ability to learn from the environment and solves a problem in such a way that it mimics human cognition [8].

AI has become proficient at linking environmental features to describe simple patterns in data. However, AI systems can struggle with many real-world problems that entail hierarchical patterns within patterns; for example, recognizing object ontologies where one object is made-up of other objects. Although it is possible to capture the overall complex structures by utilizing state-of-the-art deep networks, the knowledge is often stored in layers that do not take advantage of the potential benefits provided by reusing patterns within the layers elsewhere in the system [9]. Moreover, conventional AI systems may learn complex patterns but they need a huge/deep network of homogeneous knowledge, human-in-the-loop intervention, and complex architectures [9, 10, 11]. This thesis considers how AI, especially machine learning, can better model hierarchical relationships and therefore produce better solutions to complex problems.

Machine learning (ML) is an important field of AI. It enables machines to interact with their environments and learn through experience [12]. The discipline has five major tribes: symbolists, Bayesians, analogizers, connectionists, and evolutionaries [13]. Symbolist researchers represent knowledge in the form of symbols. To a symbolist, intelligence arises from the manipulation of knowledge represented as symbols. Bayesian researchers use reasonable expectations to represent a state of knowledge. A Bayesian represents knowledge as a set of expectations based on cumu-

lative evidence. The resultant system works on the principle that some hypotheses become more likely as compared to others based on evidence, as the learning proceeds. Analogizing researchers consider the similarities between the environmental data. They work on the principle that if two instances share some features then there is a reason to believe that they have knowledge (other features) to share. Although these perspectives have had some success in solving problems, contemporary research in machine learning is dominated by connectionist and evolutionary perspectives. The focus of this thesis is on connectionist and evolutionary viewpoints, as described below.

Connectionist researchers represent knowledge in the form of artificial neural networks. The connections between the nodes of a network have weights. These weights are updated, by utilizing methods such as backpropagation, to support the desired learning of input to output relationships. These networks can have multiple layers that store knowledge in a distributed fashion. These networks generate a homogeneous knowledge representation, such that all features are treated equally in each layer, to learn a linearly separable relationship between features and target. Connectionist systems are useful because they identify and learn patterns within the data.

Evolutionary researchers represent knowledge in the form of genotypes, such as strings, trees, or rules. Inspired by biological evolution, an evolutionary system creates rules, solutions, or models of the data. A rule is a fundamental building block of knowledge (BBK).[1] Traditionally, a rule has two parts, i.e. a condition and an action. A rule recommends an action if its condition matches the environmental state. An evolutionary system can utilize rules to decide actions based on the current condition, i.e. if *'condition'* then *'action'*. Moreover, a rule has fitness value based on its usefulness. When rules are being formed, each is treated equally. How-

[1] A BBK is a unit of knowledge utilized by the AI agent to represent a part of a problem or whole problem.

ever, as learning proceeds, rules that have greater fitness and better fill niches[2] survive, while those that are less fit do not. Consequently, niches and fitness of rules start to build.

The connectionists' approach is a prime example of homogeneous systems, presented below.

Homogeneous Systems: The prefix "homo" is derived from Greek, meaning "same". Knowledge that is represented (stored) in the same manner can be referred to as homogeneous knowledge. For example, the knowledge stored in a layer in artificial neural networks treats each pixel in the same way even if the connecting weights are different. In this thesis, an AI system is called a homogeneous system if it equally considers all the features of an input environmental signal in the same manner.

A deep network with a convolution filter and a single pathway stride is an obvious example of a homogeneous system. Similarly, a deep network-based system with different convolution filters (e.g. $3 \times 3, 5 \times 5$) and different pathway strides, which are combined in the end in an ensemble way, is still a homogeneous system. The majority of AI systems are homogeneous. Homogeneous systems cannot differentiate between constituent level simple features and holistic level complex features because each feature needs to exist at the same time to be compared.

Homogeneous systems work well when there are only simple features in the domain or the relationship between features and target (e.g. class or action) is linearly separable (i.e. a linear combination of features can be used to separate out specific targets) [14, 15]. However, these systems struggle when there are complex patterns of features in the domain [16, 17]. Often these complex features are made-up of patterns of features, i.e. hierarchical patterns within patterns. The ability to consider the problem at multiple levels may effectively resolve such complex hierarchical features.

[2] A niche is an area of the sample space where the neighboring instances share a common property.

The evolutionaries' approach has the inherent support for heterogeneity. Heterogeneous systems are explained below.

Heterogeneous Systems: The word "hetero" is also Greek prefix, which means "different". The knowledge that is represented (stored) at different levels of abstraction can be referred to as heterogeneous knowledge. In this thesis, an AI system is called a heterogeneous system if it has the ability to consider an input environmental signal at different levels of abstraction simultaneously to generate a heterogeneous knowledge representation. A deep network with different convolution filters (e.g. $3 \times 3, 5 \times 5$) and different pathway strides, which are applied at different parts of the same image such that they can share their findings within the hidden layers can be considered as a heterogeneous system. Capsule networks are a step toward creating heterogeneous neural networks [18] they extract different spatial relationships of features within a single layer. Holistic level features are generally heterogeneous as they may entail hierarchical patterns within patterns. Heterogeneous features can be parsimonious as they only encode necessary information. However, it is still not clear how these heterogeneous features could be represented at different levels of abstraction, i.e. at a constituent level and holistic level, parsimoniously.

A level of abstraction can be considered as a perspective to address a problem. For example, in a visual task, a local viewpoint (eyes, nose, and mouth of a cat) and big-picture (whole cat) are two different levels of abstraction. Moreover, a constituent level means a representation of the most basic elements of knowledge, i.e. individual features and simple niches; whereas, a holistic level means a more abstract knowledge representation; i.e. higher-order features extracted across niches.

The raw features from the environment can be considered the lowest level in abstraction. These are linked together in a rule in a series of disjunctive/conjunctive normal forms. If higher-level features can be constructed, this forms the next level of the hierarchy and so forth. The ability to use heterogeneous features as a complex single feature, known as fea-

ture construction, is not often leveraged in existing learning systems.

Biological Inspiration: A biological brain is not an amorphous system. It is made up of different modules that communicate through electrochemical signals. These modules can be considered as metamodel nodes that subserve exiguous low-level functions. Moreover, metamodel nodes form local networks that are linked through excitatory and inhibitory connections. Each module solves a specific problem or part of a problem. The knowledge learned from small-scale and simple problems can be reutilized by the brain to solve large-scale and complex problems in similar and related domains [19, 20, 21, 22, 23].

Two organizing principles of vertebrate (and many invertebrate) brains — lateralization and modularity of function — support reuse of learned knowledge at different levels of abstractions [20, 24, 25, 26, 27, 28]. The propensity of a specific cognitive process to be performed more efficiently and precisely by one hemisphere as compared to the other is called hemispheric lateralization [29]. At the macro-structural view, the left and right hemispheres look alike. However, they have distinct neuroanatomy, neurochemistry, and functional architecture [29, 30]. An environmental signal is simultaneously presented to both the hemispheres such that they process it at different levels, i.e. constituent level and holistic level. Lateralization enables the biological brain to process both individual features and global patterns in parallel. For example, in many domains, the left hemisphere processes elementary (constituent) information while the right hemisphere works at a higher (holistic) level of abstraction. It has been hypothesized that this asymmetry enhances cognition as well as neural efficiency [21, 23, 31].

In biological intelligence, both hemispheres contribute to most higher-order cognitive and perceptual tasks, and the differences between them are more relative than absolute. Some hemispheric differences are concerned with the scale at which the same sensory inputs are represented for subsequent processing. For example, in visual perception, the left hemi-

sphere processes information at a local (or constituent) level while the right hemisphere processes information at a more global (holistic) level [32, 33, 34]. Hence in a face recognition/identification task, the individual features of the face are identified by the left hemisphere, e.g. eyes, nose, lips, etc; whereas, the configural arrangement of the face is handled by the right hemisphere. Similarly, in speech perception, the left hemisphere processes segmental information (individual phonemes that make-up words) while the right hemisphere processes super-segmental information (global intonational patterns that reflect emotion or intention of the speaker) [35, 36, 37].

Lateralization is one-way brains achieve heterogeneity. Other neural systems also use heterogeneous representations. For example, in navigation, different frames of reference are utilized to provide multiple representations of an environment from different viewpoints [27, 28]. This frame-of-reference (FoR) based learning enables the biological brain (vertebrate and many invertebrate) to process the same information at different levels of abstraction, i.e. constituent level (local viewpoint) and holistic level (world viewpoint or complete map) [24, 26]. For example, animals log their position with respect to three frames of reference while navigating through an environment, i.e. egocentric, allocentric, and routecentric [27, 28, 38, 39]. Egocentric FoR provides a local viewpoint from the animal's perspective; allocentric FoR provides a global map of the environment (a bird's eye view), while route-centric FoR describes the path of the animal through the environment. These three FoRs are BBKs, when combined, provide a world viewpoint (complete map) of the environment.

Effective cognition requires that complementary computations (whether distributed across hemispheres or modules) be coordinated. Recognizing faces requires that we integrate individual features (left) with their configural arrangement (right) [40]; understanding a joke requires that we integrate the literal meanings of individual words (left) with their alternative subtext (right) [41]; understanding a song requires that we integrate the

lyrics (left) with the melody (right) [42]. It is the coordination between the left and right hemispheres that enables the transfer of critical information at different levels of abstraction.

Vertebrate brains have the ability to select the computations required to perform a specific task from the most suitable and relevant hemisphere. Goal-driven processes analyze the problem at hand and shift control to the superior and suitable hemisphere. For example, if the emotional state of a conversational partner is most relevant, outputs from right hemisphere speech processing systems will dominate; however, if the linguistic elements are of concern, then left hemisphere computations are prioritized [31, 43]. The connections between hemispheres in vertebrate brains can be *excitatory* or *inhibitory*, allowing for either integration or inhibition, as goals dictate [44]. The ability to identify which hemisphere is best matched to the task is important in practical situations.

Although the representation of the environment at different (i.e., constituent and holistic) levels of abstraction may yield benefits, it could also increase workload and waste resources. An effective system should therefore reuse features that occur at both constituent and holistic levels, and not simply enumerate each level. Moreover, it should make good use of excitatory and inhibitory signals to avoid redundant processing.

The importance of considering the same problem at different levels of abstraction can be illustrated by a problem that originated in steel mills twenty years ago [45, 46], when existing data mining techniques missed a critical relationship that was present in the data. In steel mills, the width of a steel strip needs to be guided into downstream manufacturing processes. For example, if the side-guide-setting was $80cm$ and the width of the strip was $82cm$ then there would be a problem. Similarly, if the side-guide-setting was $70cm$ and the width of the strip was $74cm$ then there would be a problem. Data mining techniques were able to extract this information about individual problems but could not extract the more abstract information that *if the side-guide-setting is less than the width of the strip, then there*

would be problems. Data mining techniques could only consider the constituent level and therefore missed the global problem. There are many similar examples in the real-world. For example, when we recognize a car, we can identify it as a vehicle for transportation, but we can also identify windscreen or hubcaps, depending on the problem we are trying to solve. In real-life we often face problems that have multiple parts and are also multifaceted, where it is useful to be able to consider the lower level of detail and/or higher levels of abstraction.

Conventional AI systems do not have the ability to consider the same input signal at different levels of abstraction, i.e. constituent level and holistic level. Considering the same problem at different levels of abstraction may enable AI systems to reformulate a complex problem as a simple problem and efficiently resolve it. These limitations motivate the development of new heterogeneous feature-based techniques, that incorporate principles of lateralization and modular learning, to solve complex problems. It is worth developing heterogeneous feature-based systems to solve complex hierarchical problems. However, it is not clear/known how lateralization can be incorporated in AI systems, i.e. (i) how to process the same input signal at different levels of abstraction such that the constituent and holistic levels can be considered in a complementary manner, (ii) how to (re)utilize the features (constituent level and holistic level) that are computed due to the processing of the input signal at multiple levels, (iii) how to use excitatory and inhibitory signals to avoid extraneous processing and produce optimal solutions.

1.2 Motivations

The majority of existing AI systems develop a huge network of homogeneous knowledge to solve complex problems, which is neither reusable nor scalable, and so fail in unexpected situations [9, 16, 47, 48]. Transfer learning of basic features has been used but this does not consider fea-

ture manipulation or take advantage of the relationship between patterns within patterns. Although AI systems are becoming trusted sufficiently to be used in daily life [1, 4], they are still unable to exhibit the intelligence of a four-year-old human child in many domains [49, 50].

Modern classifier systems can effectively classify targets that consist of simple, homogeneous features. However, they struggle to deal with many real-world problems that entail hierarchical patterns within patterns. For example, when classifying instances of environmental features in the Boolean domain, evolutionary computing is proficient at linking features together but can fail to detect patterns that are made up of patterns of features [51]. Although it is possible to capture such complex structures in homogeneous systems, they require large/deep networks of knowledge and do not take advantage of the potential to transfer knowledge between levels in the hierarchy [9]. Thus a new system is needed that can handle complex hierarchical patterns by (re)utilizing BBKs at constituent and holistic levels of abstraction.

One problem with homogeneous knowledge representation is that it does not exhibit robustness against noisy and irrelevant data. For example, connectionist systems are highly vulnerable to adversarial attacks in visual classification tasks [48, 52]. A single, targeted pattern can disrupt classification performance. Moreover, a small (imperceptible to a human) perturbation to an image can fool many homogeneous systems resulting in the wrong prediction made with high confidence [53, 54]. Homogeneous systems work well when the relationship between features and target is linearly separable. These systems encourage linear behavior for performance efficiency. However, this hallmark can be exploited to fool homogeneous systems [52, 55, 56].

The majority of homogeneous systems struggle to capture complex structures in an environment spatially (in single-step problem domains) but also temporally (in multi-step domains). For example, perceptual aliasing is a long-standing problem for artificial agents in applying reinforce-

ment learning (RL) to many multi-step tasks [10, 16, 57, 58, 59, 14, 60]. It occurs when the agent's internal representation confounds external world states, i.e. the agent's current perception is unable to distinguish environmental states which appear identical but require different actions [57]. One reason may be that the agent only has a local FoR, and cannot act simultaneously on the global level (i.e. cannot make decisions informed by the world level map). FoR based learning is a feature of vertebrate intelligence that allows multiple representations of an environment at different levels of abstraction. Thus a FoRs-based learning agent is needed that can consider the input environmental signal at different levels of abstraction to handle perceptual aliasing problems. This enables the resolution of patterns made-up of patterns of features. Heterogeneous features could represent knowledge at different levels of abstraction in compact building blocks of knowledge that are relevant and sufficient to solve a specific problem [17]. However, it needs to be investigated how these features can be combined or recombined to form hierarchies of knowledge.

The motivation is to develop a new technique that can incorporate lateralization and modular learning to exhibit robustness against noisy and irrelevant data and solve complex problems. The performance benefits of lateralization are so far not clear. Lateralized AI systems have not been investigated for the following reasons: (i) a simple problem does not need lateralization; (ii) lateralized systems need to process the same input signal at the constituent level and holistic level, which appears to double the workload. So a major performance benefit is needed in compensation; (iii) two different techniques are required to process the same signal at constituent and holistic levels; (iv) it is not always clear how to take a single instance and split it up (decompose) such that the constituent and holistic level can be considered not just at the same time, but also in a complementary manner; (v) it is not known how to utilize excitatory and inhibitory signals for the selection of appropriate knowledge structures to avoid extraneous computations; (vi) it is not clear how to store and (re)use learned

lateralized knowledge in similar, related, and different domains; (vii) the robustness of lateralized systems, to handle uncertainty and noise in data, is unknown; (viii) lack of availability of an underlying lateralized framework that can be adapted to create lateralized artificially intelligent systems for a wide range of problem domains; (ix) it is not clear whether lateralization can be applied to a wide range of problems and scenarios, e.g. single or multiple steps, supervised or reinforcement learning, Boolean or real-valued features, and Markov or partially observable Markov decision processes.

1.3 Thesis Statement

The proposed thesis is that:

> *An artificial intelligence system that enables lateralization and modular learning at different levels of abstraction has the ability to solve complex hierarchical problems that a similar homogeneous system cannot.*

1.3.1 Research Questions

This thesis raises the following research questions:

(i) *How to create a lateralized framework for AI systems?* Lateralization is a significant feature of biological intelligence. It is ubiquitous in vertebrates and manifests its advantages in survival and reproduction. Lateralization has not been investigated as a feature in AI systems. It is a challenging task to devise a lateralized framework, inspired by the principles of natural intelligence, that can be adapted to develop AI systems for a wide range of problem domains. This includes the identification of critical methods and associated parameters that are required for an AI system to behave as a lateralized AI system.

(ii) *How to simultaneously process a single environmental input at different levels of abstraction?* Biological intelligence simultaneously processes the same environmental signal at different levels of abstraction, i.e. at the constituent level and the holistic level. Many real-world and complex problems entail hierarchical patterns within patterns. Novel techniques must be developed that empower AI agents to consider the given environmental instances at different levels of abstractions.

(iii) *How to create a heterogeneous representation of knowledge that can be (re)used at different levels of abstraction?* Heterogeneity can be seen in biological intelligence, which can learn new knowledge from simple and small-scale problems and then apply it to resolve more complex problems in similar and related domains. It is challenging for AI agents to effectively resolve problems that consist of complex heterogeneous patterns of features, i.e. hierarchical patterns within patterns. It is required to investigate how to create novel strategies that have the ability to (re)utilize knowledge components at different levels of abstraction. For example, a holistic knowledge component at one level could be (re)utilized as a constituent knowledge component at a higher level of abstraction.

(iv) *When and how to inhibit or excite different system components?* The connections between hemispheres in vertebrate brains can be excitatory or inhibitory, allowing for either integration or inhibition of computational structure, as goals dictate. This ability to identify the most suitable and relevant hemisphere with respect to the task is important for real-life problems. It needs to be investigated how novel techniques can be developed that have the ability to enable communication between different system components through inhibition and excitation signals to efficiently resolve the problem and avoid extraneous computations.

1.4 Goals

The comprehensive goal of this thesis is to accomplish lateralized learning, inspired by the principles of biological intelligence, in AI systems. In order to achieve this goal and answer the research questions, the following objectives have been set:

1. *Create a general framework for lateralized AI systems.* This objective aims to devise a general framework that can be adapted to develop a lateralized AI system for a wide range of problem domains. To achieve this objective, the following sub-objectives have been set:

   (i) Establish the essential principles of lateralization from cognitive neuroscience, especially drawing from the cognitive architecture in vertebrate brains. The following aspects of lateralization that are relevant to this thesis will be explored in detail: (a) representation and processing of sensory information received from the environment, (b) coordination between hemispheres and integration of knowledge among different regions of the brain, and (c) achieving goal-driven processing through inhibition and excitation signals for performance efficiency.

   (ii) Devise a fundamental framework for lateralized AI systems. Identify the important features of a lateralized AI system, which may include knowledge perception, knowledge representation and utilization, and connectivity patterns. Determine the essential functionality that is required to be incorporated into an AI system to behave as a lateralized AI system.

   It is important to note that an individual lateralized AI system needs to be developed for a specific task by adapting the general framework of the novel lateralized AI system. This framework includes all the critical functions of a lateralized AI system. Each novel lateralized system needs to include the essential functionality of the

fundamental framework. If any of the essential functions of the fundamental framework are missed, the system will not be a lateralized AI system.

2. *Develop a lateralized AI system and test on interrogatable, single-step, scalable, and complex problem domains.* The first step will be to develop a novel lateralized AI system for complex Boolean problems to obtain the proof-of-concept of the lateralized approach. The RL technique will be applied to solve single-step classification tasks. To achieve this objective, the following sub-objectives have been set:

   (i) Develop a lateralized system such that a single input can be processed at different levels of abstraction, i.e. at the constituent level and/or the holistic level. Instead of mapping features to knowledge in a homogeneous manner that considers all input features equally, the problem will be split into two halves. One half will map sub-groups of features to knowledge at a constituent level, whereas the other will map all features to knowledge at a holistic level.

   (ii) Represent BBKs in a heterogeneous manner. Different sized blocks of knowledge can be recombined in a recursive manner, i.e. a holistic block can be (re)used as a constituent block at a higher level of abstraction.

   (iii) Identify and reuse the relevant BBKs to efficiently resolve complex Boolean problems, i.e. those consisting of patterns of patterns.

   (iv) Enable communication between different system components through inhibition and excitation signals to efficiently resolve the problem and avoid extraneous computations.

3. *Develop a lateralized AI system that shows robustness in a real-world domain that includes uncertainty, noise, and irrelevant and redundant data.*

This objective will show that the underlying lateralized framework can be adapted to complex real-world problems. A novel lateralized AI system will be developed to show robustness against adversarial attacks. The supervised learning technique will be applied to solve single-step visual classification tasks. To achieve this objective, the following sub-objectives have been set:

(i) Develop a lateralized system that can simultaneously process a single visual input at constituent and holistic level of abstraction.

(ii) Represent knowledge in a heterogeneous manner. Different knowledge components are utilized or re-utilized at different levels of abstraction, i.e. a holistic knowledge component at one level can be (re)utilized as a constituent knowledge component at a higher level of abstraction. Different system components coordinate to reuse the learned knowledge at different levels of abstraction.

(iii) Enable communication between different system components through inhibition and excitation signals to efficiently resolve the problem and avoid extraneous computations.

4. *Develop a frame-of-reference based AI system to address perceptual aliasing in multi-step decision making tasks.* Both the Boolean and visual problems are single step problems in the spatial domain. However, most biological tasks, where heterogeneity is applied, are temporal in nature, requiring multiple steps to achieve a solution. This objective aims to evaluate the effectiveness of the heterogeneous feature-based approach in the temporal domain. A novel FoR based AI system will be developed to evaluate the effectiveness of the novel approach in resolving perceptual aliasing in multi-step decision making tasks. The reinforcement learning technique will be applied to

resolve multi-step state-action transitions. To achieve this objective, the following sub-objectives have been set:

(i) Create a novel FoRs based system that has the ability to process a single input at different levels of abstractions to provide multiple environmental views, i.e. a local viewpoint (constituent knowledge) and a world viewpoint (holistic knowledge, complete map) of the same state.

(ii) Create a heterogeneous representation of knowledge, i.e. local viewpoint (constituent knowledge) and world viewpoint (complete map, holistic knowledge). This knowledge will be utilized or re-utilized at different levels of abstraction to generate constituent representation and holistic representation, which will allow interpretation of learned policies.

(iii) Integrate different building blocks of knowledge at different levels of abstraction to generate an unambiguous representation of knowledge. The resultant knowledge will be used to disambiguate complex patterns of aliased states, which will enable the learning of stable policies.

(iv) Create a strategy to activate/deactivate policies[3] such that the agent can reach the goal state by using the minimum number of steps.

1.5 Major Contributions

This thesis makes the following major contributions to the fields of machine learning and AI.

(i) This thesis proposes a general framework for creating a lateralized AI system for the first time. The important features of a lateralized

[3]A policy, like a route, can be considered as a large pattern prescribing state transitions from a starting to the goal state.

AI system are identified as knowledge perception, knowledge representation and utilization, and connectivity patterns. Moreover, the critical functions are identified that are needed to be implemented by an AI system to behave as a lateralized system, i.e. simultaneous processing of input signal at different levels of abstraction, coordination and integration of knowledge among different system components, utilization of inhibition and excitation signals to avoid extraneous computations. This framework can be adapted to develop a lateralized AI system for a wide range of problem domains. Supporting evidence is provided in the following contributions.

(ii) Lateralization and modular learning are successfully applied at different levels of abstraction to resolve single-step, scalable, and complex problems. A novel strategy is developed to consider the same problem at different levels of abstraction (i.e. constituent level and holistic level). Considering the same problem at different levels of abstraction enables the novel system to reframe complex problems as simple problems and efficiently resolve them. The results of analyzable Boolean tasks show that the lateralized system has the ability to encapsulate underlying knowledge patterns in the form of building blocks of knowledge. Problems with a natural hierarchy of patterns are solved to a scale beyond previous work, and reusing learned general patterns as constituents for future problems advances transfer learning.

(iii) Lateralization is successfully applied to develop a visual classification system that exhibited robustness against uncertainty, noise, irrelevant, and redundant data. A new technique is developed to simultaneously process the visual input from an environment at different levels of abstraction, i.e. constituent level and holistic level. A novel strategy is created to handle simple problem instances at the context phase, whereas, more attention is automatically given to the

noisy and corrupt problem instances based on the feedback from the context phase. It enables the novel lateralized system to make correct decisions for badly corrupted images where either the constituent predictions are confused or the holistic prediction favors the wrong class. A novel strategy to inhibit or excite the most appropriate learning component of the lateralized system is implemented. It offers efficiencies and effectiveness beyond ensemble or co-evolutionary approaches. The experimental results demonstrate that the lateralized system successfully exhibits robustness against adversarial attacks. The novel system outperformed state-of-the-art deep models for the classification of normal and adversarial images.

(iv) FoR based learning is successfully applied at different levels of abstraction to learn stable policies for multi-step tasks in partially observable Markov environments. A novel code-path based strategy is developed to consider the same environmental instance at different viewpoints, i.e. local viewpoint (constituent-level BBKs, egocentric FoR) and world map (holistic-level BBKs, allocentric and routecentric FoRs). This empowers the learning agent to successfully address perceptual aliasing problems by identifying and disambiguating the aliasing patterns. The experiments demonstrate that the novel system has the ability to utilize or re-utilize relevant learned BBKs at different levels of abstraction to learn aliasing patterns consisting of patterns of features. A step-change in performance is achieved, e.g. the state-of-the-art heavily aliased mazes are successfully resolved.

1.6 Thesis Organization

The remainder of this thesis is organized as follows. The literature review of the required background knowledge is presented in Chapter 2. Chapter 3 presents the benchmark problems and approaches that are used to evaluate the effectiveness and robustness of the developed systems. Each of the

four subsequent chapters addresses one of the established research objectives, respectively. A comprehensive discussion on lateralized approach, implemented for different problem domains, is presented in Chapter 8. Finally, Chapter 9 presents the conclusion and future work.

Chapter 2 provides a literature review of the required and relevant background knowledge from Cognitive Neuroscience to AI. It includes a detailed description of lateralization and modularity in biological intelligence. This chapter also illustrates the different machine learning techniques that are used to develop lateralized systems for this thesis.

Chapter 3 describes a wide range of benchmark problems, drawn from different domains, that are utilized to evaluate the effectiveness and robustness of the developed lateralized systems. It includes single and multiple step problems, supervised and reinforcement learning problems, Boolean and real-valued features problems, and problems that entail Markov or partially observable Markov decision processes. Finally, it presents the relevant state-of-the-art benchmark techniques that have previously been developed to address the hierarchical problems that are used to evaluate the lateralized AI systems.

Chapter 4 presents the general framework that can be used to develop a lateralized AI system for a wide range of problem domains. It explains all the critical methods and associated features of a lateralized AI system. Finally, it illustrates a basic architecture that can be adapted to develop a lateralized system for problems in a wide range of domains.

Chapter 5 presents a novel lateralized AI system which is developed to solve complex Boolean problems. It illustrates how the lateralized approach enables the novel system to reframe a complex problem as a simple problem and efficiently resolve it. Finally, this chapter presents the experimental results which demonstrate that the lateralized system solved complex Boolean problems beyond previous work, e.g 18-bit hierarchical multiplexer problem and n-bit parity problem.

Chapter 6 presents two novel lateralized AI systems which are devel-

oped to show robustness against noisy and irrelevant data in computer vision problems. It explains a novel strategy that is developed to inhibit or excite the most appropriate learning structure. This empowers the novel systems to effectively and efficiently resolve the corrupt images. Finally, this chapter presents the experimental results which demonstrate that the lateralized system successfully exhibited robustness against adversarial attacks beyond previous work.

Chapter 7 presents a novel frame-of-reference based AI system which is developed to resolve perceptual aliasing problems in non-Markov environments (navigation problems). It explains a novel code-path based strategy to provide environmental viewpoints at different levels of abstraction, i.e. local viewpoint and world map. Finally, the experimental results have presented that show the effectiveness of the novel approach in resolving perceptual aliasing problems in state-of-the-art complex mazes, e.g. Maze10.

Chapter 8 provides a comprehensive discussion of the lateralized framework for AI systems and its adaption for different problem domains. The limitations of the lateralized approach and obstacles in developing lateralized AI systems are discussed. Moreover, it provides insight into the benefits and costs of lateralization for agents addressing complex tasks.

Chapter 9, concludes this thesis. It highlights the achieved objectives and major research contributions of this thesis. This chapter also suggests open questions and opportunities arising for future research work.

2

Background

The background chapter serves as a foundation for this thesis. It highlights the important aspects of the research topic, identifies the research gaps, and critically evaluates the relevant research studies. As the main aim of this thesis is to create lateralized artificially intelligent systems based on the evidence from cognitive neuroscience, a more thorough revision of cognitive architecture, in vertebrate brains, is provided than is typical in an engineering thesis. The goals of this chapter are three-fold: first, to introduce the relevant principles of cognitive neuroscience that will inform the work and to describe the trade-off between the benefits and costs of lateralization in biological intelligence; second, to present relevant aspects of biological and artificial cognitive architectures; third, to review the current state of knowledge, from machine learning, that will provide a foundation upon which the novel modular systems will be constructed.

2.1 Natural Cognitive Architecture

Cognition is a process used by the biological brain to acquire knowledge and comprehension based on experience, thoughts, and senses [61]. It comprehensively incorporates critical processes such as representing an environment, learning, reasoning, evaluating, decision making, producing language, and generating solutions to tasks. Cognition is implemented by neural processing mechanisms [62, 63]. Miller and Gazzaniga introduced the term "Cognitive Neuroscience" [64] to describe a branch of science that deals with the biological processes and features related to cognition. It explains the functionality and connectivity of the brain that allow it to produce adaptive thoughts, feelings, and actions. Cognitive processes produce new knowledge by utilizing previously learned knowledge. According to a broad definition of cognitive systems (natural and artificial), they are information processing systems that can perform various tasks [61].

Biological intelligence is associated with brains. The brain of a vertebrate has a complex architecture that consists of a massively parallel, extremely interconnected, and distributed system. It has the ability to efficiently utilize parallel and distributed resources to achieve learning, recognition, reasoning, evaluating, communicating, decision making, and executing actions. Cognitive integration is accomplished when the brain exploits all of the resources to understand the current situation, comprehend the environment, recall relevant past memories, analyze future perspectives, and decide actions accordingly [65].

A biological brain is not an amorphous system. It is made up of different modules that communicate through electrochemical signals [21, 22]. These modules can be considered as metamodel nodes that subserve exiguous low-level functions. Moreover, metamodel nodes form local networks that are linked through excitatory and inhibitory connections [23]. In biological brains, cognition arises out of networks of neurons which are organized in ways that are modular but interconnected. Each module

solves a specific problem or part of a problem. The knowledge learned from small-scale and simple problems is (re)utilized by the brain to solve large-scale and complex problems in similar and related domains [19, 20].

Vertebrate brains have a functional architecture that allows them to abstract knowledge from simple and small-scale problems and then reuse it to solve complex problems. It is not the intention of this thesis to model the specific architecture of a specific species; rather it takes inspiration from basic principles of functional organization that are fundamental to vertebrate intelligence. This thesis focuses on three such principles: lateralization, semantic knowledge, and frames-of-reference.

2.1.1 Hemispheric Lateralization

The propensity of a specific cognitive process to be performed more efficiently and precisely by one hemisphere as compared to the other is called hemispheric lateralization [29]. A brain can be divided into two major parts, i.e. left and right cerebral hemispheres. These hemispheres are separated by the medial longitudinal fissure, as shown in Fig. 2.1. They primarily communicate with each other through the corpus callosum. At the macro-structural view, the left and right hemispheres look alike. However, they have a distinct neuroanatomy, neurochemistry, and functional architecture [29, 30].

Hemispheric differences, with respect to structure, functionality, and connectivity, are too complex to be encapsulated in a simple dichotomy. Lateralization, with respect to structure, can be divided into two major types, i.e. module asymmetry and circuit asymmetry [67].

Module asymmetry is a type of lateralization in which a specific module or circuit component exists in only one hemisphere. It is an obvious type of hemispheric lateralization. For example, in hermit crabs, motor neurons that innervate pleopods are only on the left side of the brain [68]. In fruit-flies, Drosophila melanogaster (which has an asymmetrical body) has an unknown structure only in the right hemisphere [69]. Similarly, the

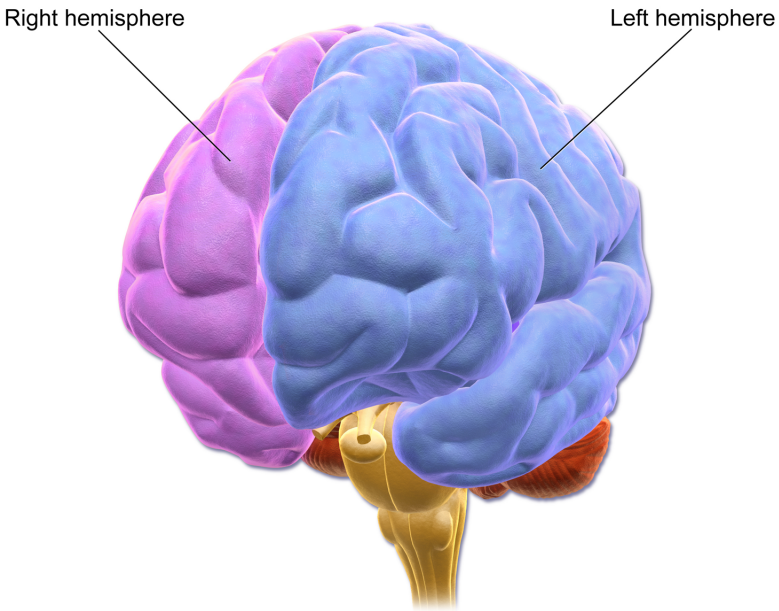


Figure 2.1: Human brain seen from the front (Source: [66]).

parapineal nucleus is only on the left side of the fish brain [67].

Circuit asymmetry is a type of lateralization in which an asymmetric circuit, with similar types of neurons, exists in both hemispheres. However, these circuits may have different synaptic connectivity of neurons, a different number of neurons, and neurons of different sizes. This type of asymmetry provides the simplest way to evolve lateralization in the brain, hence can be commonly observed at different levels of structural organization of the brain. For example, in chicks, thalamofugal fibres that are opposite to the side of the light-stimulated eye are more numerous than the thalamofugal fibres on the other side. Similarly, circuit asymmetry can also be observed in pigeons (tectofugal pathway), crabs (different sizes and numbers of cheliped motor neurons), and rats (hippocampal subcellular synaptic asymmetry) [67, 70, 71, 72].

Language is strongly associated with the left hemisphere, but the right

hemisphere complements the left in many aspects of language comprehension and production. The left hemisphere is responsible for the processing of systemized and rigid aspects of the language, whereas, right hemisphere takes care of the flexible aspects of the language [73]. The left hemisphere dominates in processing the phonological units of language, the syntax that links words together into sentences, and the literal meanings of words and phrases. However, the right hemisphere makes important contributions to language processing too, and dominates in processing the rhythm and tone of voice (or prosody), and figurative or alternative meanings. Moreover, both the cerebral hemispheres apply a different strategy for the processing of semantic information [74]. The right hemisphere has superior ability to process remote, novel, and atypical semantic relations, whereas, the left hemisphere emphasises the common or literal meanings associated with the word [75, 76, 77, 78, 79, 80].

Although hemispheric asymmetry was originally thought to be unique to human brains, a large body of research now demonstrates asymmetries in the brains of mammals, birds, and fish [81, 82, 83]. Hemispheric lateralization and its effects on animals' and humans' behaviour will be used as inspiration for developing novel methods in this thesis. Three aspects of lateralization are relevant for applications to AI:

Representation and Processing

Some functions are strictly lateralized to one hemisphere or the other. For example, each hemisphere receives sensory inputs from the opposite side of the body and controls the contralateral musculature. But, for higher-order cognition, differences between hemispheres are more relative than absolute, with both hemispheres contributing to most tasks. Often these hemispheric differences concern the scale at which the same sensory inputs are represented for subsequent processing. For example, in visual perception, the left hemisphere processes information at a local (or constituent) level while the right hemisphere processes information at a more

global (holistic) level [32, 33, 34]. Similarly, in speech perception, the left hemisphere processes segmental information (individual phonemes that make up words) while the right hemisphere processes super-segmental information (global intonational patterns that reflect emotion or intention of the speaker) [35, 36, 37].

These fundamental differences in representational scale may arise through filtering. For example, the Double Filtering by Frequency model proposes that the left hemisphere acts as a high pass filter, allowing it to represent detailed information that is available in high spatial or temporal frequencies. At the same time, the right hemisphere acts as a low pass filter, allowing it to represent global patterns that emerge in low spatial or temporal frequencies [84, 85]. Such complementary forms of representation are not limited to sensory information, however. For example in language processing, the left hemisphere may activate single, literal, meanings of words or sentences, while the right hemisphere keeps alternative, metaphorical, or figurative meanings active [73, 74]. This ability to represent and process the same problem instance at a local constituent level and a global holistic level will be incorporated in the work presented in this thesis.

The ability to consider the same problem at different levels of abstraction is an essential feature of lateralization. But left/right (or right/left) representation is not the only way to process information at a constituent level and holistic level. Biological brains also use heterogeneous knowledge representation. Lateralization is a special type of heterogeneity. In vertebrate brains, the same sensory information is represented and processed by different regions at different scales. For example, in animals' navigation an internal *egocentric* frame-of-reference (FoR) represents a local viewpoint, whereas *allocentric* and *routecentric* FoRs represent the world viewpoint of the environment [27, 28, 38, 86].

Coordination

Effective cognition requires that the computations carried out in opposite hemispheres be coordinated. Recognizing faces requires that we integrate individual features (left) with their configural arrangement (right) [40]; understanding a joke requires that we integrate the literal meanings of individual words (left) with their alternative subtext (right) [41]; understanding a song requires that we integrate the lyrics (left) with the melody (right) [42]. It is the coordination between the left and right hemispheres that enables the transfer of critical information at different levels of abstraction. This coordination will be included in the modules created here.

Goal-driven Processing

Vertebrate brains have the ability to select the computations required to perform a specific task from the most suitable and relevant hemisphere. Goal-driven processes analyze the problem at hand and shift control to the superior and suitable module/hemisphere. For example, if the emotional state of a conversational partner is most relevant, outputs from right hemisphere speech processing systems will dominate, however, if the linguistic elements are of concern, then left hemisphere computations are prioritized [31, 43]; egocentric and allocentric processing have been associated with right and left posterior cortex, respectively [87]; dorsal stream and frontal areas are active during egocentric coding, whereas, dorsal and ventral regions are involved during the processing of allocentric coding [88, 89]; similarly, sequential organization of consecutive choices is managed by the left hippocampus, whereas, allocentric or map-based navigation is handled by the right hippocampus [90]. The connections between hemispheres in vertebrate brains can be excitatory or inhibitory, allowing for either integration or inhibition, as goals dictate [44]. The ability to identify which computational structure is the most suitable to the task is important in practical situations. In the novel lateralized artificial system,

a strategy will be developed for the automatic activation/deactivation of the most suitable module/policy to handle the on-going situation while resolving a complex problem.

2.1.2 Semantic Knowledge

The study of the representation of symbols and their meanings in our brain is called semantics. Semantic memory is factual knowledge about the world [91]. Semantic knowledge is the reflection of the semantic database that we have built based on verbal and non-verbal experiences. It produces meanings, expressions, responses, and conceptual generalization for a verbal and non-verbal stimulus.

The semantic system is an example of biological intelligence system that takes advantage of both modularity and lateralization. It illustrates how modules and control processes interact to create a flexible knowledge system that can create new knowledge and provide optimal solutions to reach a given goal. The modular architectures of the novel systems will be designed based on inspiration from the hub and spoke model and concept formation hypothesis of biological semantic system [92, 93, 94].

Early models of semantic memory focused on its structure, proposing either an interconnected network of concepts or lists of features that define or characterise those concepts [95, 96, 97]. In both types of model, exposure to an exemplar (a word, a picture, or an object itself) activates the node or the feature list, leading to the activation of similar or nearby nodes and features through a process called priming. In these models, conceptual knowledge is static and localised within the semantic system.

More contemporary models emphasise the distributed and dynamic nature of semantic knowledge. The dominant theory to take this approach is the hub-and-spoke model. It extends earlier models by incorporating mechanisms of learning that underpin the development of conceptual knowledge [93, 94, 98]. The hub and spoke theory integrates two already existing ideas. First, concepts are developed by utilizing learned information

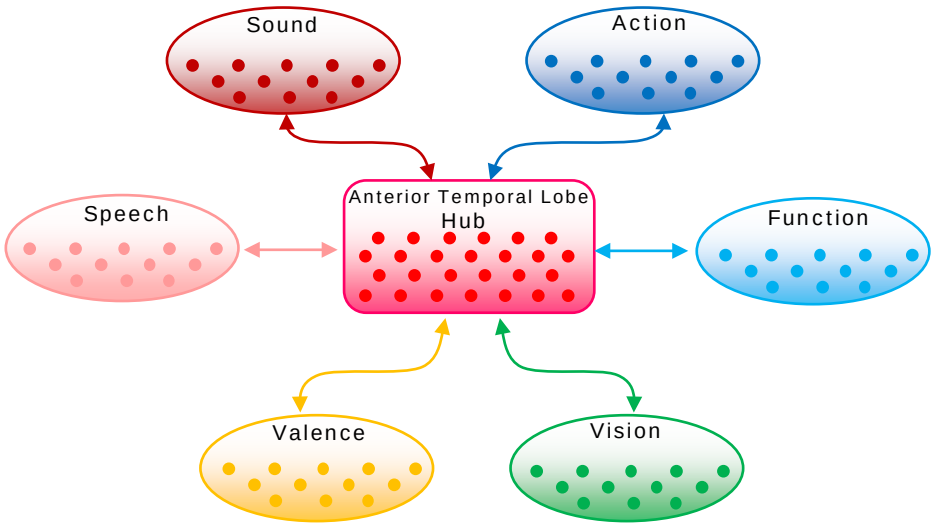


Figure 2.2: Computational framework of hub and spoke model (reproduced from [92])

based on verbal as well as nonverbal experiences. Moreover, the modality-specific sources of information (i.e. spokes) are distributed throughout the cortices. Second, the interaction among these sources of information is controlled by a single transmodal hub that is located in anterior temporal lobes [99, 100]. A schematic illustration of the computational architecture of hub and spoke model is shown in Fig. 2.2.

A spoke is a source of information. These spokes are interconnected through a central node known as a hub. In the hub and spoke model, spokes log modality-specific information in different processing units at different layers. These spoke layers are connected to a central transmodal hub. The model takes the inputs generated by the spokes in sequence and shares the updated information with other spokes connected with the hub. A neuroanatomical representation of spokes and hub is given in Fig. 2.3. It is important to note that the model is supported by data from brain

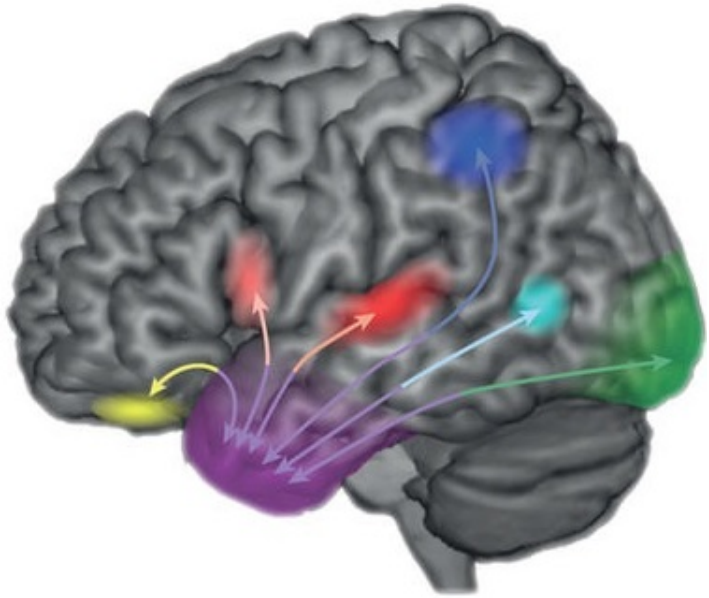


Figure 2.3: Neuroanatomical sketch of hub and spoke model (reproduced from [92])

damaged patients with semantic dementia (who lose entire concepts because they have lost the "hub"), and from neuroimaging studies of healthy brains that show that the same concepts activate different distributed networks (or spokes) depending on the goal.

Semantic cognition is the competency to utilize, control, and generalize learned knowledge during the execution of verbal as well as nonverbal tasks. It is an association of neurocognitive strategies that assist semantically inspired behaviors. These neurocognitive strategies convert noisy sensory input into meaningful information. Consequently, a brain observes the current environment, performs identifications, and makes conclusions.

Semantic cognition is based on two types of neural systems, i.e. semantic representation and semantic control. Semantic representation logs conceptual knowledge based on the higher-order relationships among dif-

ferent sources of information. These sources of information, such as motor, linguistic, sensory, exist in different parts of the cortex. Semantic cognition extracts conceptual representations based on experience and generalizes those concepts to other knowledge domains. Semantic control utilizes these conceptual representations to induce conclusions and control behaviors accordingly [93, 94, 98, 99, 101, 102, 103, 104, 105, 106, 107].

Semantic Representation

A brain stores representations of symbols at different levels of the physical cortex. Elementary level symbols are stored in the primary and secondary sensory cortices as physical entities. In spite of their simplicity, the symbols have a pivotal characteristic to provoke higher level multifaceted symbolic representation. The major portion of the cortex stores the higher level multifaceted representation of symbols by utilizing distributed neural networks. Semantic representations are the inner states that manifest a human's knowledge of the meaning of symbols. Symbols are the verbal or visual representations of objects that can be used to randomly identify the types of other objects. The complex and higher order representational geometries can be generated by utilizing the primary symbolic representations. Symbols generate entirely different sensory representations based on the type of physical inputs such as visual representation or auditory representation.

Concept Formation

Semantic representation records conceptual knowledge based on the higher order relationships among different sources of information. Concept formation has remained a focus of research in many disciplines such as neuroscience, philosophy, cognitive science, etc. Wernicke and Meynert developed a conceptualization model to describe the neural mechanisms for the formation and reactivation of concepts [108]. Their model made three key assumptions, i.e. (i) Concepts are stored in the form of building blocks of

semantic information, i.e. engrams. Specific building blocks are stored in the locality of such brain regions that are responsible for the corresponding knowledge domain, (ii) These regions of the building blocks are massively interconnected, (iii) The integration of these building blocks is the basis of conceptualization. Recently, Ralph presented the most assertive concept formation hypothesis. He considers that instead of forming a single representation in some specific region of the brain, the concepts are a product of integration and manipulation of semantic information stored in various regions of the brain [109].

Embodied theories and symbolic theories are two different hypothesis on the procedure of concept formation. Embodied theories consider that concepts are a product of the learned knowledge, verbal and non-verbal, based on experience. The activation of relevant experiential knowledge, termed as features, is still unresolved. There are two views regarding the activation of experiential knowledge, i.e. (i) activation at the time of creation or updating the concept (ii) activation whenever the concept is retrieved [110]. Symbolic theorists contend that logical, consistent and generalizable concepts cannot be generated by features only, it also requires experientially independent symbols [111]. These symbolic theories explain concept formation and generalization but they cannot explain the associations between the concepts and their relevant experiential features. Partially unifying theories highlight the significance and centrality of experience for the generation of concepts. In addition, these theories describe features to concepts mappings as well as generalization of knowledge [112, 113, 114, 115].

The hub and spoke model enhances the idea of unifying theories and provides explanations for the formation of coherent and generalizable concepts as well as the mappings between features and concepts. It is one of the well-supported models for semantic representation.

A *concept* is a coherent collection of knowledge about the world [92, 109]. According to the hub-and-spoke model, the features that comprise

a concept are distributed throughout the brain in modal spokes (motor, auditory, color, shape, etc.) that are formed through sensory-motor experience and/or abstracted from statistical regularities in the environment. These modal spokes then connect to a cross-modal hub (anatomically located in the anterior temporal lobe in humans), in which related concepts are connected to each other. The hub is, therefore, able to represent generalized concepts that are independent of any specific instance. The activation of a concept (e.g. a hammer) then entails activation of its cross-modal hub, along with modality-specific activation of its features (how it is held, what it is used for, its visual form, etc.) [93, 94, 98, 99, 109, 110, 100]. Although the network is bilateral, left and right hubs display subtle asymmetries by virtue of their connections to lateralized perceptual and motor systems [116].

Semantic Control

A controlling mechanism is required to make sure that the system generates the most relevant semantic representations and conclusions with respect to the current task as well as ongoing situation. Sometimes tasks have unconventional semantic requirements such as to highlight inferior meanings, utilize submissive features, or suppress a conceptual association. Moreover, the interpretation of the same concepts may generate different meanings with respect to time and nature of the ongoing task. Therefore a semantic control mechanism exists within the neural network, which is called the controlled semantic cognition framework. This mechanism utilizes the semantic representations and controls the semantic cognition to produce relevant meanings of the concepts with respect to the ongoing situation [92].

The controlled semantic cognition theory proposes both semantic representation and semantic control networks within the semantic framework. These networks have separate existence but they communicate with each other to generate conceptual meaning with respect to the current task. This

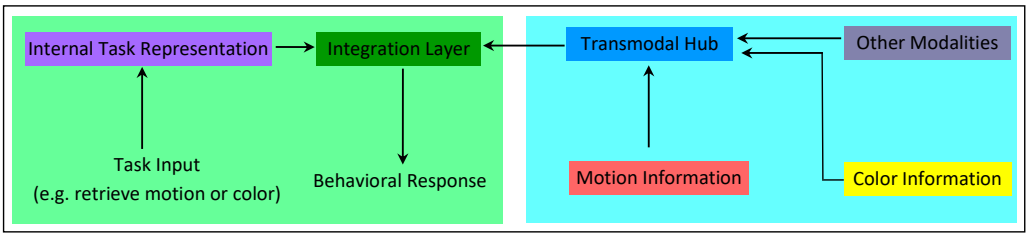


Figure 2.4: Controlled semantic cognition (reproduced from [92])

hypothesis resolved a challenge related to specificity and generalization. Concepts are supposed to be generalized as well as have specific meanings in association with the task at hand. According to the hub and spoke model, generalized concepts are formed by the interaction of transmodal hub and modality-specific semantic information, whereas the context of the current task is represented by a region through a separate semantic control network. These two networks integrate with each other to generate most appropriate behavior in association with time, context, and situation as shown in Fig. 2.4.

The learned semantic knowledge (e.g. about birds, flowers, trees, and fishes) is distributed across modality-specific nodes (e.g. what they look like, move like, and colors they have). These distributed nodes are linked together through the amodal hub. It is required to have different representations of the learned semantic knowledge to fulfil different goals, for example, to spot a bird, its color is highly relative, whereas, to catch a bird, its motion (hops) is the most valuable feature. These two tasks require focus on two different features (color vs motion). Therefore, the integrative layer utilizes task-independent structure in the transmodal hub to generate task-dependent representations, as shown in the Fig. 2.5.

Not all features of a concept are relevant in all contexts. Anatomically distinct semantic control mechanisms (localized in the frontal cortex) activate the most relevant meanings with respect to the current task or subtask [92]. For example, when asked to describe the similarities between a

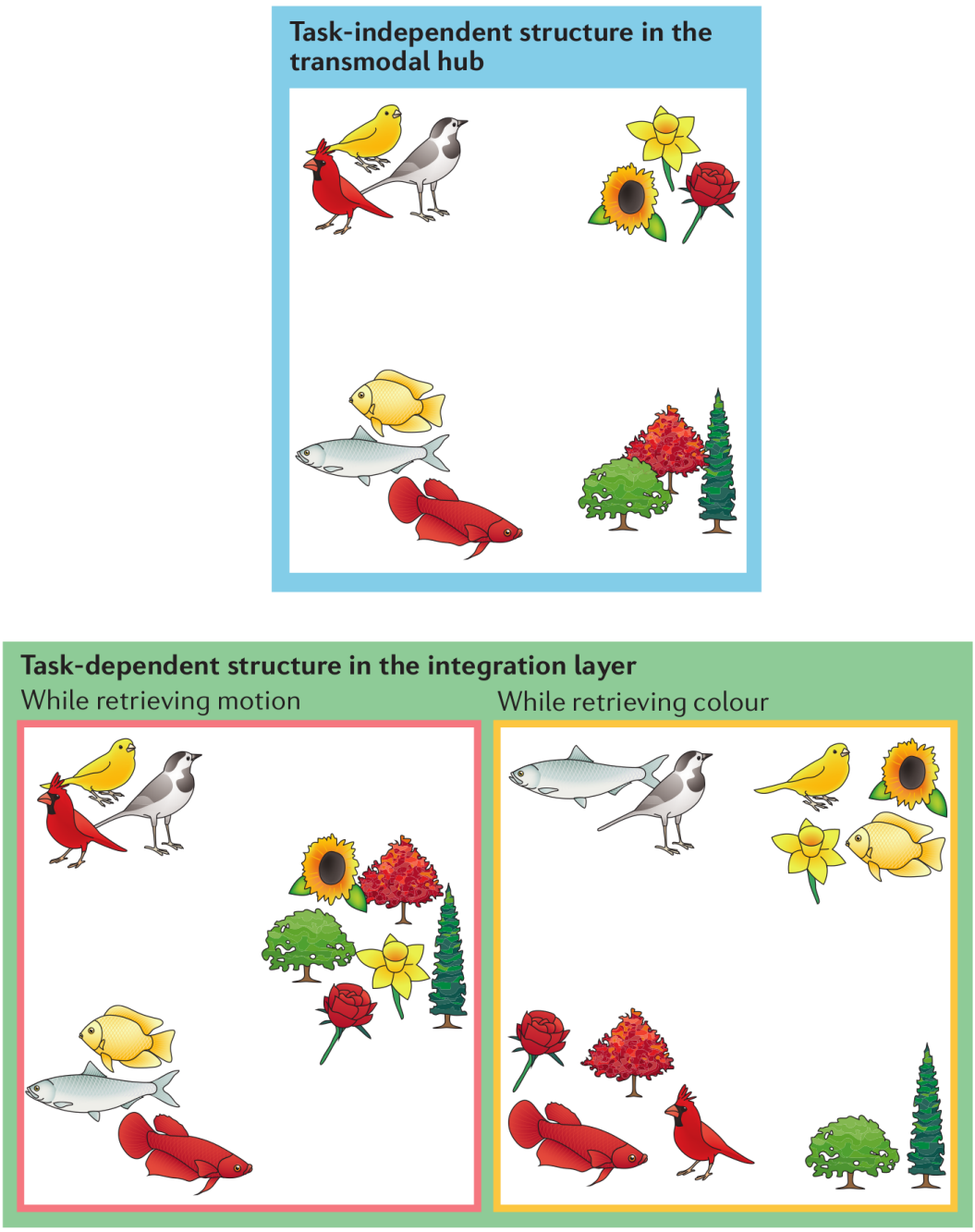


Figure 2.5: Illustrative example for controlled semantic cognition (reproduced from [92])

fire extinguisher and a tomato, people show increased activation in color-processing areas of the brain. In contrast, reflecting on the similarity between hammers and hacksaws activates areas that generate motor movements. Controlled semantic cognition therefore allows for concepts to be constructed from learned building blocks of knowledge (BBKs) as dictated by current needs [92, 93, 94, 98, 99, 101].

Concepts are therefore flexible; able to represent both generalizable and task-specific knowledge. This flexible use of concepts needs to be incorporated in this work to allow the representation of knowledge at different levels of abstraction through the activation of relevant BBKs. Both specific and generalized knowledge need to be utilized in concert to resolve heterogeneous problems. The cognitive concept hypothesis will be utilized to generate a block of knowledge (named *concept*) for each learned problem (see Chapter 5). The novel lateralized system will have the ability to utilize a learned BBK at a constituent level or holistic level depending on its occurrence in the problem instance.

2.1.3 Frames of Reference

Frames-of-reference play critical role in animals' navigation. They provide another example of how neural modules can be used dynamically as BBKs to represent complex environments and guide navigation. Although navigation is most commonly studied in rats, the assumption is that the neural and cognitive mechanisms that support learning in the rat also apply to all mammalian (and probably all vertebrate) brains. It has been reported that rats have the ability to efficiently utilize already learned knowledge during the exploration of similar and novel environments [27, 86, 117].

There are several research studies that analyze an animal's behavior during spatial navigation in familiar and novel environments. Generally, animals follow a specific path while moving from one position to another in an environment. The knowledge of a specific path within an environment assists animals for fluent and elegant movement. A specific path,

called route, can be described as a sequence of turns connected by straight segments of different lengths. A route has a particular shape and different routes may have similar shapes at different scales.

Animals log their positions with respect to different FoRs while navigating through an environment. Egocentric, allocentric, and routecentric are three important FoRs in this regard. Egocentric is an internal FoR, whereas, allocentric and routecentric are external FoRs. The information related to animal's head direction as well as motor and sensory actions, associated with navigation, is logged via an egocentric FoR. The information related to external cues and boundaries of the environment is logged via an allocentric FoR, whereas, the information related to the route spaces, distances between those spaces, and series of actions on the planned route is logged via a routecentric FoR [38, 39, 118, 119]. This ability to represent the same environmental state at different levels of abstraction (constituent level or local viewpoint; holistic level or world viewpoint) will be incorporated in this thesis.

Different regions of the brain, cortical and subcortical, produce these FoRs. Hippocampal place cells log a particular location of the route within the environment, whereas, posterior parietal cortex (PPC) and retrosplenial cortex (RSC) neurons have the ability to log the position of the animal within the route. These regions coordinate with each other and assist the animal to exhibit intelligent behavior during spatial navigation [27, 86, 117, 120, 121, 122].

Hippocampus

The hippocampus is one of the important components in the brain of vertebrates, as it is considered the hub of learning. Several studies have been conducted to explore the role played by hippocampal interneurons during spatial navigation. Place specific excitation in hippocampal activity was initially discovered by O'Keefe and Dostrovsky [123]. They observed that the peaks in the firing of hippocampal pyramidal neurons have a strong

association with a specific location in the environment. Hence they are called hippocampal place cells. Subsequently, a large number of experimental studies have been conducted to explain the place-specific excitations in the neurons of subregions of the hippocampus prominently in Cornu Ammonis (CA1, CA3) and Dentate Gyrus (DG) [119, 124, 125, 126, 127, 128, 129].

Retrosplenial Cortex

The cortical area of the brain located towards the back (posterior) is called the retrosplenial cortex. It plays a role in many aspects of memory and cognition [130, 131, 132, 133, 134]. The neural connectivity and structural location of RSC in the brain allow it to play a translational role in spatial navigation. The RSC neural structure is heavily interconnected with the regions that produce different FoRs, i.e. allocentric, egocentric, and routecentric FoRs [135, 136]. Therefore it facilitates the transformation of information between different FoRs. It has been reported that a subpopulation of RSC neurons exhibits activity patterns associated with egocentric, routecentric, and allocentric FoRs simultaneously [27].

Posterior Parietal Cortex

The region of parietal neocortex located behind the primary somatosensory cortex is called the posterior parietal cortex. The PPC plays a significant role in spatial navigation, planned movement, and attention. Damage to the PPC region impairs attention, eye movement, and spatial memory [137]. The PPC plays a significant role in transforming and mapping different spatial relationship. It has been identified as a cortical hub due to its important afferent and efferent connectivity with many other cortices and cortical regions [138].

Several studies on monkeys suggested that neural structures in parietal cortex generate spatial functions such as localising sensory stimuli,

motor actions, and attention [139, 140, 141, 142, 143]. The segregated spatial functions are integrated through afferent and efferent connections of the parietal cortex. It has been suggested, based on studies in monkeys, that neurons in the parietal cortex exhibit activity patterns associated with self-motion and allocentric information [142, 144, 145, 146]. In rats, it has been observed that firing patterns of parietal neurons are directly associated with locomotor behavior, spatial orientation, and spatial position. These patterns allow the brain to update allocentirc spatial representations as the animal moves [134, 147].

The role of the PPC in integrating spatial information across multiple frames of reference is illustrated by a study by Nitz and colleagues [28]. They designed squared spiral tracks such that animals' position always remained within three FoRs simultaneously. The spiral tracks consisted of five uninterrupted loops. These loops have identical shape but varying length (gradually increasing or decreasing). Each loop has four segments of track. This structure of track generates repeated action sequence at the level of segments and loops. These segments and loops combined to make a complete route. These structures are shown in Fig 2.6.

They observed that PPC neurons exhibit fast and slow firing patterns. These firing patterns have associations with spatial locations and persist across trials. A subgroup of PPC neurons exhibit firing patterns in connection with locomotor behavior. These firing patterns recur as the locomotor behavior is repeated across the segments of the loop. Moreover, the intensity of firing patterns related to turn-behavior depends on the track position. They observed that PPC neural firing patterns are independent of head direction, velocity of the animal, track rotation, and darkness.

Based on these experimental results they inferred that PPC neurons log the animals' position with respect to the three FoRs simultaneously. They observed that firing activities for different loop segments can be differentiated based on their shape or magnitude. A population of PPC neurons exhibits firing patterns with respect to whole route position. Therefore

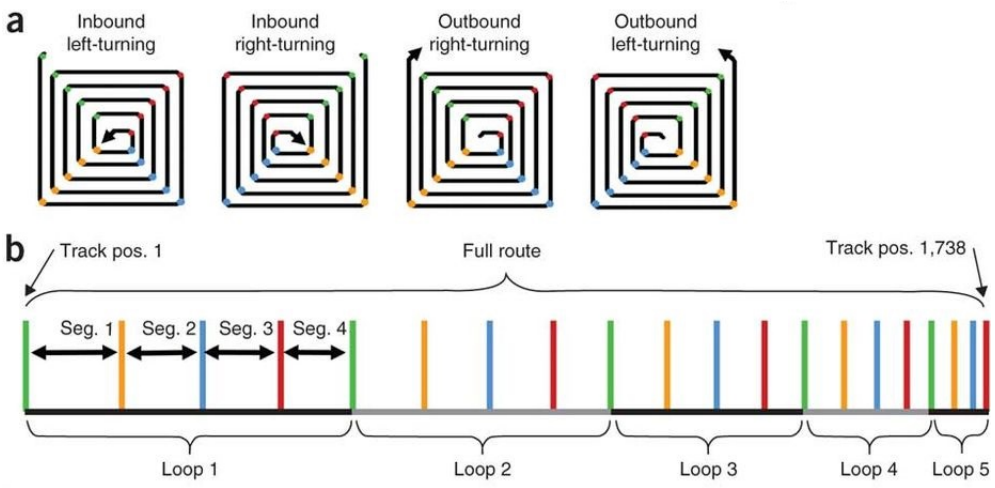


Figure 2.6: (a)Route schematics (b) Track position w.r.t. segment, loop, and complete route (source: [28])

loop and route position can be distinguished based on the firing patterns of PPC neurons. Finally, they concluded that PPC neurons have distinct logging patterns with respect to segments and routes, i.e. smallest and largest external FoRs. These firing patterns collectively make a pattern with respect to the loop, i.e. the middle FoR. These neural activities with respect to multiple external FoRs reflect the procedure that can be used to relate small units to the whole route.

Entorhinal cortex

The entorhinal cortex is a segment of the medial temporal lobe of the brain. It has dual connections with hippocampus and neocortex regions. It plays the role of a hub for the processing of memory and navigational information. The integration of entorhinal cortex and hippocampus play a significant role in spatial as well as declarative memories. Hafting et al. discovered grid cells in entorhinal cortex and hypothesized that entorhinal cortex neurons log a triangular grid map for spatial navigation [148]. Moreover,

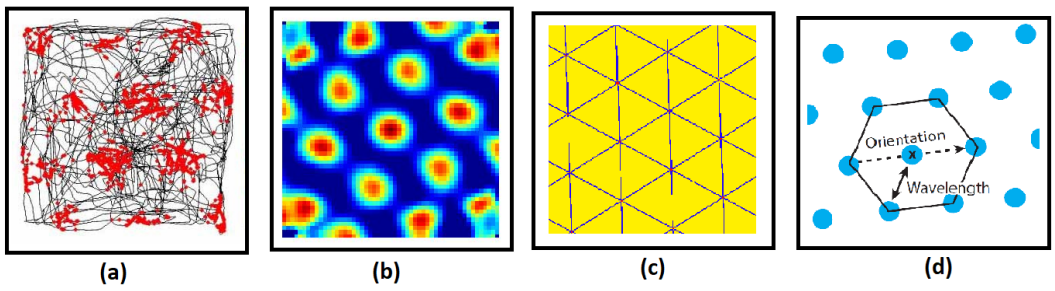


Figure 2.7: Grid Cells: (a) logging, (b) autocorrelogram patterns of grid field, (c) triangular, (d) hexagonal. (source: [155, 156, 157])

these neurons log general information in the current context of the environment, e.g. directional activities with respect to the environment. The hippocampus place cells utilize this information to generate location specific representations [149].

A grid cell is an entorhinal cortex neuron whose firing activities at different locations in space forms a triangular grid. This triangular grid is utilized by a brain's coordinate mechanism for spatial navigation. These cells log Euclidean space and form a positional system within the nervous system. Grid cells' activity patterns are independent of visual inputs and they persist even in darkness. Moreover, the triangular grids align themselves with respect to external cues. Any change in angular positions of external cues results in an analogous change in the grid pattern. Grid patterns remain the same for an environment, whereas, they maintain their spacing and relative offset in a different environment. Grid cells exhibit firing patterns across the whole environment, whereas, place cells firing activities are limited to a specific location within the environment. At a holistic level, this triangular grid of firing activities forms a hexagonal pattern, as shown in Fig 2.7. These firing units of triangular or hexagonal patterns are equally spaced and they are positioned in such a way that they have a separation of 60° [148, 150, 151, 152, 153, 154].

Inspired by the above-mentioned understanding of learning in rat brains, a novel strategy will be developed to solve navigational problems. The developed artificial agent will be able to navigate the spatial environment by incorporating egocentric, allocentric, and route-centric FoRs analog to the strategies adopted by rats during spatial navigation. The FoRs based strategy for navigation needs to be applied in the algorithms created here (see Chapter 7).

2.1.4 Benefits and Costs of Lateralization

Given that lateralization is ubiquitous in brains, some evolutionary benefits can be assumed, at least in some domains. But that does not mean those benefits extend to all domains. The research community has been struggling to determine the trade-off between the benefits and costs of lateralization. It has been hypothesized that lateralization has benefits that may counterbalance its costs [82, 158, 159]. Across domains, lateralization has been associated with both poor and good performance [160, 161].

The relationship between lateralization and cognitive performance many depend on the task at hand. Some tasks show that better performance is associated with greater lateralization. For example, lateralized chicks have better performance to detect the model predator while searching for food as compared to the non-lateralized chicks [31, 43, 81]. However, lateralization is not universally advantageous, and may entail costs [162]. It has been hypothesized that lateralization has benefits that may counterbalance its costs [82, 158, 159]. The benefits of lateralization relative to its costs is still a debatable topic.

Handedness is a physical manifestation of lateralization so could provide insight into the relationship between lateralization and cognitive performance. It is the preferential use of one hand that is faster, better, more capable, and gives a more precise performance on manual tests [163]. Although most (i.e., 90%) of humans are right-handed, there is individual variability in both the direction and degree of handedness. The relation-

ship between handedness and cognitive abilities has been explored by several studies but is ambiguous yet [164]. For example, cognitive task-based higher performance has been associated with the right-handers [165, 166]. However, the performance difference has also been associated with the strong and weak hand preference, rather than the left- or right-handers. That is the higher performance is associated with handedness strength rather than handedness direction [167, 168].

Similarly, the relationship between cognitive performance and behavioral lateralization (behaviourally assessed hemispheric asymmetry) is equivocal. The performance of verbal tasks, assessed with dichotic listening, is positively associated with the left-hemispheric language strength [161, 169]. In contrast, the performance of verbal tasks, assessed with visual half-field representaion, is associated with the symmetric language representation [161]. One speculation is that the functions that lateralized at the start and end of the ontogenetic development exhibit a positive laterality-performance correlation, whereas, the functions that lateralized at the intermediate stage exhibit a negative laterality-performance correlation [160].

Inter-hemispheric interaction is another factor that can be considered when investigating lateralization. Brain hemispheres primarily communicate with each other through the corpus callosum. The connections between them can be excitatory or inhibitory. The excitatory signals are important for the transmission of information and allow integration. Consequently, increased inter-hemispheric communication results in weaker independent lateralization. However, inhibitory signals are also important enabling one hemisphere to dominate processing depending on task goals. Consequently, increased inter-hemispheric communication can result in strong lateralization [44, 170].

From the above discussion, it is clear that although there is ample evidence that cognitive performance and lateralization are associated, there is inconsistency regarding the nature of this relationship, i.e. whether later-

alization is positively or negatively correlated with cognitive performance, or has no effects. These inconsistencies may arise because of trade-offs between the costs and benefits of lateralization, and which dominate in a given task. This thesis will investigate lateralization in artificial agents' decision-making to obtain evidence of the trade-off between benefits and costs from artificial intelligence (AI) in order to inform cognitive neuroscience.

Research communities have been creating artificial cognitive architecture, frameworks, and models that can be used to create AI systems. The next section provides an overview of the artificial cognitive architecture.

2.2 Artificial Cognitive Architecture

An artificial cognitive architecture provides an appropriate abstraction for defining a standard model of the human mind that can be used to develop human-like artificially intelligent systems [171, 172, 173]. It explains structures, mechanisms, functions, and general settings that are required to yield intelligent behavior while solving complex problems. There is no clear definition and general principles, agreed upon, to create a cognitive architecture. Each cognitive architecture has been created with a particular set of assumptions and premises [174]. Recently, efforts have been made to create a standard model of mind [171], see Fig. 2.8.

A cognitive architecture may resemble an artificially intelligent system because it has data representation, memory storage, control components, and input/output devices. Generally, an AI system is developed as a fixed model to solve a specific task(s). In contrast, a cognitive architecture, generally, has the ability to update itself through development and utilize learned knowledge to solve new tasks [175].

Cognitive architectures have been created from two perspectives, i.e. scientific and engineering. According to the scientific approach (or more precisely psychological approach), a cognitive architecture models human

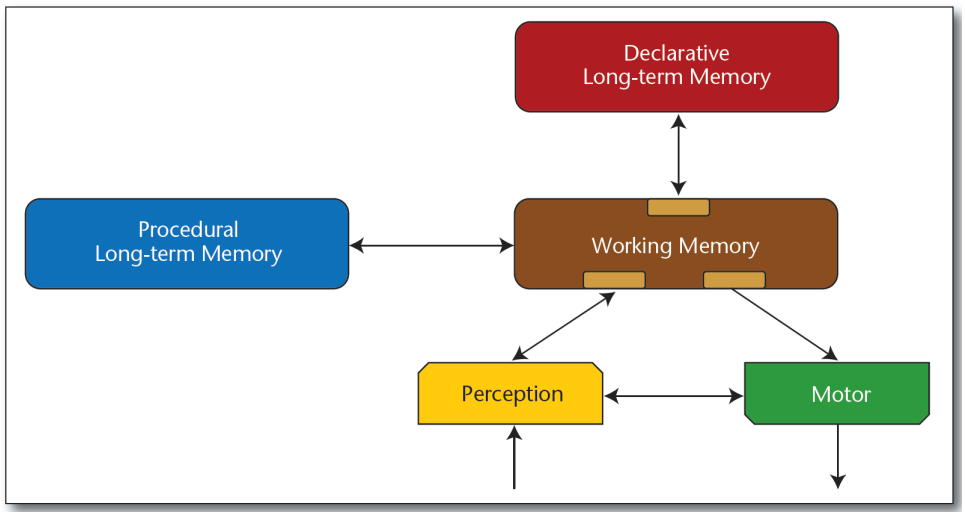


Figure 2.8: A standard model of the mind (Reproduced from [171])

behavior and underlying cognitive processes to facilitate the investigation of the human mind [172, 176]. In contrast, according to the engineering approach, a cognitive architecture provides a structure of mental representations and mechanisms that can be developed to enable intelligent behavior to solve complex tasks [177, 178, 179, 180]. Moreover, the engineering approach uses cognitive principles to achieve the best outcome, even if it does not model the human (or animal) mind.

The important components of an artificial cognitive architecture are presented below.

2.2.1 Perception

Perception is an important component of cognitive architecture. It provides a mechanism to convert an environmental signal into an internal representation. Different systems may have different perception modules, or a system may have multiple perception modules. These modules process various types and varying amounts of data, depending on the nature and complexity of the environmental signals. The most common sensory

inputs are: touch, hearing, vision, smell, proprioception, and non-human senses (keyboard, graphical user interface, sensors). Moreover, some of the sensory inputs are generated by using simulations, e.g. vision is one of the commonly used perceptions and the majority of the systems use simulation-based input signal instead of actual camera captures [174].

2.2.2 Attention

The process of selecting the relevant information and ignoring the irrelevant information from the environmental signal is called attention or perceptual attention. Attention plays a critical role in cognitive processes [181]. It acts as a bottleneck that restricts the information available for further processing. Attentional elements can be grouped into the following three classes of information reduction mechanisms: *selection*, these mechanisms are used to select one from many; *restriction*, these mechanisms are used to choose some from many; and *suppression*, these mechanisms are used to suppress some from many [182].

2.2.3 Action Selection

The procedure to determine the next course of action, based on the current situation, is called action selection. It could be divided into two parts, i.e. decision making and motor actions (decision execution) [183]. Planning and dynamic action selection are two major approaches to action selection. In planning, AI algorithms plan to solve a given problem or identify a sequence of steps to reach the goal before actual execution. In dynamic action selection, AI algorithms select the best action and execute it. The best action is selected, based on the defined criteria, by utilizing the current knowledge.

2.2.4 Memory

Memory is an important component of a cognitive architecture. It can be used to store, retrieve, and maintain information. This information is used by an AI system, for different purposes at different stages, while solving a problem. Depending on the nature and requirement of use, memory can be divided into different types. Long-term memory, which includes declarative memory, is used to store concepts, facts, and problem-solving rules. It has the following three subtypes: semantic memory to store factual information, procedural memory to store condition-action information, and episodic memory to store experienced information. Short-term memory is used to store the current state of the model or information of the goal stack. It has following two sub-types: sensory memory, also known as perceptual memory, to store recent percepts; working memory to temporarily store information that is associated with the current task or relevant to the current focus [184].

2.2.5 Learning

Learning is the ability of an AI system to enhance its performance based on experience. It can be divided into two major types, i.e. declarative and non-declarative. Declarative learning involves explicit knowledge acquisition, whereas, non-declarative learning includes procedural, perceptual, associative, and non-associative learning [185].

2.3 Artificial Learning Methods

The intelligence exhibited by machines based on the principles of natural intelligence is called Artificial Intelligence. The majority of AI-based systems employ artificial agents to execute various tasks. An artificially intelligent agent often receives environmental input, analyses it, and generates appropriate action that maximizes its current or future reward. Moreover,

an artificially intelligent agent may have the ability to solve a problem in such a way that it mimics human cognition [8].

Machine learning (ML) is an important field of AI. A large number of machine learning applications have been developed that are playing a significant role in many aspects of everyday life [186, 187, 188]. ML based applications have the ability to interact with their environments, automatically extract useful knowledge patterns, and learn through experience [189].

The learning process in ML can be divided into three main categories based on the nature of the feedback, i.e. (i) supervised learning, (ii) unsupervised learning, and (iii) reinforcement learning [190]. (i) In supervised learning, the artificially intelligent agent is trained with the set of data such that the ground truth is known in advance. Consequently, the agent learns a generalised function that maps inputs with the desired outputs [191]. Some of the well-known examples of supervised learning methods are: artificial neural networks, naïve Bayes, and decision tree. (ii) In unsupervised learning, the artificially intelligent agent is trained with the set of data such that the ground truth and reward are not available. The agent learns the hidden patterns in the given data in the form of groups and generates a model to predict the future instances. K-means clustering [192] and hierarchical clustering [193] are two well-known examples of unsupervised learning methods. (iii) In reinforcement learning, the artificial agent directly interacts with the environment. The agent affects the environment by generating the actions against the input data. Consequently, the environment affects the agent through feedback in the form of reward or punishment based on the suitability of the agent's action. The agent learns by generating the actions in such a way that maximizes the future rewards from the environment [194]. Temporal difference learning is a well-known example of a reinforcement learning method [191, 194, 195].

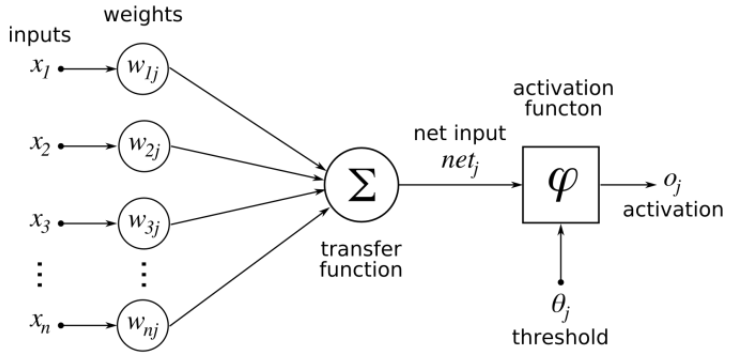
The novel lateralized systems for Boolean and navigation problems will be developed by applying the reinforcement learning strategies for

three main reasons. First, in a majority of the daily life tasks, the natural cognitive systems learn in the absence of ground truth information. They adapt their behavior based on the feedback (reward or punishment) from the environment. Second, the majority of experimental studies conducted in cognitive labs are based on a reinforcement learning strategy. Finally, a reinforcement learning based system has the ability to learn from inexact input data or from a noisy environment. The novel lateralized system for computer vision (CV) problems will be developed by applying the supervised learning strategy because the majority of visual classification systems apply supervised learning, especially those using deep networks. This thesis takes inspirations from neuroscience to create AI systems for different problem domains. The following section presents an overview of neuroscience inspired AI.

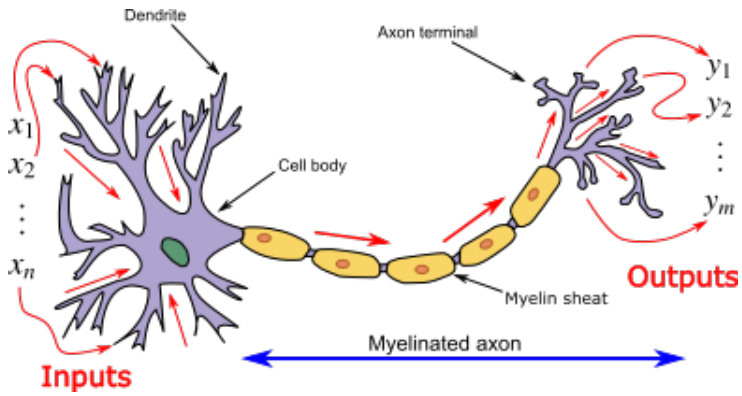
2.3.1 Neuroscience Inspired AI

Neuroscience has been a major source of inspiration to create AI systems. It has been playing a key role to create a wide range of AI systems ranging from the systems that emulate human-like intelligence to the systems that mimic brain structure [171, 196]. AI systems are generally created to efficiently and precisely solve specific tasks. Cognitive neuroscience is used as inspiration to create AI systems for problems where biological intelligence outperforms AI. Instead of reproducing the underlying mechanism or neurological architecture, AI focuses on understanding natural intelligence, takes inspiration at the algorithmic level, and develops solutions. For example, biological neural networks have been used as inspiration to create mathematical function based artificial neural networks, see Fig. 2.9.

The biological brain is not an amorphous system. It is made of different modules such that each module solves a specific problem or a part of a problem. Hence, the brain can be understood at different levels and/or perspectives, which have impacted AI [198, 199]. For example, the phenomenological perspective has been used as inspiration to create AI sys-



(a)



(b)

Figure 2.9: An artificial neuron is a mathematical function conceived as a model of biological neurons (source [197])

tems that utilize functionality such as episodic memory [200], imagination [201], attention [202], continual learning [203], and transfer learning [204]; the mechanistic perspective has been used as inspiration to incorporate neural inspired elements in standard training techniques, e.g. recurrence in visual processing [205, 206, 207, 208, 209]; the biophysical perspective has been used as inspiration to create AI systems such as artificial neural networks [210, 211, 212], spiking neural networks [213], spine stabilization [203], neurogenesis [214], and context-dependent activation [215, 216].

The efficacy of neuroscience on AI can be extended by considering the neural systems at an abstract level and paying attention only to the relevant principles. For example, a predictive gain modulation has been observed in the neurons of the dragonfly visual system. It results in selective enhanced visual responses to predicted prey-position [217], even in the presence of another potential target [218]. From a phenomenological perspective, this selective enhancement in visual responses in dragonfly's visual system has a resemblance with the selective visual attention in macaque's visual cortex [219, 220]. These visual systems (both the dragonfly and non-human primate) are still under investigation to further explore the visual mechanisms and underlying neural circuitry. The abstract level representation of the dragonfly visual system has been used as inspiration to create function level models [221] and develop target tracking algorithms for robotics [222, 223, 224].

AI systems can be developed by taking inspiration from neuroscience, based on the understanding of the neural systems at that time. At a later stage, the AI systems can be enhanced and modified to accommodate the novel findings in neuroscience. For example, place cells [123] and head-direction cells [225] were initially observed in the hippocampus where they play a role in spatial navigation. The hypothetical functional descriptions of these cells were used as inspiration to create navigation algorithms [226, 227]. Later on, there are novel discoveries in neuroscience to explain navigation such as grid cells [228, 229, 230], $3-$dimensional repre-

sentation [231], and a general framework for navigation [232, 233]. These advances have been incorporated into the AI systems to create novel semi-autonomous and autonomous navigation systems [196]. This thesis aims to create novel lateralized AI systems inspired by the principles of biological intelligence, derived from the current understanding of learning mechanisms implemented in the brains of human and non-human animals.

Evolutionary computation is an important field of AI that takes inspiration from the natural evolutionary process. The following section briefly discuss the aspects of this technique that are relevant to this thesis.

2.3.2 Evolutionary Computation

Evolutionary Computation (EC) is a paradigm consisting of population based problem solving techniques. It is a methodology that has been used to develop cognitive systems based on the principle of biological evolution. Inspired by the natural evolution process, these techniques evolve a population of fittest candidate solutions based on the principles of biological and Darwinian evolution. The evolutionary process consists of selection, reproduction, updating, and deletion mechanisms. Generally, the population is initialized by randomly generated individuals. Subsequently, the evolutionary process evolves the population of fitter candidates based on the fitness criteria [234, 235]. Learning classifier systems (LCSs) are a state-of-the-art rule-based machine learning technique. Genetic algorithms (GA) and genetic programming are two important EC strategies that have been used for both the basis of representation and rule discovery in LCSs [51, 236, 15]. The novel lateralized systems will be developed by utilizing an evolutionary computation framework of LCSs due to their niche-based algorithm and built-in support for heterogeneity (i.e. different rules co-exist in the same population). The relevant EC approaches are presented below.

Genetic Algorithm:

Genetic algorithm is a search based heuristic method that works on the principle of natural selection [237]. GA can generate fitter solutions for optimization problems by applying biologically inspired operators such as selection, cross-over, and mutation. The evolutionary process in GA can be explained by utilizing the schema theory. A schema is a template for describing the states of an individual member by utilizing an alphabet. The set of alphabets often consists of specified and 'don't care' symbols. For example, if the set of the alphabet is ternary $\{0, 1, \#\}$ then 0 and 1 are specified symbols, whereas # is 'don't care' symbol that can be matched for both 0 and 1. Goldberg utilizes the concept of schema for the comprehension of the behavior and performance of GA [238]. He observed that the combination of the short-length and high-performance schemata generate high-performance individuals. He termed these schemata as building blocks of knowledge. However, he found that these building blocks can only be generated if there is low epistasis[1] among the genes.

The normal process of GA consists of the following steps. Initially, a population of candidate solutions is randomly generated. Then a selection operator is applied to the population for the selection of fittest solutions. Consequently, the crossover and mutation operators are applied to generate offspring solutions. These newly generated solutions are hypothesised to be fitter than their parents. Moreover, to keep the population size in a given bound, a deletion mechanism is applied that removes the weaker solutions from the population. Finally, the evolution process of GA is repeated until the ending criteria are met [238].

[1]A problem is said to exhibit epistasis if the value of one of its features affects the importance of another feature [239].

Learning Classifier Systems

Learning classifier systems are a concept for developing a rule-based machine learning technique by applying discovery algorithms and learning components. LCSs seek to identify innate patterns in the data that they encounter. They can develop context-dependent rules based on those patterns. These rules collectively provide knowledge of the environment. Moreover, LCSs make predictions by applying this knowledge in a piecewise manner [240, 241]. Holland and Reitman developed the first LCS and named it CS-1, i.e. "Cognitive System One" [242]. It is one of the earliest AI systems that was developed on the principles of cognition.

Generally, LCSs are analogous to a rule-based artificial agent that integrate machine learning and evolutionary computing to devise a solution for the given problem. An LCS based artificial agent communicates with the (initially unknown) environment through input sensors and output effectors. It observes the current state of the environment and performs a suitable action concerning the ongoing situation. Consequently, it receives a reward from the environment. The agent learns by applying a strategy to take appropriate action that maximizes its current or future rewards. LCSs evolve a population of classifier rules that collectively solve the given problem. Moreover, to keep the population within a bound, LCSs apply different condensation and compaction techniques to remove the weak, and redundant classifier rules from the population [243]. LCSs have the ability to handle epistasis and heterogeneity within a problem.

A broad range of LCS based AI applications have been developed in different domains such as behavior modeling, data mining, classification, modeling, regression, optimization, and approximation problems. LCS based applications can exhibit robustness against a small amount of noise, whereas, against a large amount of noise, these applications exhibit more robustness as compared to other AI-based applications [244].

The LCS paradigm can be comprehended by explaining its four conceptual components, i.e. learning, classifier, system, and problem proper-

ties. Learning plays a critical role in AI-based applications. Michalski et al. provide an excellent definition of learning, i.e. "Learning is constructing or modifying representations of what is being experienced" [245]. Learning is the process of acquiring knowledge through interaction with the environment. It can be done off-line by utilizing already saved data or online through direct interaction with the environment at runtime. Generally, learning is influenced by the reward received from the environment as the system learns to improve that reward. For better performance of LCSs, the input data is supposed to contain patterns, ideally generalizable, as this is a prerequisite for learning in any AI-based application. In the absence of such patterns, LCSs cannot learn. LCSs form a group of learned rules, which is called a population. These rules have fitness based on their contribution toward the optimized solution. In the learning process, LCSs improve the strength of good rules, produce new fitter rules, and remove the rules with poor strength. Instead of a single rule, a set of rules constitute a solution for the given problem.

Traditionally, assigning a specific class to the given data instance is referred as a classification. The number of available classes is generally known before learning. For example, there are two classes in a multiplexer problem, i.e. '0' and '1'. In LCSs, the term 'action' is used to refer to the predicted class of a data instance. LCS utilizes rules to decide actions based on the condition, i.e. if 'condition' then 'action'. A rule has fitness value based on its usefulness. These rules are the fundamental building blocks of LCS paradigm. A rule itself does not provide information about the accuracy of 'condition-action' mapping and its usefulness for the population of rules. Therefore rules have associated statistics to provide all of this information. The important rule statistics are fitness, error, reward prediction, numerosity, lifespan, reproduction, and experience. A rule with its statistics is referred to as a 'classifier'.

A system can be defined as a bounded structure that takes inputs and generates outputs. In LCSs, feedback from the environment plays a crit-

ical role to generate appropriate and fitter rules. A system, with respect to LCSs, has four main components, i.e. (i) rule condition matching with input, (ii) best action selection, (iii), evaluation of the rule, and (iv) generate fitter rules. The research community has developed many variants of each component. This variation supports the hypothesis that *LCSs are a concept rather than a technique*. LCSs interact with the environment to map domain areas with separate classes. Moreover, LCSs have different sample space and search space. The number of unique instances of a problem domain is called sample space, whereas, the number of unique rules that can be created to map the sample space is called the search space. For example, a 6-bit multiplexer with ternary alphabet representation has, 2^6 sample space and 3^6 search space.

LCSs Functional Cycle

A typical functional cycle of LCSs consists of nine steps that could be repeated for a fixed number of iterations or until a stopping criterion is achieved. A schematic diagram of the LCSs functional cycle is shown in Fig 2.10. The LCSs algorithm executes these steps in a sequence mentioned by numbers in small circles. (i) In the first step, the LCSs get an input instance from the environment. This input is selected randomly from the input dataset. The selection process runs on the whole dataset in a cyclic way. Moreover, the selection of the instances is not repeated in a cycle, i.e. all the instances are selected only once in a single cycle. (ii) The selected input instance is passed to the population of classifiers. The population of the classifiers [P] is significant component of LCSs paradigm. It contains all the classifiers generated and preserved by the system. However, the maximum size of [P] is defined by the user. Initially, the [P] has no classifiers and new classifiers are added through covering. The covering is a mechanism that generates new classifiers whenever there is no classifier in [P] that matches the condition and class of current training instance. The covering method creates a new classifier of the same class as that of train-

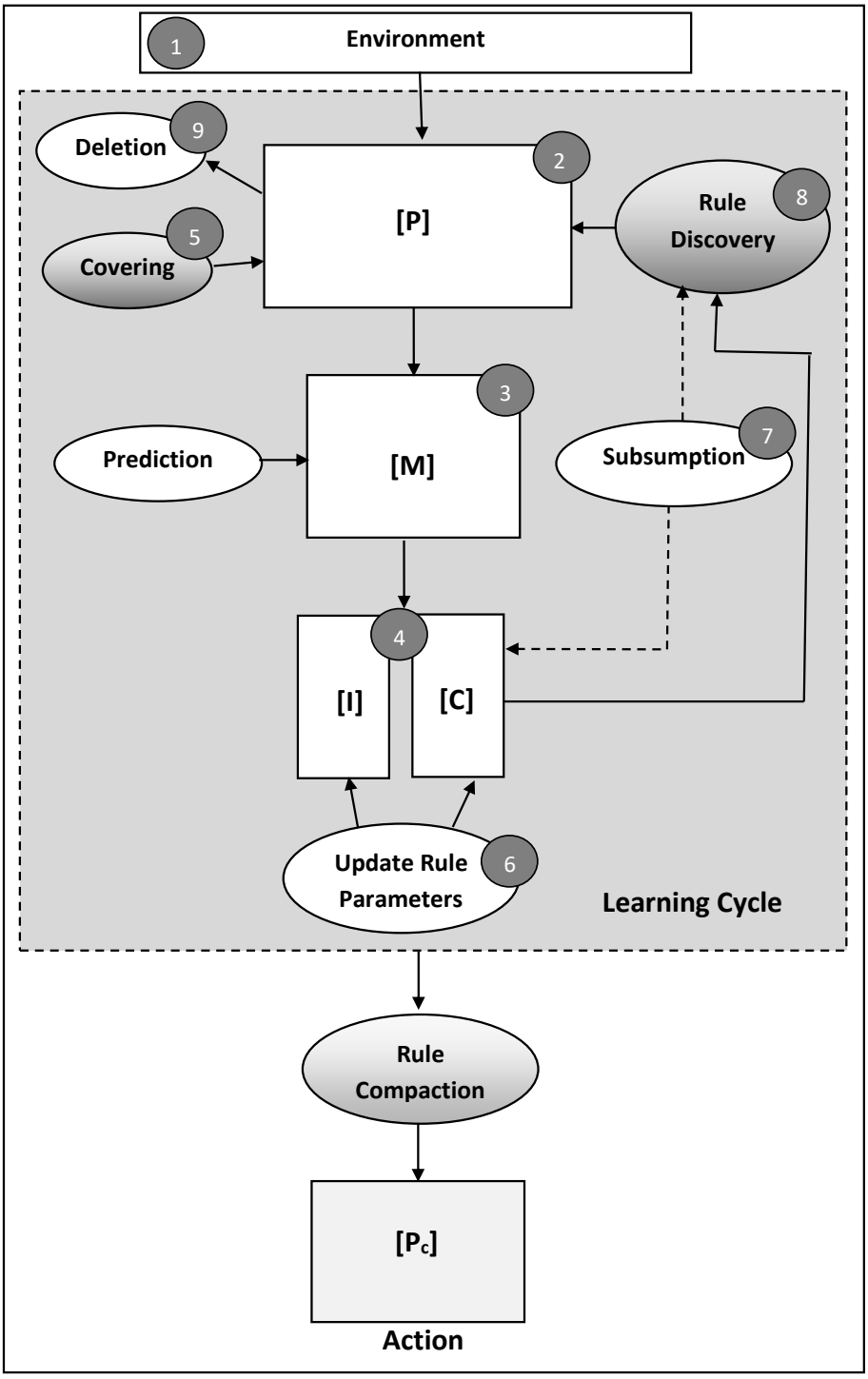


Figure 2.10: Schematic of Michigan style LCS algorithm with supervised (adapted from [239])

ing instance in supervised learning, randomly otherwise. Moreover, this classifier has a randomly generated conditions (including any "don't care" symbols for generality) that matches the conditions of the current training instance in supervised learning, randomly otherwise. (iii) The match set [M] consists of all the classifiers from [P] whose conditions match with the attributes (features/states) of the current instance. (iv) The classifiers of [M] are divided into different classes based on their actions. In the specific case of a binary classification in supervised learning, they are divided into correct [C] or incorrect [I] class by matching their actions with the class of training instance. This can only occur in supervised systems, such as UCS [246]. (v) If none of the classifier from [M] have the same class as that of current instance, i.e. [C] is empty, then covering is activated and a new classifier is generated. (vi) The parameters of the classifiers in [M] are updated based on different strategies e.g. in supervised classifiers the fitness of a classifier is a ratio of its appearance in [C] to its appearance in [M]. (vii) The majority of the LCSs apply subsumption techniques. It is a strategy to remove the overspecific classifiers by merging them into more general and accurate classifiers. (viii) Rule discovery technique is applied at this step. Genetic algorithm is one of the commonly used technique in LCSs. In the majority of LCSs algorithms, GA is applied at [C] instead of [M] or [P]. It is called niche GA as it helps to explore the unique niches in the problem space. (ix) Finally, the weak classifiers from [P] are deleted if the size of [P] is greater than the user-specified limit.

LCS Approaches

LCSs have been developed by applying two different approaches, i.e. Michigan approach and Pittsburgh approach [247, 248]. These approaches have a different algorithmic architecture. The Michigan approach based LCSs evolve a population of cooperative classifiers such that each individual classifier represents a distinct and unique rule with associated parameters. Therefore, GA is applied to individual classifiers. These systems can

perform on-line as well as off-line learning. Moreover, these systems have incremental learning and the whole population of the classifiers produces the learned solution. This approach is different from other evolutionary computation techniques, because it does not require population initialisation [239]. In a majority of the cases, LCSs start learning with empty population of rules. Covering is applied to generate the matching rules if no existing rule match the current problem instance. Initially, LCSs were developed by applying the Michigan approach and to date, the majority of LCSs are based on this approach. Pittsburgh LCSs evolve a population of rule-sets such that each rule-set forms a classifier with associated parameters. Therefore, GA is applied to rule-sets. Generally, these systems perform batch learning and the best rule-set from the population is selected as the learned solution.

The strategy applied to evaluate the fitness of a classifier plays a significant role in the performance of LCSs. There are two main types of LCSs based on the technique they apply to compute the fitness of the classifier i.e. strength-based LCSs and accuracy-based LCSs. Strength is a measure of the reward received from the environment and the fitness of the classifier is computed based on the value of the reward. This strategy can balance the classifiers in case of infrequent niches. However, it yields overgeneral classifiers that decrease the performance in a majority of cases. ZCS is a good example of Michigan style strength-based systems [249]. However, the consistency in prediction is often a more important factor than the value of the reward. The fitness of the classifier is computed based on the consistency of predicting the reward correctly. This strategy does not produce overgeneral classifiers and leads the system to generate maximally general as well as accurate classifiers. The resultant solution contains a complete mapping and has more building blocks of knowledge as compared to ZCS. XCS is the most common example of Michigan style accuracy based systems [250, 251].

2.4 Associated Techniques

This section presents a brief introduction to the associated computer vision techniques (i.e. image features and adversarial attack) that are used in this thesis (see Chapter 6). Often, image features have been used as input instances for the classifiers. A feature can be considered as a piece of information about the visual contents of an image. The number of features are few as compared to the large image data (raw data of an image) [252, 253]. Adversarial attacks have been used to generate noisy, redundant, and irrelevant data [55, 52, 56]. The resultant images can be used to evaluate the robustness of classification techniques.

2.4.1 Features

The scale-invariant feature transform (SIFT) is one of the powerful techniques that has been widely used for image classification and object detection [254]. The SIFT descriptor performs image measurements in the form of receptive fields over which local scale selection is used to establish local scale-invariant reference frames. The SIFT features are invariant to orientation, uniform scaling, and illumination changes. Consequently, a SIFT feature descriptor exhibits robustness against disruption by clutter, occlusion, or noise [255, 256].

The histogram of oriented gradients (HOG) is another widely used and well-recognized descriptor in computer vision [257]. A HOG descriptor is generated by computing the pixel-wise histograms of gradient directions. This distribution of local gradient empowers the HOG descriptor to accurately detect complex shapes and object deformation, e.g. muscle movements and facial expressions [258]. Moreover, they are invariant to color variation, light conditions, and geometric transformations. The inclusion of these features will assist the novel system to correctly classify objects based on facial expressions.

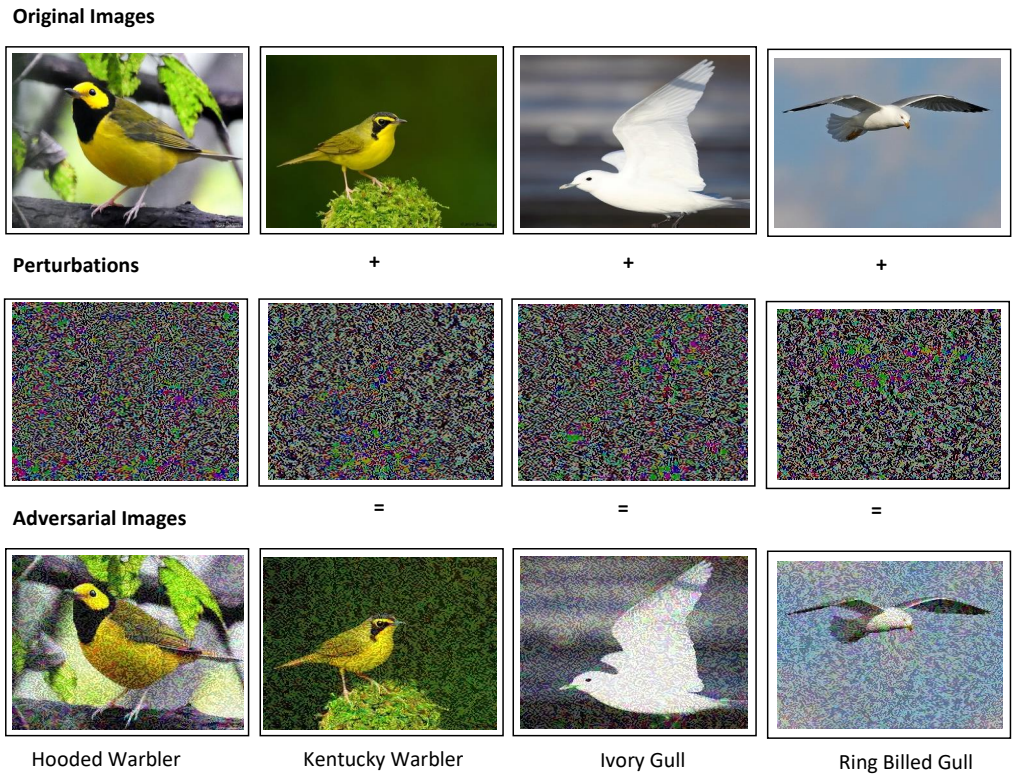


Figure 2.11: Example bird images of four different species. Original images are in the first row. The relevant adversarial perturbations are in the second row. The resultant adversarial images are in the third row.

2.4.2 Adversarial Attacks

An adversarial attack is a technique that attempts to fool AI models by generating deceptive input [53, 54]. The fast gradient sign method (FGSM) is one of the well-recognized and commonly used methods to generate adversarial attacks. Deep networks encourage linear behavior for efficient learning. However, the images generated by FGSM exploit the linearity of a deep model in higher-dimensional space. The FGSM generates perturbations to increase the loss of the classifier. For this purpose, it performs one-step gradient updates at each pixel [55].

In many real-life applications, the input data is not directly available to the deep networks; instead it is passed through devices (e.g. sensors, cameras). The iterative adversarial method is proposed to generate adversarial attacks for real world applications [56]. Instead of a one-step, the iterative method extends FGSM to make a small change (finer optimization) in multiple iterations. The images generated by FGSM and Iterative methods will be used to evaluate the robustness of the novel system. Sample intact and adversarial images are shown in Fig. 2.11.

2.5 Chapter Summary

The basic concepts and terminologies of cognitive neuroscience that are relevant to this thesis have been reviewed in this chapter. This chapter presented a brief overview of hemispheric lateralization, semantic knowledge, and frames-of-reference. Moreover, it described the trade-off between the benefits and costs of lateralization in biological intelligence. This chapter discussed the relevant aspects of natural cognitive architecture that will be used as inspiration to create novel methods in this thesis. It also presented an overview of artificial cognitive architecture. Then, this chapter reviewed the current state of knowledge of the artificial learning methods that will be used as a foundation to develop more modular and lateralized systems in this thesis. Finally, associated techniques to interact with visual environments and test the robustness of classification techniques are presented.

This thesis will create a lateralized framework that will be adapted to develop lateralized AI systems for Boolean, computer vision, and navigation domains. The next chapter will present a range of benchmark problems, from different domains, that will be used to obtain proof-of-concept and to evaluate the effectiveness and robustness of the novel lateralized systems. Moreover, it will provide an overview of the benchmark techniques that have been developed to address those complex problems that will be used to evaluate this thesis.

3

Benchmark Problems and Approaches

Benchmark problems and approaches are an important part of the evaluation process in the field of artificial intelligence. They are used for a wide range of purposes during the development of an artificial intelligence system. For example, benchmarks are used to obtain a proof-of-concept, to evaluate the generalizability of solutions/approaches, to determine the advancements between state-of-the-art methods and novel methods. Moreover, a benchmark problem may have specific characteristics that make it particularly appropriate for evaluating a specific aspect of a system such as reliability, scalability, or robustness.

The goals of this chapter are two fold. First, this chapter presents a range of benchmark problems, from different domains, that will be used to obtain proof-of-concept and to evaluate the effectiveness and robustness of the novel lateralized systems. It highlights the important characteristics of the problem domains

that are suitable to evaluate different aspects of the novel lateralized systems. Second, this chapter highlights the strengths and limitations of the existing state-of-the-art relevant approaches that have been developed to address those complex benchmark problems that will be used to evaluate this thesis.

3.1 Benchmark Problems

This section presents an overview of the benchmark problems that will be used to evaluate the novel systems developed in this thesis.

3.1.1 Boolean Problems

Boolean problems are single-step problems that have known solutions so that the exactness of the produced solution can be interrogated. These problems can exhibit heterogeneity and epistasis, which are characteristics known to cause problems in classification techniques as they can not make the assumption linear separable. A problem is said to exhibit heterogeneity if different patterns can cause the same effect, i.e. separate niches (a distinct subset of features/values) map to one action/class. A problem is said to exhibit epistasis if the value of one of its features affects the importance of another feature [239]. Boolean problems also possess identifiable components of a solution, rather than only the solution to the complete problem, that are transferable to other problems. This helps to evaluate transfer learning abilities, i.e. the ability to identify and transfer important parts of knowledge.

Boolean problems have measurable search space, dependency structure, and distributed niches (subsolutions) [244]. Although Boolean problems can be considered as 'toy' benchmark problems, the heterogeneity and epistasis characteristics make them analogous to real-world problems, such as finance, bioinformatics, and behavior modeling. These qualities

make Boolean problems an ideal test set to obtain the proof-of-concept and show the effectiveness of the lateralized approach.

The input problem instance space of a Boolean problem is restricted to be binary, i.e. $\mathcal{S} \subseteq \{0,1\}^l$ where 0 is false, 1 is true, l is the length of the problem instance. Moreover, the output of the problem instance is also commonly binary, i.e. supports two output classes $\mathcal{A} = \{0,1\}$. The well known Boolean problems (except satisfiability (SAT) problems as this category is constraint optimization rather than classification) will be used to check the validity of the lateralized system, i.e. multiplexer problems, parity problems, and carry problems. Moreover, the effectiveness of the novel system will be demonstrated by utilizing the more complex derived versions of these problems, i.e. hierarchical-multiplexer problems and higher-level parity problems.

Multiplexer Problems

The Multiplexer problems are multi-modal, non-linear, and have epistasis characteristics, i.e. address bits are critical as they determine the importance of data bits [244, 259]. A multiplexer is an electronic circuit that selects one of the several input signals and forwards it as an output signal [260, 261]. The selection of the output signal is based on the value of particular input lines, known as select lines or address lines. The number of address lines has a direct relationship with the total number of input lines. The relationship between the input lines and address lines in a multiplexer is:

$$\mathcal{L} = k + 2^k \tag{3.1}$$

where $\mathcal{L}$ is the length of the multiplexer and k is the number of required address lines. The address lines are intrinsically bounded to the data lines such that the value of address lines determines the importance of the corresponding data line [244, 259].

In a multiplexer problem instance, the value encoded by the address bits k is utilized to select one bit from the remaining 2^k bits. The value of the selected bit is considered as the output class for the given problem instance. For example, 101011 is a problem instance of a $6-$bit multiplexer problem. Here the output class is 1 as the first k bits (10) represents the value 2 which is the third bit from the remaining 2^k bits (1011). Similarly, the output classes for the problem instances $100010, 000111,$ and 110101 are $1, 0, 1$, respectively.

The multiplexer is one of the complex Boolean problems. It accepts the input strings of length given by the eq-3.1. The length of a multiplexer increases exponentially with respect to the number of address bits that increase the search space in a similar way. For example, the multiplexer with 5 address bits has a length of $5 + 2^5$ bits and its search space consists of 2^{37} possible combinations.

Parity Problems

The parity problems are non-linear and non-monotonic [262]. It is hard to make any useful generalization for parity problems domain using standard ternary alphabet based condition and static numeric action [244, 263]. The parity problems are based on the total number of ones in the given instance. The class of the given parity problem will be 0 or 1 if the total number of ones in the given instance is even or odd, respectively.

Carry Problems

The carry problems are considered as niche imbalance and the solutions to these problems have strongly overlapped rules [244]. The class of the carry problems is based on the triggering of a carry bit, as a result of the addition of two binary numbers of the same length. The class is one, if the addition triggers a carry, and zero otherwise. For example, the class of a $3, 3-$carry problem instance 101110 is 1 because the addition of binary numbers 101 and 110 trigger a carry, whereas the class of the problem instance 011010

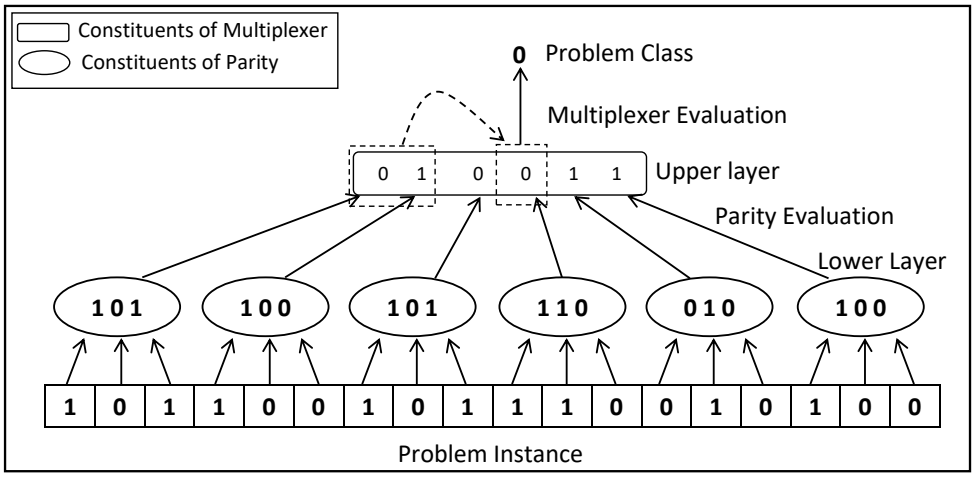
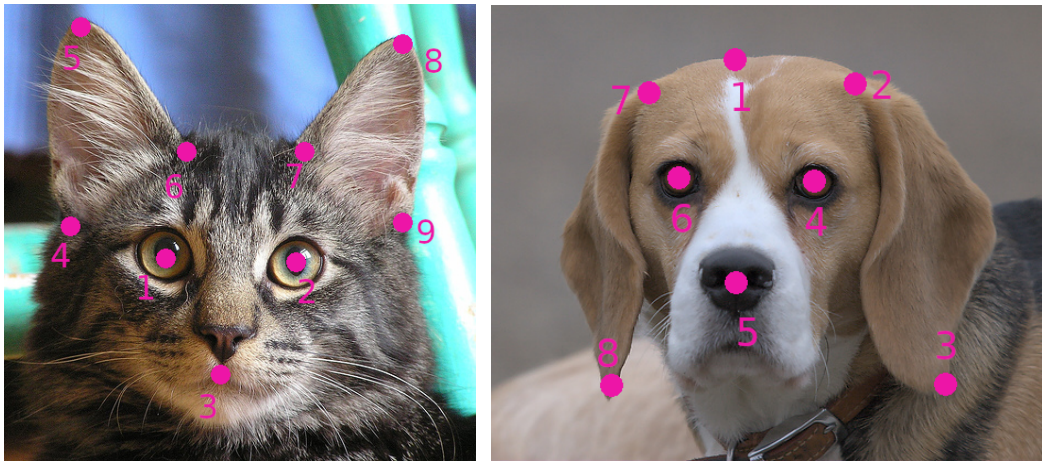


Figure 3.1: An instance of 18-bit hierarchical multiplexer problem. The lower layer consists of 3-bit Parity problem, whereas, the upper layer is a 6-bit multiplexer problem.

is 0 because the addition of binary numbers 011 and 010 does not trigger a carry.

Hierarchical Boolean Problems

Hierarchical Boolean problems have an additional layer of complexity, so they are hard to resolve as they have low sparsity and hierarchical distribution of knowledge [51, 244]. Therefore, these problems will be used to evaluate the effectiveness of the lateralized approach. The hierarchical Boolean problems consist of two layers. The lower layer is composed of multiple instances of a specific Boolean problem. The evaluation and integration of the lower layer generate the instance of the upper layer. The upper layer is another Boolean problem. For example, 18-bit hierarchical problem may be composed of the following layers (i) lower layer based on 3-bit even parity problem, and (ii) upper layer based on 6-bit multiplexer problem. An example instance of the 18-bit hierarchical mul-



(a) (b)

Figure 3.2: Sample cat and dog images with 9 and 8 points annotation of the head, respectively.

tiplexer is shown in Fig 3.1. It is important to note that an agent has no prior knowledge on how the bit string is split.

3.1.2 Computer Vision Problems

Computer vision (CV) problems are single-step visual classification tasks taken from the real-world domain. The ground-truth information is often available for these problems when humans label the data. Moreover, these problems include uncertainty, noise, and irrelevant and redundant data. These characteristics make CV problems ideal candidates with which to evaluate the robustness of the lateralized approach.

CV problems need to have ground-truth information about both the constituent level (parts) and holistic level (whole) representations. Two lateralized systems will be developed to show the scalability and effectiveness of the lateralized approach. The first lateralized system will be developed to address binary-class image classification tasks, whereas, the second lateralized system will be developed to address multi-class image

classification tasks. Publicly available cat and dog datasets will be used as exemplars to evaluate the robustness of the binary-class lateralized system. These datasets have ground-truth information about the parts (nose, eyes, ear, mouth, head, face) and the overall image. The cat dataset is taken from Kaggle competition [264]. It includes more than 9000 cat images along with ground-truth files. Each image contains 9 points annotation of the head, i.e. (1) Left Eye, (2) Right Eye, (3) Mouth, (4) Left Ear-1, (5) Left Ear-2, (6) Left Ear-3, (7) Right Ear-1, (8) Right Ear-2, (9) Right Ear-3 (see Fig. 3.2a).

The dog dataset is taken from dlib (C++ library for ML) [265], which is a modified copy (modified missed annotations and loose BBoxs) of the data used by Liu et al. [266]. It includes more than 8000 dog images along with the ground-truth information. Each image contains 8 points annotation of the head, i.e. (1)head top, (2) left ear base, (3) left ear tip, (4) left eye, (5) nose, (6) right ear base, (7) right ear tip, (8) right eye (see Fig. 3.2b).

Another publicly available Caltech-UCSD Birds-200-2011 dataset [267] will be used as exemplar to evaluate the robustness of the multi-class lateralized system. This dataset contains $11,788$ photographic images of 200 bird species. The ground-truth information about the parts and overall image is available. Each image contains 15 points annotation of the bird, i.e. (1) back, (2) beak, (3) belly, (4) breast, (5) crown, (6) forehead, (7) left eye, (8) left leg, (9) left wing, (10) nape, (11) right eye, (12) right leg, (13) right wing, (14) tail, (15) throat (see Fig. 3.3).

3.1.3 Navigation Problems

The navigation problems are multi-step path planning problems that provide virtual environments related to real-world problems. Mazes have been used in a wide variety of navigation based research from cognitive neuroscience to artificial intelligence [244, 27, 14, 123, 268, 269, 270, 271, 272, 273], as they approximately simulate real-world navigation problems. Mazes have a structure that allows experimenters to easily control and

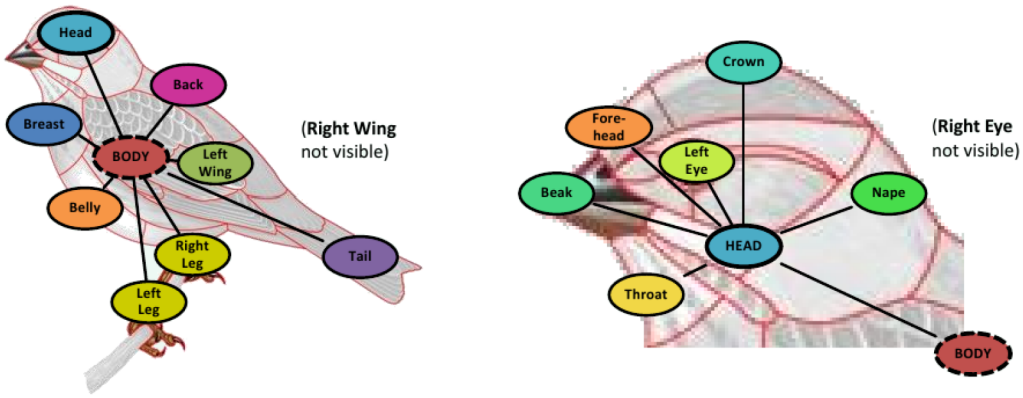


Figure 3.3: A sample bird image with 15 points annotation of the parts (source [267]).

trace the behavior of an agent during the learning process. They offer a wide range of complex environments that artificial agents struggle to solve. This includes complex non-Markov mazes that are characterized by heterogeneity in action probability in a given state and clusters of such aliased states. An artificial agent needs to consider the local viewpoint to resolve aliased states within a cluster, and a world viewpoint to uniquely identify the position of a cluster within an environment. These characteristics make maze problems an ideal research paradigm with which to evaluate the effectiveness of the lateralized approach.

A maze is a two-dimensional rectangular grid, which provides a virtual environment for navigation. This will be used to evaluate the effectiveness of the heterogeneous feature-based approach in the temporal domain[1]. Each cell of the two-dimensional grid is considered as a state. A state can be empty or blocked. The agent can execute an action and transit to a neighboring empty state but it cannot visit a blocked state. For this thesis, an empty state is represented by 1, and a blocked state is represented by 0. For this thesis, there are eight possible actions that an agent

[1]A maze can be considered a temporal domain because the agent navigates the maze through a series of steps

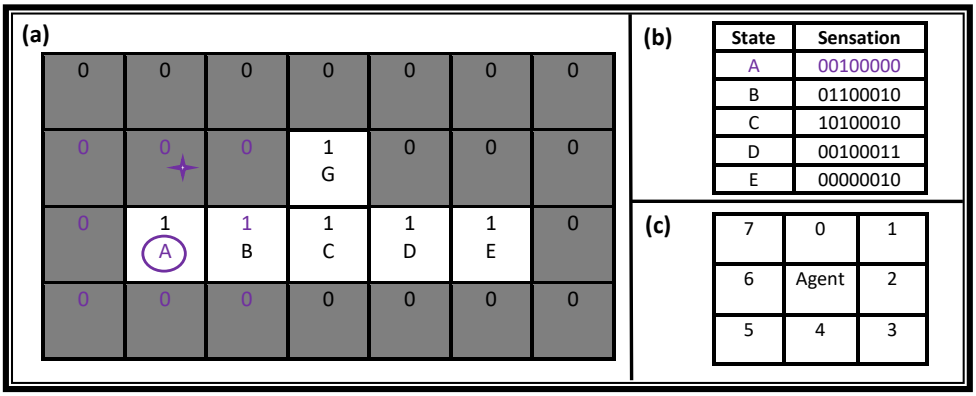


Figure 3.4: (a) A sample maze, here an empty state is represented by 1 and a blocked state is represented by 0. (b) States with sensation values, e.g. starting from the north (star), the sensation value of state 'A' is 00100000. (c) An agent with corresponding action values to visit the respective adjacent state.

can execute to visit an adjacent state. These actions are represented by incremental numbers from A_0 (starting from the top) to A_7, as shown in Fig. 3.4-c.

An environmental state (input signal) is represented by a binary string that is computed by concatenating the agent's immediate sensation of eight adjacent states. For example, in Fig 3.4 starting clockwise from the north (star), the input signal for state 'A' is "00100000". The third bit is '1' because the state on the east side of the state A is empty. A sample maze and input signal (binary string representation) of each empty state are shown in Figs. 3.4-a, 3.4-b.

A maze can behave as a stochastic Markov environment for an agent if the agent's immediate sensation provides all the required information to take the best action in all states/situations. Whereas, a maze can behave as a non-Markov environment if the agent's immediate sensation in a state does not provide the required information to behave optimally.

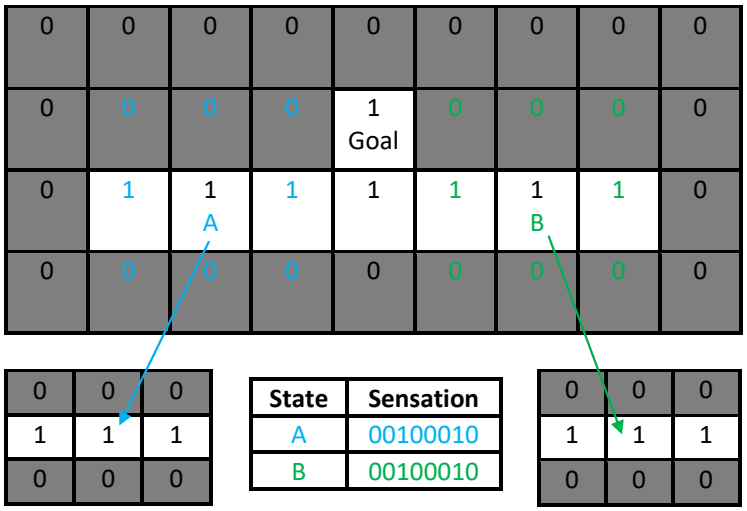


Figure 3.5: A sample non-Markov maze with two aliased states.

Such a state of a maze could be an aliased state. The existence of multiple aliased states on the navigation path generates complex hierarchical patterns [239].

A sample non-Markov maze is shown in Fig. 3.5. The states 'A' and 'B' are two aliased states. In these states, the agent's immediate sensation provides the same input signal, i.e. '00100010'. But the agent needs to take different actions to optimally reach the goal states. That is, in-state 'A', the agent needs to take action '2' to move to the right state; whereas, in-state 'B', the agent needs to take action '6' to move to the left state. Consequently, the agent is faced with a perceptual aliasing problem.

3.2 Benchmark Approaches

This section provides a brief description of the state-of-the-art benchmark approaches that have been developed to address the complex problems that are used to evaluate the AI systems developed in this thesis.

3.2.1 Existing Lateralized AI Systems

A systematic search was conducted to find studies that create lateralized artificial intelligence systems. For this purpose, three search strings are used, i.e. (i) "lateralized systems", (ii) "lateralized artificial intelligence", and (iii) "lateralized machine learning". The search was conducted on popular scientific search engines, i.e. IEEE Xplore, Science Direct, Elsevier, Scopus, and Google Scholar. The hits were less than 50 for the majority of the search engines, however, for Google Scholar, the hits were upto 2500. The studies that appear in the search results have the following major patterns: (i) magnetic resonance imaging-based studies on different parts of the brain to identify, highlight, or simulate lateralized functionality; (ii) electroencephalogram based studies to learn lateralized patterns for the identification of different diseases; (iii) studies to find a relationship between lateralized neural activity patterns and behaviors; (iv) studies to learn brain activity. It is important to note that none of the studies is conducted to create a lateralized artificial intelligence system inspired by the hemispheric lateralization. Lateralization has not been investigated in ML systems due to a number of reasons, which are discussed in Chapter 1.

3.2.2 Relevant Approaches for Boolean Problems

Different techniques have been developed to solve complex Boolean problems. The well-known and state-of-the-art approaches are described below.

Layered Learning Genetic Programming

Layered learning genetic programming (LLGP) is a methodology for learning complex problems by (a human-in-the-loop) decomposing them into a hierarchy of subtasks [274]. These subtasks are separately learned in a sequence by utilizing suitable learning algorithms. The knowledge learned for a subtask at the lower layer of the hierarchy is utilized for the learning

of a task at the upper layer [275, 276]. The layered learning approach is suitable for complex problems where direct learning is intractable and hierarchical decomposition is possible. The layered learning approach typically requires human intervention for the decomposition of the complex problem and selection of a suitable algorithm for the learning of a subtask. Moreover, the layered learning methodology requires a strict sequence of learning to solve complex tasks [277]. In contrast, the novel lateralized system will have the ability to simultaneously analyze the complex problem at both the more abstract level and the constituent level. Consequently, the novel system will identify and utilize relevant BBKs without human intervention.

Recently, a common sub-tree based transfer learning technique has been introduced in genetic programming (GP) [278]. The resultant system has the ability to automatically find relevant information that can be transferred between problems. However, it could not transfer information between different problem domains. Moreover, this system struggles to completely learn complex problems such as $7 - bit$ parity problems [278]. The novel lateralized system will have the ability to handle heterogeneous and complex Boolean problems.

Cartesian Genetic Programming

Cartesian genetic programming (CGP) is a methodology for developing graph-based programs to solve a problem. In CGP, a program is represented by a two-dimensional grid of nodes in the form of a graph. The program has the ability to implicitly reuse nodes in the graph. Consequently, the mapping for genotype to phenotype in such a program is many-to-one [279, 280]. The CGP technique generates an arbitrary sequence of computer programs that can solve only a particular problem; the learned solution cannot be reused to solve other complex problems. Moreover, the size of the solutions is very large for large-scale and complex problems. Consequently, the learning is intractable [281]. The novel

lateralized system will have the ability to generate a compact solution by utilizing the learned BBKs at different levels of abstraction.

Cooperative Coevolution

Cooperative coevolution is a methodology for solving a complex problem by dividing it into subcomponents and resolving each subcomponent independently [53]. The learning of subcomponents generates subpopulations that can only interact through cooperative evaluation, the process by which an individual from a subpopulation concatenates with the fitter individuals from other subpopulations of the same group [21]. The utilization of subjective fitness and complicated dynamics of cooperative coevolution systems are the main reasons for the failure of these systems [282]. Moreover, these systems do not allow mating between the individuals of heterogeneous subpopulations [283, 284, 285]. In contrast, the novel lateralized system will have an elementary architecture where solutions will be generated by utilizing homogeneous as well as heterogeneous BBKs.

Code Fragments based LCSs

Code fragment (CF) representation is an important encoding scheme that has been introduced in learning classifier systems (LCSs) to achieve high-level knowledge representation [15]. It has been used in LCSs to store and transfer learned knowledge. A CF is a GP-like tree in which the internal nodes have operations that are selected from a predefined set of operations, e.g. AND, OR, NOT, NAND, and XOR. The leaf nodes, in contrast, have environment variables or previous CFs. The inclusion of CFs in LCSs resolves previously unresolvable complex problems (such as 135-bit multiplexer (Mux) problem, n-bit Parity problem, and n-bit Mux problem), enables the transfer of learned knowledge, and generates compact rule sets for an optimal population [51, 236, 15].

However, CF-based systems struggle to solve complex problems such as hierarchical Boolean problems (see Chapter 5). Moreover, the solution

proposed to resolve the n-bit Mux problem requires a strict ordering of layered learning, prior knowledge of problem decomposition, and much human intervention [51]. Whereas, the cyclic graphs based technique (XCSSMA), that can solve n-bit Parity problems, is not beneficial for multiplexer problems. The finite state machines utilized in the action are unable to capture the patterns in the search space [281]. Recently, CF-fitness based techniques have been introduced to search for useful and relevant features for efficient learning. Nevertheless, these systems struggle to completely learn 18-bit hierarchical multiplexer (HMux) problems due to the homogeneous nature of their knowledge representation [286]. A novel CF-based technique will be developed to facilitate the efficient learning of complex BBKs at different levels of abstraction without human intervention.

3.2.3 Relevant Approaches for Computer Vision Problems

Existing techniques provide only a partial solution to adversarial attacks in visual classification tasks. The majority of these techniques provide deep learning based solutions. Deep learning (DL) is a methodology for extracting higher-level features and useful patterns from raw data by utilizing multiple layers in artificial neural networks [287]. A DL-based model progressively learns by transforming each lower layer input into a higher-level abstract representation. The chain of information transformation from the input layer to the final output layer is known as the credit assignment path (CAP). The depth of the CAPs in a feedforward neural network is the number of hidden layers plus one (output layer), whereas, it is unlimited in a recurrent neural network (a signal may propagate through a layer multiple times) [288]. Moreover, a DL model considers all the features homogeneously. It learns by optimally placing and selecting features at different levels that improve performance [289]. In contrast, the novel system will have the ability to simultaneously analyze the information at the constituent level and holistic level. Consequently, the novel system will generate a robust solution by utilizing the learned knowledge at dif-

ferent levels of abstraction.

Adversarial training techniques reduce over-fitting by regularizing the deep networks, which improves robustness against adversarial attacks. However, these techniques are considered non-adaptive due to their dependency on already existing adversarial data. Moreover, the need for adversarial training results in an increase in the training data size and expensive network architecture [55, 290]. Furthermore, it has been reported that adversarial-trained networks can again be fooled by creating effective adversarial examples [291].

Compression-based techniques are another approach that has been investigated as a defense against adversarial attacks. The results suggest that compression alone is inadequate to provide an effective defense [292, 293, 294]. Moreover, it is hard to find appropriate compression for a dataset. Smaller compressions are unable to handle adversarial perturbations, whereas, larger compressions decrease the classification accuracy of clean images. Modification of the deep networks is yet another area where efforts have been made to improve adversarial robustness. However, it has been reported that the majority of these methods are either unable to provide an effective defense or too complex so require very large training instances [295, 296, 297].

3.2.4 Relevant Approaches for Navigation Problems

A stochastic environment can be considered Markov if the agent's immediate perception provides all the necessary information to *decide* the best action in all situations/states. Such decision processes are called Markov decision processes (MDPs). An environment can be considered non-Markov if the agent's immediate perception does not provide all the necessary information to decide the best action in all situations/states. Such hidden states can be *aliased states*, which are only partially observable and the agent needs more information to take the best action. Such decision processes are called partially observable Markov decision processes (POMDPs)

[239]. The majority of RL agents can easily learn Markov environments, but they struggle to learn non-Markov environments. Many techniques have been developed to address perceptual aliasing in non-Markov environments. These techniques are discussed below.

Initial attempts to cope with perceptual aliasing by additionally learning an immediate perception could not solve the majority of non-Markov environments because of impractical and strict assumptions such as noiseless sensing, deterministic actions; and incomplete perception [57, 298, 299, 59]. Further attempts to utilize belief state methods to address perceptual aliasing failed due to two factors: (i) difficulty in calculating value functions, and (ii) updating belief states. These two factors make the belief state-based strategies intractable for large and complex non-Markov environments [300, 301, 302].

Another way to handle perceptual aliasing is to have a message list where states and actions are stored and passed to future decisions but this is inconvenient because of the storage size and contents. It is hard to determine the optimum memory relevant to the decisions. For example, an agent may store neutral messages and/or incorrect messages, making it hard to search for only the appropriate messages. The resultant systems could not evolve optimal rules and so were unable to solve the majority of non-Markov environments [249, 303, 304, 305].

Internal memory-based systems successfully solve simple non-Markov environments, however, they struggle to disambiguate aliased states in complex non-Markov environments. This is due to the likelihood of path entrainment, i.e. becoming stuck in a local optimum as once a route to the goal has been discovered, it could be exploited such that the exploration of potentially better routes does not occur. Moreover, multiple versions of the same route may be stored due to a lack of generalization. Consequently, internal memory-based systems are neither robust nor able to achieve optimal performance for complex problems [272]. Thus, these systems provide only ad-hoc solutions [59, 303]. Similarly, internal action-table based

systems struggle to obtain an optimal policy for complex non-Markov environments. Moreover, they require extra computations to find an appropriate policy [306, 307].

Artificial agents based on the psychological principle of *anticipatory behavioral control* provide a partial solution for non-Markov environments [273, 14]. Latent learning-based anticipatory classifier systems (ACS), and the more advanced ACS2, predict the next state of the environment, but still struggle to learn complex aliasing patterns. The methodology adopted by ACSs fails to identify the aliased states in the majority of cases. Consequently, the agent is unable to take the required best action [14, 273, 272, 308]. Even ACS2, with behavioral sequences, fails to efficiently address non-Markov environments. For example, these systems struggle to address environments with loops in states and environments that have the same aliased states on one path [273]. Recently, Orhand et al. presented a study that implemented behavior sequences in ACS2 [309]. The resultant systems, BACS2 and BACS3, performed well on the majority of mazes; however, these systems were unable to counteract all of ACS2's inherent issues to effectively resolve complex environments, such as Maze10, Littman57, and Woods102.

Zatuchna and Bagnall applied the idea of imprinting (adopted from psychology and ethology) to resolve non-Markov environments. They incorporated memory mechanisms and associative perception techniques in a single system to address aliased states. Moreover, they modified the evolutionary and reinforcement mechanisms of the standard LCSs. The resultant system, named *AgentP*, has a complex architecture that relies on tailored methods, such as memory and imprinting. AgentP observes and stores many states as it progresses, which eventually renders the learning intractable. Moreover, it inherits the above-mentioned limitations and drawbacks of ACS2 and memory-based systems. Consequently, it is still unable to completely learn complex non-Markov environments, such as Maze10 [60, 272].

Deep RL is another approach to address the perceptual aliasing problem. Deep RL agents require high computational resources to handle perceptual aliasing. This becomes worse in complex non-Markov environments, e.g., Minecraft [10], in which many states have common visual features. Interactive machine learning (IML) based techniques have been proposed to improve the performance and reduce the extraneous computations utilized by the RL agents to handle perceptual aliasing in non-Markov environments[310]. However, these techniques could not be generalized and required a human-in-the-loop to assist the agent. Moreover, it is hard to decide *when*, e.g. triggering based on low confidence, action advice/critique and *how much*, e.g. how often and level of expertise, human intervention is appropriate [10, 311, 11].

Another approach to handle the perceptual aliasing problem is the use of subgoals [312, 313, 16]. However, it is hard to select an appropriate number of subgoals and define the complexity of the subgoals. Consequently, the resultant systems have poor learning efficiency. Recently, a genetic algorithm-based strategy has been developed to find appropriate subgoals [16]. However, this technique requires extraneous computations and cannot find appropriate subgoals if the system fails to achieve the task at the initial population generation.

In summary, although methods to address perceptual aliasing exist, they either fail under challenging circumstances or entail unreasonable computational overhead. It is hypothesized that a novel lateralized system can identify aliased states at a constituent level and place them uniquely in holistic level policies.

3.3 Chapter Summary

One of the important objectives in this chapter was to provide an overview of the benchmark problems, from different domains, that will be used for the evaluation of the novel lateralized systems. The state-of-the-art bench-

mark problems are explained from Boolean, CV, and navigation domains (mainly used in Chapters 5, 6, and 7, respectively). These benchmarks include a wide range of problems, i.e. single and multiple step problems, supervised and reinforcement learning problems, Boolean and real-valued features problems, and problems that entail Markov or partially observable Markov decision processes. Consequently, these benchmark problems provide an ideal test-bed not only to obtain the proof-of-concept but to evaluate the robustness and effectiveness of the lateralized approach.

Another important objective was to highlight the strengths and limitations of the state-of-the-art relevant benchmark approaches. EC techniques have become proficient at linking environmental features to describe simple patterns in data. These techniques can easily handle simple Boolean problems but they struggle to handle complex hierarchical Boolean problems. State-of-the-art EC techniques can only handle those complex Boolean problems where direct learning is intractable and hierarchical decomposition is possible. The majority of the EC techniques require human-in-the-loop intervention and a strict sequence of learning. They do not have the ability to transfer/reuse learned knowledge between different problem domains. The lateralized framework will be adapted to create a lateralized AI system that can automatically identify and utilize/reutilize learned BBKs at different levels of abstraction to solve complex Boolean problems (see Chapter 5).

DL-based methods can efficiently and precisely handle visual classification tasks where the relationship between features and target is linearly separable. An adversarial attack can easily fool the majority of the DL models. Existing state-of-the-art DL-based techniques provide only a partial solution to handle adversarial attacks. The solutions offered by these techniques are non-adaptive, computationally expensive, too complex, and inadequate to provide an effective defense. The lateralized framework will be adapted to create a lateralized AI system that can provide robust solutions for visual classification tasks (see Chapter 6).

Reinforcement learning agents can easily handle Markov environments but they struggle to address perceptual aliasing in non-Markov environments. The solutions provided by the majority of the techniques are intractable, complex, computationally expensive, require human-in-the-loop, and rely on tailored methods. The lateralized framework will be adapted to create a lateralized AI system that can utilize FoRs based Learning to overcome perceptual aliasing in non-Markov environments (see Chapter 7).

The majority of the relevant benchmark techniques develop a huge network of homogeneous knowledge to solve complex problems. These techniques can effectively handle problems that consist of homogeneous patterns of features. However, they struggle to address problems that consist of complex heterogeneous patterns of features, i.e. hierarchical patterns within patterns. Heterogeneous feature-based lateralized AI systems have not yet been investigated. One factor that may have hampered progress in creating lateralized AI systems is the lack of availability of an underlying lateralized framework. In this thesis, a novel lateralized framework will be created that will be adapted to develop lateralized AI systems for Boolean, CV, and navigation problem domains (see Chapters 4, 5, 6, and 7, respectively).

4

Framework of a Lateralized Artificial Intelligence System

Lateralization is ubiquitous in vertebrate brains and is considered an important factor in biological intelligence. The broad distribution of lateralization phenotypes in vertebrates indicates that a lateralized framework could enhance neural efficiency and effectiveness in performing tasks. The cognitive architecture of vertebrate brains provides an appropriate computational abstraction that can be used to create a lateralized framework.

The purpose of a framework is to provide guidance for creating a specific type of system for a wide range of problem domains. Eventually, it may capture community consensus and become a cumulative reference point. Moreover, it can be revised and extended to accommodate new findings. Such a framework is to be created for lateralized artificial intelligence systems.

This chapter aims to devise a novel lateralized framework inspired by the principles of biological intelligence, derived from the current understanding of learning mechanisms implemented in the brains of human and non-human animals. The goal here is not to reproduce any specific lateralized vertebrate brain network; instead, to be inspired by the principles of lateralization, drawing on key mechanisms of knowledge perception, representation, and utilization, and patterns of connectivity within and between cognitive modules. The novel lateralized framework spans key aspects of knowledge perception, knowledge representation and utilization, and connectivity patterns. It determines the essential functionality, critical methods, and associated parameters that are required to be incorporated into an AI system to behave as a lateralized AI system. It is expected that the novel lateralized framework can be adapted to create lateralized artificial intelligence systems for a wide range of problem domains.

4.1 Introduction

The majority of AI systems can effectively handle problems that consist of homogeneous patterns of features. However, they struggle to address problems that consist of complex heterogeneous patterns of features, i.e. hierarchical patterns within patterns. For example, learning classifier systems (LCSs) can easily learn 20-bit multiplexer problems but they struggle to learn 18-bit hierarchical multiplexer problems [51]; artificial agents can easily solve deterministic environments but they struggle to handle non-Markov environments [272]; deep networks accurately classify visual stimuli but struggle to address adversarial images [48, 53]. Conventional AI systems, such as LCSs and deep networks, may learn complex patterns but they need a huge/deep network of homogeneous knowledge, human-in-the-loop decomposition, and/or complex architectures [9]. Moreover,

homogeneous systems are able to learn when the relationship between features and target is linearly separable. However, this hallmark can be exploited to fool these systems [52, 55, 56].

A plausible solution is to develop lateralized systems that can utilize heterogeneous features. Heterogeneous features can be parsimonious as they only encode necessary information to solve a problem (or part of a problem) and can provide multiple representations of the same object (or environment), i.e. at a constituent level and holistic level. This enables hierarchies that can provide knowledge representation at different levels of abstraction such that each knowledge component, a building block of knowledge (BBK)[1], has sufficient information to solve a specific part of a problem or the whole problem. An input signal can be considered at the constituent level and holistic level simultaneously. This enables the AI systems to address both individual features and global patterns in parallel.

Biological intelligence supports this type of heterogeneity. A brain is not an amorphous system. It is made up of different modules that communicate through electrochemical signals. Each module solves a specific problem or part of a problem. The knowledge learned from small-scale and simple problems can be re-utilized by the brain to solve large-scale and complex problems in similar and related domains [21, 22, 23].

Two organizing principles of vertebrate (and many invertebrate) brains — lateralization and modularity of function — support reuse of learned knowledge at different levels of abstractions [24, 26, 27, 28]. Lateralization enables the biological brain to process both individual features and global patterns in parallel. For example, in many domains, the left hemisphere processes elementary (constituent) information while the right hemisphere works at a higher (holistic) level of abstraction. It has been reported that this asymmetry enhances cognition as well as neural efficiency [21, 23, 31].

In this thesis, I hypothesize that the incorporation of these principles

[1] A BBK is a unit of knowledge that is transferable and can be used or reused to solve a part of a problem or the whole problem.

into AI systems could overcome the limitations inherent in conventional homogeneous systems, allowing for an efficient and robust solution to complex hierarchical problems. Lateralized AI systems need to have knowledge representation at the constituent level and holistic level. The basic elements of knowledge, i.e. individual features and simple niches[2], are represented by one half of the system (which in this thesis is termed the *left* half)); whereas the higher-order abstract representations that are extracted across niches are handled by the other (*right*) half[3]. Consequently, the constituent BBKs provide the local viewpoint, while the holistic BBKs provide the big picture (world viewpoint) of the overarching patterns in the data. This splitting of knowledge can be repeated at different levels of abstraction depending on the nature and complexity of the problem. Finally, the left half and right half coordinate to effectively solve the problem. The majority of real-life problems are composed of sub-problems. Hence, the ability to simultaneously consider the problem at different levels of abstraction (constituent level and holistic level) is critical for efficient and robust learning. A schematic illustration of conventional homogeneous and novel lateralized approaches is shown in Figure 4.1.

Lateralization has not been investigated in artificial intelligence systems (see Section 1.2). One factor that may have hampered progress in creating lateralized AI systems is the lack of availability of an underlying lateralized framework. The lateralized framework needs to include critical components of a lateralized AI system so that it can be modified to suit varied tasks. This framework will provide a coherent foundation that allows researchers to create lateralized systems across domains. Instead of creating a lateralized system for each problem domain from scratch, this framework can be adapted, customized, and extended as per the requirements of the problem domain. The notion of a lateralized framework has

[2]The area of a sample space where neighboring instances share a common property is called a niche.

[3]The terms *left* and *right* are used here metaphorically, and do not intend to model the specific left and right hemispheres of any human or animal brain.

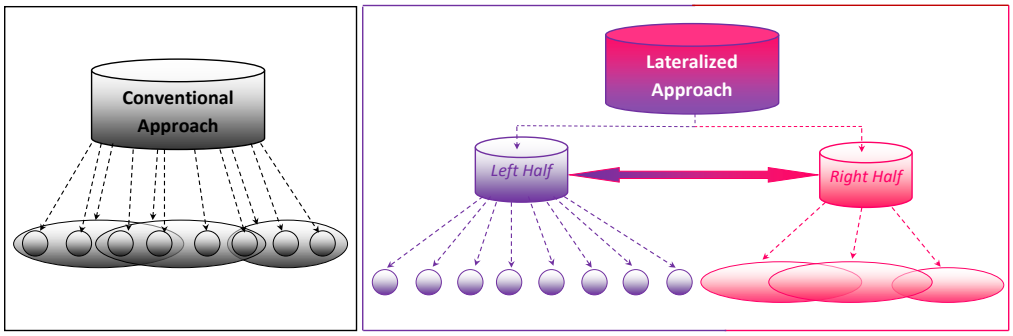


Figure 4.1: A schematic illustration of a conventional homogeneous approach and a lateralized approach. A conventional AI approach (left side) considers individual features and niches in a homogeneous manner. Whereas, a lateralized AI approach (right side) splits a complex problem in constituents and abstract knowledge. (color key: constituent, holistic, and mix knowledge proceedings are represented by purple-white, pink-white, and light purple-pink gradients, respectively)

its roots in biological and artificial intelligence, where efforts have been made to create a standard models of the mind, i.e. a common computational framework across AI, cognitive neuroscience, and robotics [171].

4.1.1 Chapter Objectives

The main objective reported in this chapter is to devise a lateralized framework, inspired by the principles of biological intelligence, that can be adapted to develop a lateralized AI system for a wide range of problem domains. To achieve this objective, this chapter aims at addressing the following sub-objectives:

(i) Establish the essential principles of lateralization from cognitive neuroscience, especially from the cognitive architecture in vertebrate brains. Identify and explore in detail the aspects of lateralization that are relevant to this thesis. For this purpose, lateralization will be ex-

plored from the following perspectives.

- (•) The representation and processing of sensory information, received from an environment, by the hemispheres.
- (•) Coordination between hemispheres to share the relevant information which is required to solve a problem. It includes the integration of knowledge among different regions of the brain.
- (•) Goal-driven processing through inhibition and excitation signals for performance efficiency. This perspective highlights the needs for mechanisms to select the the most suitable and relevant hemisphere for a specific task to optimise the required computations.

(ii) Devise a general framework for lateralized AI systems, derived from the current understanding of learning mechanisms observable in vertebrate brains. The sub-objectives required to achieve this objective are:

- (•) Identify and explain the important features of a lateralized AI system.
- (•) Determine the essential functionality, critical methods, and associated parameters that are required to be incorporated into an AI system to behave as a lateralized AI system.

(iii) Highlight the problem domains which could benefit from the lateralized approach.

4.1.2 Chapter Organisation

The remainder of this chapter is organized as follows. Section 4.2 provides the relevant literature from cognitive neuroscience that inspires the lateralized framework. The lateralized framework for AI systems is presented

in Section 4.3. It explains the important features and critical components of a lateralized AI system. Section 4.4 describes aspects of highlighted problem domains which could benefit from the lateralized approach. Section 4.5 provides a further analysis of the proposed lateralized framework. Finally, the chapter summary is presented in Section 4.6.

4.2 Lateralization in Vertebrate Brains

Vertebrate brains have a functional architecture that allows them to abstract knowledge from simple and small-scale problems and then reuses it to solve complex problems. It is not the intention of this work to model the specific architecture of a specific species; rather it takes inspiration from basic principles of functional organization that are fundamental to vertebrate intelligence. This thesis focuses on lateralization, which is one such principle.

The propensity of a specific cognitive process to be performed more efficiently and precisely by one hemisphere as compared to the other is called hemispheric lateralization [29]. At the macro-structural view, the left and right hemispheres look alike. However, they have distinct neuroanatomy, neurochemistry, and functional architecture [29, 30]. Three aspects of lateralization, relevant for application to AI, are presented below.

4.2.1 Representation and Processing

Some functions are strictly lateralized to one hemisphere or the other. For example, each hemisphere receives sensory inputs from the opposite side of the body and controls the contralateral musculature. But, for higher-order cognition, differences between hemispheres are more relative than absolute, with both hemispheres contributing to most tasks. Often these hemispheric differences concern the scale at which the same sensory inputs are represented for subsequent processing. For example, in visual

perception, it is common that the left hemisphere processes information at a local (or constituent) level while the right hemisphere processes information at a more global (holistic) level [32, 33, 34]. Similarly, in speech perception, commonly the left hemisphere processes segmental information (individual phonemes that makeup words) while the right hemisphere processes super-segmental information (global intonational patterns that reflect emotion or intention of the speaker) [35, 36, 37].

These fundamental differences in representational scale may arise through filtering. For example, the Double Filtering by Frequency model proposes that the left hemisphere acts as a high pass filter, allowing it to represent detailed information that is available in high spatial or temporal frequencies. At the same time, the right hemisphere acts as a low pass filter, allowing it to represent global patterns that emerge in low spatial or temporal frequencies [84, 85]. Such complementary forms of representation are not limited to sensory information, however. For example in language processing, the left hemisphere may activate single, literal, meanings of words or sentences, while the right hemisphere keeps alternative, metaphorical, or figurative meanings active [73, 74]. This ability to represent and process the same problem instance at a local constituent level and a global holistic level will be incorporated in the lateralized framework presented here.

4.2.2 Coordination

Effective cognition requires that the computations carried out in opposite hemispheres be coordinated. Recognizing faces requires that we integrate individual features (left) with their configural arrangement (right) [40]; understanding a joke requires that we integrate the literal meanings of individual words (left) with their alternative subtext (right) [41]; understanding a song requires that we integrate the lyrics (left) with the melody (right) [42]. It is the coordination between the left and right hemispheres that enables the transfer of critical information at different levels of ab-

straction. This coordination will be included in the modules of the lateralized framework.

4.2.3 Goal-driven Processing

Vertebrate brains have the ability to select the computations required to perform a specific task from the most suitable and relevant hemisphere. Goal-driven processes analyze the problem at hand and shift control to the superior (suitable) hemisphere. For example, if the emotional state of a conversational partner is most relevant, outputs from right hemisphere speech processing systems will dominate; however, if the linguistic elements are of concern, then left hemisphere computations are prioritized [31, 43]. The connections between hemispheres in vertebrate brains can be excitatory or inhibitory, allowing for either integration or inhibition, as goals dictate [44]. The ability to identify which hemisphere is best matched to the task is important in practical situations. A strategy will be developed for the identification of the most suitable module with respect to the given problem instance.

4.3 Lateralized Framework

The lateralized framework provides an architecture that could be used to develop a lateralized AI system for a wide range of problem domains. The goals of this section are two-fold: first, to highlight the important features of a lateralized AI system and to show that how the neural principles identified in section 4.2 can be implemented in an artificial intelligence system; and second, to present a general architecture that could be used to create a lateralized AI system.

4.3.1 Features of a Lateralized AI System

A lateralized AI system mimics the type of heterogeneity [4] that can be seen in biological intelligence. It is a modular system such that each module can solve a part of a problem or the whole problem. Moreover, it may have the ability to learn knowledge from simple and small-scale problems and then (re)utilize it to learn complex and large-scale problems in related and similar domains. The important features of lateralized AI systems are presented below.

Representation and Processing

An important feature of a lateralized AI system is the ability to simultaneously process the same environmental signal at different levels of abstraction, i.e. at the constituent level and the holistic level. Instead of considering the environmental signal homogeneously, the lateralized system presents the environmental signal in two halves such that one half of the system (which this thesis calls the *left* half) considers its constituents (elementary features), whereas, the other (*right*) half considers it at a higher level of abstraction (high-level features). Consequently, the *left* half represents the most basic elements of knowledge; i.e. individual features and simple niches. At the same time, the other (*right*) half creates a more abstract knowledge representation; i.e. higher-order features extracted across niches. This feature empowers the lateralized system to address the details of the problem and the higher level (big picture) at the same time.

Coordination

The coordination between modules is an important attribute of a lateralized AI system. A lateralized system enables knowledge integration by allowing the transfer of critical information. The computations carried out

[4] Lateralization can be considered as a special type of heterogeneity.

by different modules (such as hemispheres in biological intelligence) need to be integrated. Different knowledge components can be utilized or re-utilized at different levels of abstraction, i.e. a holistic knowledge component at one level can be re-utilized as a constituent knowledge component at a higher level of abstraction. Thus, this feature allows different system components to reuse the learned knowledge at different levels of abstraction.

Goal-driven Processing

The processing of the same input signal at different levels of abstraction essentially increases the workload. This goal-driven processing feature plays a critical role in reducing the extra computations. It enables the lateralized AI system to identify and utilize those modules which could effectively solve the given problem. Moreover, it empowers the lateralized system to activate or deactivate the most appropriate system module through inhibit or excite signals, as the goal dictates.

Knowledge Identification and Utilization

The identification and utilization of the relevant (constituents and holistic) BBKs is an important feature of a lateralized AI system. Although biological brains do not have a central storage unit, lateralized AI systems may have a heterogeneous knowledge pool. The knowledge pool stores all the learned BBKs. A lateralized system applies strategies to automatically identify and utilize relevant BBKs with respect to the given problem. It includes the utilization of BBKs at different levels of abstraction, e.g. a holistic level BBK for one problem may be utilized as a constituent level BBK for another higher-level problem.

4.3.2 Lateralized Architecture

The architecture of the lateralized framework provides an overview of how information is presented, processed, updated, and stored in a lateralized AI system. Moreover, it explains the flow of information and communication between the system components. The core components of the lateralized framework are perception, left-half, right-half, resolution, and heterogeneous knowledge pool. Each of these components can be developed as a single unit or further decomposed into sub-modules, depending on the nature and complexity of the problem domain. A schematic illustration of a lateralized framework is presented in Fig. 4.2.

The knowledge associated with a learned problem needs to be stored, for future use, in the knowledge pool. This knowledge can be stored as a unit of knowledge (representing the constituent level) or a block of knowledge (representing the holistic level). The constituent knowledge is stored as elementary features, e.g. code fragments; whereas, holistic knowledge is stored patterns of features, e.g. population of rules.

At the start of the learning process, the knowledge pool is empty and system behaves as an ordinary AI system. Once the system has learned a problem and achieved a given threshold performance accuracy (e.g. a 100%), all the learned BBKs (constituents and holistic) are stored in the heterogeneous knowledge pool. Let $\mathcal{K}$ be a knowledge pool, i.e. a set that holds n blocks of learned knowledge, as shown in equation 4.1.

$$\mathcal{K} = \{L_0, L_1, L_2, L_3, \ldots, L_{n-1}\} \tag{4.1}$$

The perception component directly communicates with the environment to receive the input signal. Subsequently, this input signal is simultaneously shared with the left-half and the right-half components. Both the components start processing in parallel. The left-half and right-half components are the core of the lateralized framework. These components play critical roles inspired by the brain hemispheres.

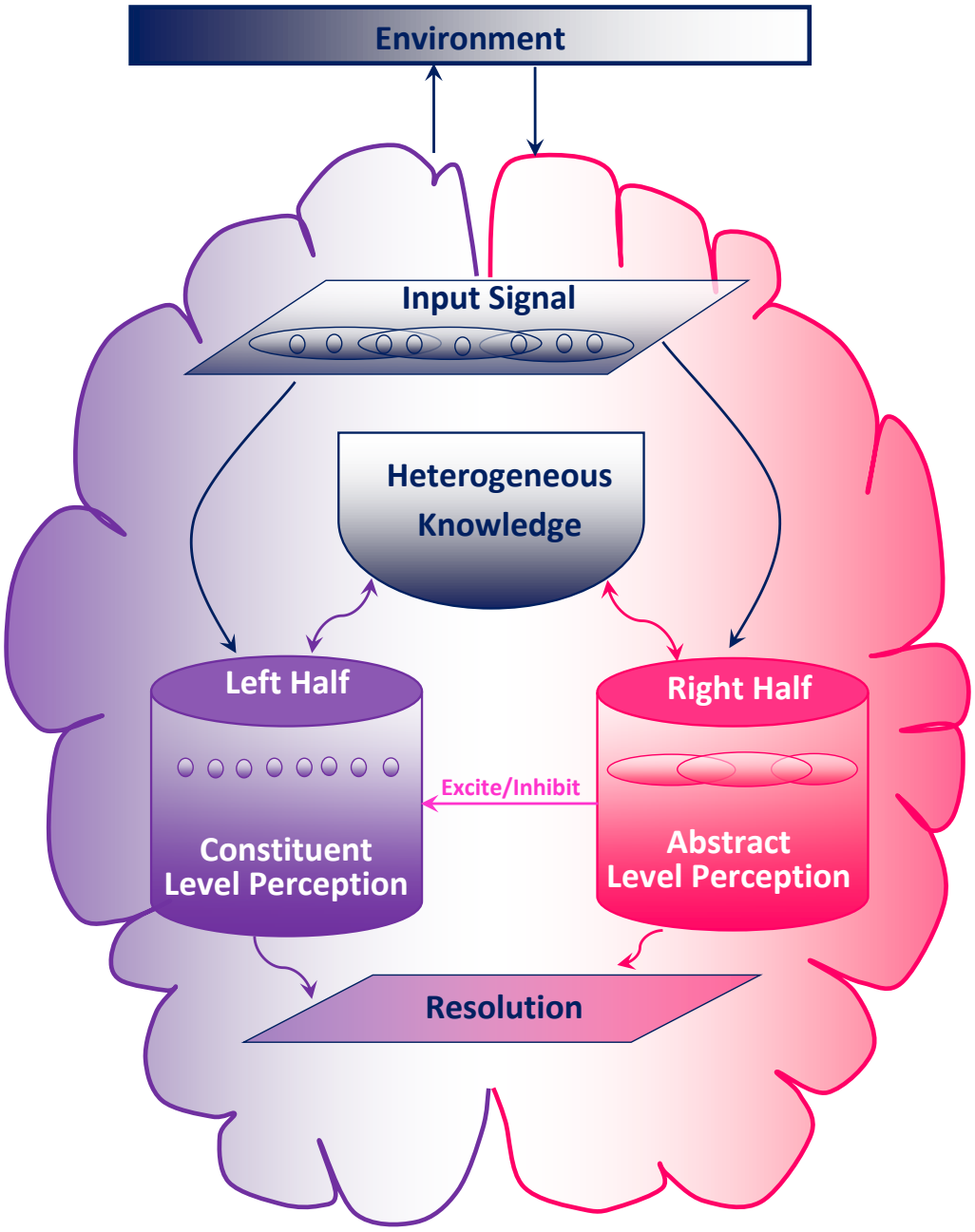


Figure 4.2: A schematic illustration of a lateralized framework. (color key: constituent, holistic, and mix knowledge proceedings are represented by purple-white, pink-white, and light purple-pink gradients, respectively)

The left-half component considers the constituents (elementary features) of the input signal. For this purpose, the left-half identifies those learned BBKs, from the knowledge pool, that could form the constituents of the input signal. These knowledge components could be repeated or combined to resolve the given problem. Initially, the left-half identifies the set of BBKs that can be considered to solve the given problem. $\mathcal{K}_a$ ($\mathcal{K}_a \subseteq \mathcal{K}$) is a set of BBKs that are applicable to solve the given problem, as shown below.

$$\mathcal{K}_a = \{L_i : \mathcal{A}pt(L_i) \mid L_i \in \mathcal{K}\} \tag{4.2}$$

where "$\mathcal{A}pt$" is a function that determines whether a BBK is applicable to the given problem or not. For example, in Boolean problems, the learned BBK of $3-$bit Parity and $4-$bit Parity problems are applicable and not applicable to solve $18-$bit hierarchical Mux problem, respectively; in computer vision (CV) problems, the learned BBKs of 'eye' and 'feather' are applicable and not applicable to identify a 'cat' image, respectively.

The left-half combines learned BBKs, from the set of applicable knowledge $\mathcal{K}_a$, and creates groups of knowledge blocks of size m that can solve the given problem. These knowledge groups can be defined as:

$$\mathcal{G}_2 = \{(L_i, L_j) : \mathcal{S}olve(L_i, L_j) \mid L_i, L_j \in \mathcal{K}_a\}$$

$$\mathcal{G}_3 = \{(L_i, L_j, L_k) : \mathcal{S}olve(L_i, L_j, L_k) \mid L_i, L_j, L_k \in \mathcal{K}_a\} \tag{4.3}$$

$$\ldots$$

$$\mathcal{G}_m = \{(L_i, L_j, \ldots L_p) : \mathcal{S}olve(L_i, L_j, \ldots L_j) \mid L_i, L_j, \ldots L_p \in \mathcal{K}_a\}$$

where $\mathcal{S}olve$ is a function that utilizes only the input BBKs to solve the given problem.

Finally, the left-half creates a set of relevant knowledge $\mathcal{R}_l$ by taking the members of all the knowledge groups that can solve the given problem with a performance accuracy above the given threshold θ, as shown below:

$$\mathcal{R}_l = \bigcup_{t=1}^{m} \mathcal{G}_t \mid Eval(\mathcal{G}_t) > \theta \tag{4.4}$$

where $Eval$ is a method to evaluate the performance accuracy of a specific knowledge group to solve the given problem. For example, $\mathcal{G}_2 = \{L_0, L_2\}$, $\mathcal{G}_3 = \{L_1, L_2, L_5\}$, and $\mathcal{G}_4 = \{L_0, L_3, L_4, L_6\}$. The performance accuracy of $\mathcal{G}_2$ and $\mathcal{G}_4$ is greater than the given threshold θ, whereas the performance accuracy of $\mathcal{G}_2$ is less than θ. The resultant set of relevant knowledge is $\mathcal{R}_l = \{L_0, L_2, L_3, L_4, L_6\}$. It is noted that the technique required to identify and evaluate the constituent level BBKs needs to be developed according to the nature of the problem. Subsequently, the left-half shares the identified relevant knowledge ($\mathcal{R}_l$) with the resolution method. The pseudo-code of the strategy adopted by the left-half is given in Algorithm 1.

The right-half addresses the higher-level abstract features of the same input signal. For this purpose, the right-half identifies those learned BBKs from the applicable knowledge set (eq 4.2) that could form the higher-level abstract patterns. Subsequently, the right-half computes the performance accuracy of each holistic level BBK to independently solve the problem. Let $\mathcal{R}_h$ be a set of relevant learned knowledge blocks, with their respective performance accuracy, that can independently solve the given problem with an accuracy above the given threshold θ, as shown in equation 4.5.

$$\mathcal{R}_h = \left\{ \, (L, Eval(L)) : Eval(L) > \theta \, \forall \, L \in \mathcal{K}_a \, \right\} \tag{4.5}$$

It is noted that the technique to identify and evaluate the holistic level BBKs needs to be developed according to the nature of the problem. During the evaluation process, if a holistic level BBK is found with a 100% (or above the given threshold) performance accuracy, the right-half generates an inhibit signal to the left-half, which immediately stops processing.

Algorithm 1: Strategy adopted by the left-half to identify the relevant blocks of learned knowledge.

```
Data: The input signal, Threshold
Result: Relevant blocks of learned knowledge
Receive an input signal from the environment;
Create Applicable Knowledge();      % Identifies the set of blocks of
  knowledge that are applicable to solve the given problem (cf. 4.2).
Generate Knowledge_Groups();        % Combines the block of
  applicable knowledge and generate knowledge groups that can solve the
  given problem (cf. 4.3).
for Each Knowledge_Group do
    Evaluate();      % Compute the performance accuracy of each
      knowledge_group to solve the given problem.
    if Accuracy > Threshold then
            % The performance accuracy is above the given threshold.
        Add Relevant_Knowledge_Set();      % Add building blocks
          of knowledge to the set of relevant knowledge (cf. 4.4).
    end
end
Return Relevant_Knowledge_Set();      % Share the identified
  relevant knowledge with the resolution method.
```

Otherwise, it generates an excite signal to the left-half, which allows it to continue processing. Finally, the right-half shares the identified relevant knowledge ($\mathcal{R}_h$) with the resolution component. The pseudo-code of the strategy adopted by the right-half is given in Algorithm 2.

The resolution method analyses the feedback from the left-half and right-half components. If it receives an inhibit signal from the right-half, the system recognizes that no new learning is required, and there is no further processing on this problem. However, if it receives an excite signal from the right-half, it analyses the feedback from the left-half. If relevant

Algorithm 2: Strategy adopted by the right-half to identify the holistic level learned knowledge that can independently solve the given problem.

Data: The input signal, Accuracy Threshold
Result: Relevant blocks of learned knowledge that can independently solve the given problem

```
Receive an input signal from the environment;
Identify Relevant_Knowledge();  % Identify holistic level learned blocks of knowledge that can independently solve the problem.
for Each Knowledge_Block do
    Evaluate();  % Compute the performance accuracy of each knowledge_block to independently solve the given problem.
    if Accuracy > Threshold then
        % The performance accuracy is above the threshold.
        Save Accuracy Value();  % Save the performance accuracy for the candidate knowledge block.
        Add Relevant_Knowledge_and_Accuracy();  % Add building blocks of knowledge and associated performance accuracy to the set of relevant knowledge (cf. 4.5).
    end
end
for Each Relevant_Knowledge_Block do
    if Accuracy > Threshold then
        % The performance accuracy is above the threshold, say 95%.
        Set Excite False();  % Set the flag generate excite signal as false. Generate Inhibit Signal();  % Generate inhibit signal to the left-half so that it can stop further processing.
    end
end
if Generate Excite is True then
    Generate Excite Signal();  % Generate excite signal to the left-half so that it continuous its processing.
end
Return Relevant_Knowledge();  % Share the identified relevant knowledge with the resolution method.
```

constituents or holistic BBKs exist, the system starts learning by utilizing those BBKs. However, if no such candidate BBKs exist, the system considers it a completely new problem, behaves as an ordinary AI system, and starts learning from a tabula rasa. The pseudo-code of the overall strategy suggested by the lateralized framework is given in Algorithm 3.

4.4 Problem Domains

The novel lateralized approach may be beneficial for a wide range of problem domains, e.g. single or multiple-step problems, supervised or reinforcement learning, Boolean or real-valued features, and Markov or partially observable Markov decision processes. The lateralized approach needs to be tested against a wide range of problems from different domains. The proof-of-concept of the novel lateralized approach will be obtained by using complex Boolean problems, whereas, the robustness and effectiveness of the lateralized approach will be evaluated by using computer vision and navigation problems, respectively.

The lateralized approach may not work any better than standard approaches for optimization problems or other such problems where the constituent knowledge is either not available or not re-utilized to solve higher-level problems. It is required to first learn constituent knowledge to take advantage of the lateralized approach. It results in extra cost and computations, which may slow down the learning rate at the start of training. However, once the constituent knowledge is learned, the lateralized system can utilize/re-utilize it to efficiently solve complex problems in similar or related domains. For example, when dealing with images the system needs to recognize salient objects at the level of constituent parts (e.g. eyes in a face or headlights in a car) to efficiently identify whether the class is a face or a car. However, once a class is learned, the lateralized system could re-utilize it, e.g. for the identification of crowd vs motorway congestion. It is considered that the lateralized systems may open a new

Algorithm 3: Overall strategy adopted by the lateralized framework to solve the given problem (cf. Fig. 4.2).

Data: The environment and system configurations
Result: Solve the given problem with no new learning, learning from existing knowledge, or completely new learning

```
Receive an input signal from the environment;
Process Left-Half and Right-Half ();   % Simultaneously process the input signal at left-half and right-half modules.
    Left-Half();   % The left-half considers the input signal at constituents level.
    Right-Half();   % The right-half addresses the input signal at holistic level.
Resolution
    Analyze RHSM_Output();   %Iteratively search through the list of holistic blocks of knowledge and check their performance accuracy values.
    if Performance Accuracy > Threshold then
            %Performance accuracy is 100 (or above the given threshold).
        Generate Inhibit Signal();   %Generates an inhibit signal so that LHSM can cease its working and further analysis on the feedback from LHSM can be stopped.
        Set_Flag "No learning is Needed";
    else
        Generate Excite Signal();
    end
    Analyze LHSM Output();
    if Relevant Knowledge Groups Exist then
            %Iteratively search through all the relevant knowledge groups.
        Mark Relevant_Knowledge();   %Marks (set Flag) all the learned blocks and their associated knowledge as a relevant knowledge.
        Set_Flag "Learning From Existing Knowledge";
    else
        Set_Flag "Learning From Scratch";
    end
    Store Learned Knowledge();   % Store all the learned knowledge in the knowledge pool.
```

door for developing systems that can efficiently learn real-world classification problems.

4.5 Discussion

It is not the intention of this chapter to create a single lateralized framework by which all the community would abide; rather to provide a solid functional architecture of a lateralized AI system that could be further refined as more is learned. Lateralization has not been thoroughly investigated to create AI systems. What is sought though is to create a general lateralized framework that highlights the important features and critical components of a lateralized AI system. This framework provides an opportunity for the community to create lateralized AI systems, at least, with similar abstract level architectures.

It is anticipated that the lateralized framework will be further evolved as the community investigates how to create suitable lateralized systems for different problem domains. During this process, researchers may tune parts of this framework, suggest improvements, or implement additional components.

4.6 Chapter Summary

The main goal of this chapter was to devise a lateralized framework that could be adapted to develop a lateralized AI system for a wide range of problem domains. This goal was achieved by creating a general lateralized framework, based on the current understanding of learning principles in vertebrate brains, for AI systems. Important features and critical components of a lateralized AI system were presented. It is anticipated that this framework will provide a solid basis, for the community, to create lateralized AI systems for a wide range of problem domains.

In order to provide a proof-of-concept, this framework will be adapted

to create a lateralized AI system for the Boolean problem, see Chapter 5. The robustness of the lateralized approach will be evaluated by adapting this framework to create two lateralized AI systems for computer vision problems, see Chapter 6. Finally, the additional benefits of the lateralized approach will be shown by adapting this framework for multi-step navigation problems, see Chapter 7.

5

Lateralized Learning to Solve Complex Boolean Problems

When classifying instances of environmental features, evolutionary machine learning has become proficient at linking features together but can miss higher-order patterns, failing to detect patterns made-up of patterns of features. This abstract reasoning is needed as constituent patterns of features often form higher-level general patterns in real-world tasks, e.g. in understanding speech or recognizing object ontologies. Biological nervous systems have the ability to abstract knowledge from simple and small-scale problems in order to then apply it to resolve more complex problems in similar and related domains. It is thought that lateral asymmetry of biological brains allows modular learning at different levels of abstraction, aiding transfer between tasks.

A lateralized framework for artificial intelligence systems has been created. This chapter aims to develop a novel evolutionary machine learning system, by adapting the lateralized framework, to obtain a proof-of-concept of the lateralized approach. Considering the same problem at different levels of abstraction enables the novel system to reframe a complex problem as a simple problem and efficiently resolve it. The results of analyzable Boolean tasks show that the lateralized system has the ability to encapsulate underlying knowledge patterns in the form of building blocks of knowledge. Problems that contain a natural hierarchy of patterns are resolved more effectively than in previous work (i.e. 18-bit hierarchical multiplexer problem). Moreover, reusing learned general patterns as constituents for future problems advances transfer learning (e.g. n-bit parity problem effectively becomes a sequence of 2-bit parity problems).

5.1 Introduction

A lateralized framework for artificial intelligence systems has been created (see Chapter 4). The next step is to develop a novel lateralized AI system, by adapting the lateralized framework, to obtain a proof-of-concept of the lateralized approach. This could be achieved by testing the novel lateralized system on interogatable, single-step, scalable, and complex problem domains.

Modern classifier systems can effectively classify targets that consist of simple, homogeneous features. However, they struggle to deal with many real-world problems that entail hierarchical patterns within patterns. Although it is possible to capture such complex structures in homogeneous systems, they require large/deep networks of knowledge and do not take advantage of the potential to transfer knowledge between levels in the hierarchy [9]. Early attempts to address this deficiency have been made in

Capsule networks that can be used to better model hierarchical relationships [18]. However, this chapter considers how evolutionary machine learning (EML) can also better model hierarchical relationships.

It is considered that a simple pattern can be represented as a disjunctive/conjunctive normal form where the clauses consist of a limited number of literals that are the environmental features. A complex pattern can be formed when, instead of the literals, the output of these clauses (i.e. subclauses) based on the literals are used, which can form a hierarchy of knowledge. These subclauses can be used/reused throughout the overall pattern as meta-features, where leveraging this phenomenon does not occur in existing learning systems.

Heterogeneous features (environmental features and the constructed meta-features) could represent knowledge at different levels of abstraction in compact building blocks of knowledge (BBKs). It is postulated that the BBKs are relevant and sufficient to solve a specific problem [17]. Heterogeneous features can be parsimonious as they only encode necessary information and represent knowledge at different levels of abstraction, i.e. at a constituent level and holistic level. It is hypothesised that a heterogeneous features based lateralized EML system could efficiently learn complex problems by splitting knowledge representation into constituent and abstract components.

Component-based EML techniques, including Genetic programming (GP), provide only a partial solution to this problem. For example, a layered learning GP (LLGP) technique has been used to cope with the hierarchical distribution of knowledge. This technique requires a human-in-the-loop hierarchical decomposition of the task and selection of suitable algorithms for learning sub-tasks. Moreover, LLGP requires a strict sequence of learning to resolve large-scale problems [274, 276]. Similarly, a transfer learning-based GP technique can easily solve simple classification tasks but struggles to completely learn complex problems, e.g. $7 - bit$ parity problems [278]. Moreover, attempts have been made to incorporate

extended compact genetic algorithms and the Bayesian optimization algorithm in the learning classifier system to identify and utilize the relevant building blocks of knowledge [314]. Cartesian GP (CGP) is another GP-based technique that implicitly reuses graph nodes to address large-scale problems [279, 315]. Although CGP has the ability to produce a many-to-one genotype to phenotype mapping, it generates an arbitrary sequence of computer programs that can solve only a particular problem. It also creates very large solutions, making the learning intractable [281].

Recently, the representation of complex knowledge in hierarchical problems has been achieved by developing code fragment-based learning classifier systems (LCSs) [51, 15]. However, these systems require a huge number of training instances, strict ordering of layered learning, and much human intervention. Moreover, these systems process everything at the same level of abstraction during the learning process, i.e. the population of rules is still monolithic [236, 15, 241]. Consequently, these systems do not have the ability to identify and learn higher-order abstract relationship(s) between the features and knowledge of a complex problem.

Even systems that are able to use learned subcomponents fail to achieve the final step of efficiently integrating knowledge that is represented at different levels. For example, cooperative coevolution methods link subpopulations of learned subcomponents but do not allow mating between individuals in heterogeneous subpopulations [283]. Consequently, individual parts of the system cannot efficiently solve the whole problem because they cannot mate across levels of abstraction.

5.1.1 Chapter Objectives

The main objective reported in this chapter is to create a novel lateralized EML system by adapting the developed lateralized framework. The system will have the ability to apply lateralization and modular learning at different levels of abstraction to solve complex problems. To achieve this objective, the following sub-objectives are set:

(i) Create a lateralized system such that a single input can be processed at different levels of abstraction, i.e. at the constituent level and/or the holistic level. Instead of mapping features to knowledge in a homogeneous manner that considers all input features equally, the problem will be split into two halves. One half will map sub-groups of features to knowledge at a constituent level, whereas the other will map all features to knowledge at a holistic level.

(ii) Represent BBKs in a heterogeneous manner. Different sized blocks of knowledge can be recombined in a recursive manner, i.e. a holistic block can be (re)used as a constituent block at a higher level of abstraction.

(iii) Identify and reuse the relevant BBKs to efficiently resolve complex problems, i.e. those consisting of patterns of patterns.

5.1.2 Chapter Organisation

The remainder of this chapter is organized as follows. Section 5.2 describes the critical components, architecture, and learning methodology of the novel lateralized system based on the developed lateralized framework. An illustrative walk-through of the developed algorithm is presented in Section 5.3. The experimental setup, problem domains, and learning order are explained in Section 5.4. Section 5.5 presents the work done to evaluate the effectiveness of the developed system. The computational overhead of the irrelevant problems and interpretation of decisions are presented in Section 5.6. Section 5.7 provides a further analysis of the developed lateralized system. Finally, the chapter summary is presented in Section 5.8.

5.2 Lateralized System

The knowledge associated with a learned problem needs to be stored for future use. Firstly, Code fragments (CFs) link represented environmental features through functional nodes. A disjunctive normal form of CFs constitute a rule, which encapsulates how well CFs link together to classify the problem. These rules are combined in a population, which enables specific niches of the problem to be combined together to solve the problem domain.

This knowledge associated with a problem is stored as a block of knowledge in the knowledge pool. This block of knowledge, termed *concept*, includes the set of rules, CFs, and attributes, i.e. the length of an instance, e.g. 3-bit Parity has length 3, and unique ID, regarding a learned problem domain. At a constituent level, the conditions in the classifiers are the encoded features of the input problem instance. As concepts have unique IDs and may appear in the terminals of a CF, they can be used in the conditions of the holistic level decision-making process. Conventional LCSs have a set of rules (population) for each learned problem. The lateralized system stores knowledge for all the learned problems, so may have more than one population. Therefore, the lateralized system stores learned knowledge in the form of a set of concepts in the knowledge pool. A concept has sufficient knowledge to solve any instance of its associated problem. A schematic illustration of concepts is shown in Fig. 5.1, with length, number of rules, and number of CFs (both in condition and action of the 'if <conditions> then <action>' rules) attributes.

The lateralized system needs to first determine if any knowledge has already been learned about the patterns in the input signal. This phase of learning is called the *knowledge identification* phase, which is critical as the system uses different learning methods determined on the quality (e.g accuracy of prediction) of the existing knowledge to solve the problem. The novel system passes the input signal to the left-half (to consider con-

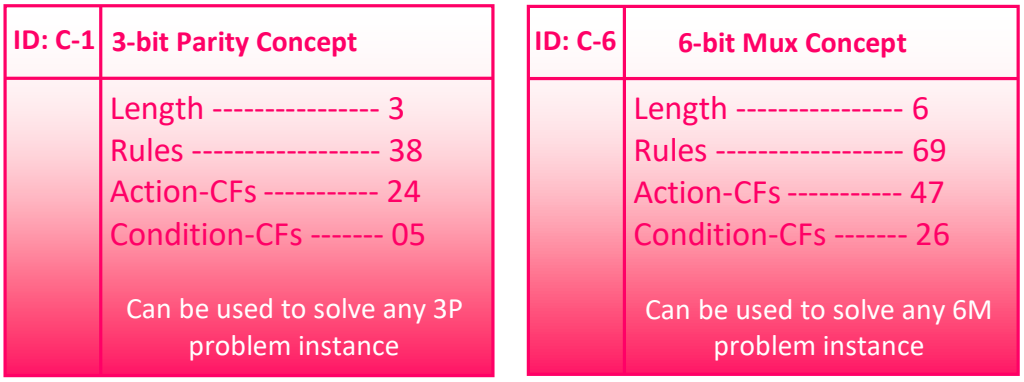


Figure 5.1: An illustration of the contents of the 3-bit Parity (C-1) and 6-bit Mux (C-6) concepts.

stituent patterns) and to the right-half (to consider higher-level holistic patterns) at the same time, see Fig. 5.2. The methods that determine the quality of the knowledge regarding the input signal are termed left hemispheric stratagem module (LHSM) and right hemispheric stratagem module (RHSM), respectively.

The subsequent *resolve problem* phase considers the quality of the identified knowledge, i.e. independently solve the problem (e.g accuracy of prediction is 100% or above a threshold), cooperate to solve the problem, or irrelevant to solve the problem. Consequently, it proceeds the action of the system through three methods, i.e. no learning needed, new learning from existing knowledge, or new learning from scratch. Despite this difference in methods to determine the action, the overall state-action-reward scheme of the novel lateralized system is similar to standard RL [194].

5.2.1 Knowledge Identification

Each environmental instance of the given problem is presented to both the LHSM and RHSM simultaneously in an on-line fashion. At the start of the learning process, the knowledge pool is empty. In such a scenario, both

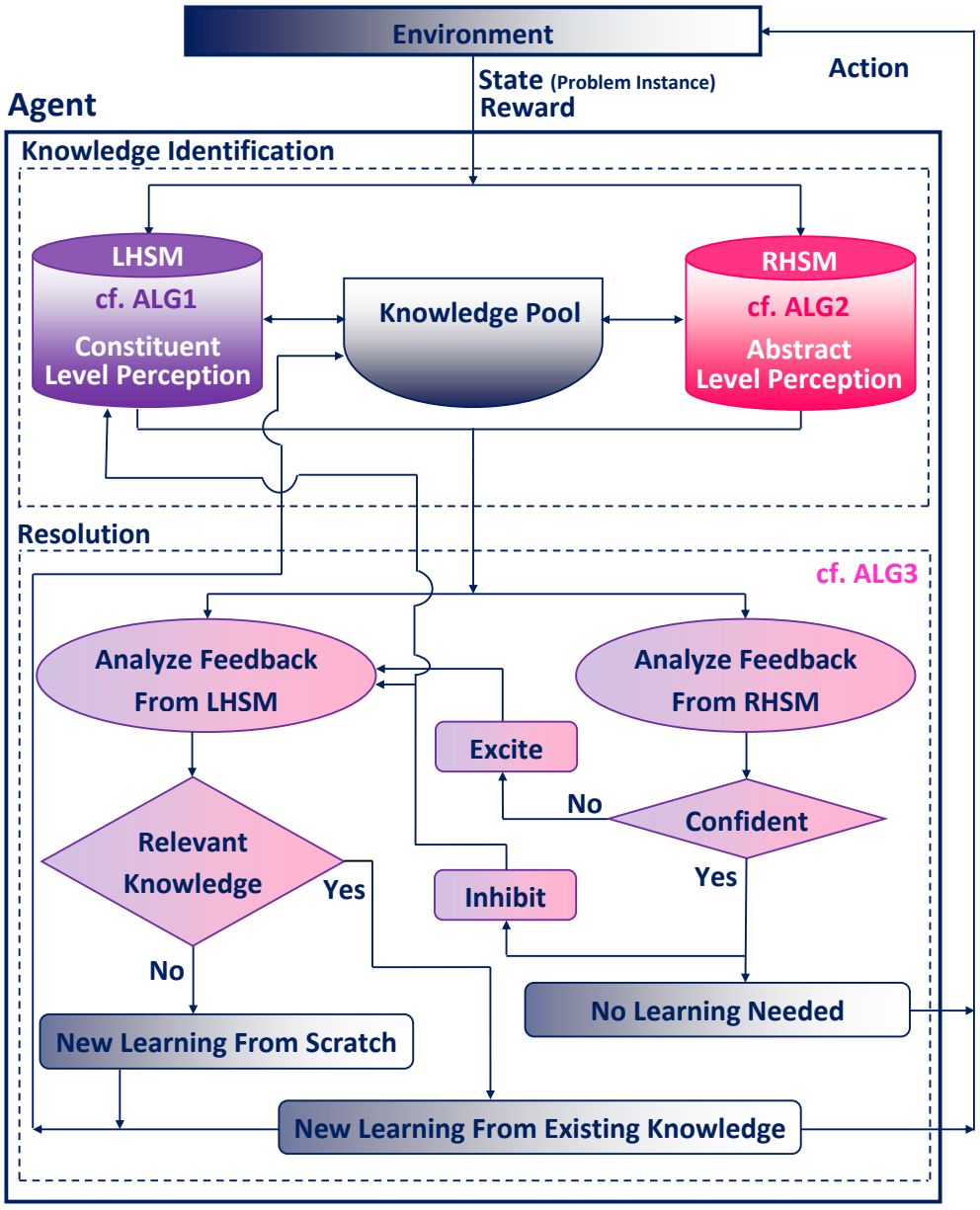


Figure 5.2: A schematic illustration of the strategies developed to achieve cognitive inspired functionality in the lateralized system. RHSM and LHSM represent right hemispheric stratagem module and left hemispheric stratagem module, respectively. (color key: constituent, holistic, and mix knowledge proceedings are represented by purple-white, pink-white, and light purple-pink gradients, respectively)

the LHSM and RHSM do not perform any computation and the system starts learning from a tabula rasa. The system is said to have learned a problem when its accuracy reaches 100% (or a given threshold). Hence, the associated concept is transferred to the knowledge pool.

The LHSM needs to identify those learned blocks of knowledge (concepts) that could be combined together to form the input instance. Concepts store their length, i.e. the number of stored features. The LHSM iteratively searches through the knowledge pool to identify those concepts that have a length that is a factor of the length of the current instance. That is, repeating those concepts could form the complete instance. The identified concepts (termed constituent-concepts) could form the lower layer of a solution to a hierarchical problem. For each constituent-concept, the LHSM splits the given problem instance into parts such that the length of each part is equal to the length stored in the concept. Subsequently, each part is resolved by utilizing the constituent-concept in exploit mode (i.e. by selecting the best action). The resulting action bits are concatenated together to form an abstract layer, to be treated as a separate learning problem to be solved by the LHSM only.

Now the LHSM again searches through the knowledge pool to identify those concepts that have a length equal to the length of the abstracted layer problem instance. The identified concepts are termed holistic-concepts. The LHSM generates the candidate groups of concepts by pairing the constituent concept with each holistic-concept. For example, the system is tasked to solve a problem instance of length 18 bits (features). Suppose that the lower layer constituent-concept C_1 has a length 3 (the number of stored features). The resultant upper layer generated by utilizing the C_1 has a length 6 (because C_1 is be repeated for 6 parts). Suppose that there are two learned concepts (C_5 and C_6) that have length 6. These concepts are selected as holistic-concepts. The resultant candidate groups of knowledge are: (i) lower layer constituent-concept C_1 and upper layer holistic-concept C_5, and (ii) lower layer constituent-concept C_1 and upper layer

holistic-concept C_6. This process is repeated for each constituent-concept.

The LHSM evaluates each candidate group by performing a preliminary test, for example by utilizing 1000 environmental instances. During this evaluation process, the LHSM addresses a given problem instance by utilizing constituent-concept and holistic-concept in exploit mode at the lower layer and higher layer of the problem instance, respectively. Subsequently, if the prediction accuracy is equal to (or above) a given threshold, the candidate group is marked (a flag is set) as relevant knowledge, otherwise, it will be discarded. Finally, the LHSM passes the relevant knowledge groups to the resolve problem method. The pseudo-code of the strategy adopted by the LHSM for the identification of the relevant knowledge is given in Algorithm 4 and an illustrative walk-through is presented in the Section 5.3.

The RHSM needs to identify those learned blocks of knowledge that have sufficient knowledge to solve the problem independently (prediction accuracy is 100% or above a threshold). The concepts in the knowledge pool that have a length equal to the length of the given instance are termed as candidate-concepts.

The RHSM computes confidence for each candidate concept. *Confidence* is the prediction accuracy of the RHSM in exploit mode. The RHSM computes confidence in two steps. Initially, the RHSM performs a preliminary test for each candidate concept. During the preliminary test, the RHSM solves 1000 problem instances by utilizing the candidate concept in exploit mode and computes the confidence. If the confidence is below a specific threshold, e.g. 90%, the candidate concept is removed from the candidate list. This process is repeated for each candidate concept.

In the next stage, the RHSM performs an intermediate level test for each candidate concept remaining in the list. During this stage, the RHSM is tasked to solve more problem instances, i.e. $5000\times$length of the given problem instance. The RHSM solves these problem instances by utilizing the candidate concept in exploit mode and computes the confidence again.

Algorithm 4: Strategy adopted by the LHSM to identify the candidate groups of concepts that have the potential to efficiently learn the given problem (cf. Fig.5.2).

Data: The environment and problem configurations
Result: Relevant Groups of Concepts

```
Initialize the parameter settings and global variables;
Identify Constituent-Concepts();   % Generate a list of concepts that have a length (pre-determined value) which is a factor of the length of the current problem instance.
for Each Identified Constituent-Concept do
    Split Problem();   % Split the problem instance into parts (sub-problems) such that each part has length equal to the length of the constituent-concept.
    Solve Sub-Problems();   % Solve each part by utilizing the constituent-concept in exploit mode.
    Get Higher Layer();   % The actions are concatenated to form a higher layer instance.
    Identify Holistic-Concepts();   % Identify concepts that have a length equal to the length of the higher layer instance.
    Generate Candidate Groups ();   % Group pairs of the constituent-concept with each holistic-concept.
end
Preliminary Test();   % Perform preliminary test for each candidate group by utilizing 1000 problem instances.
Mark Relevant Knowledge();   % Mark candidate groups of concepts that have a prediction accuracy equal to (or above) a given threshold.
Return Relevant Knowledge;
```

If a confidence score above a defined threshold is produced, the RHSM identifies that it has seen this problem before. Otherwise, the RHSM removes this candidate concept from the candidate list. This process is repeated for each candidate concept. Finally, the RHSM shares the list of candidate concepts, along with their confidence values, with the resolve problem module, see Algorithm 5.

5.2.2 Resolve Problem

The resolve problem module receives feedback about the candidate concepts and candidate knowledge groups from RHSM and LHSM, respectively. During the analysis of the feedback from RHSM, the resolve problem module iteratively searches through the list of candidate concepts to find a concept that has confidence value 100% (or above a given threshold). If such a concept is found, the resolve problem module marks the given problem as an already learned problem. Consequently, the system runs in exploit mode and utilizes that concept to solve any future problem instances given by the environment. Moreover, the mode is set as *no learning is needed*. The resolve problem module generates an inhibit signal so that LHSM cease and further analysis on the feedback from LHSM is stopped. Else, the resolve problem module generates an excite signal to LHSM to identify the relevant knowledge.

During the analysis of the feedback from LHSM, if the relevant knowledge groups exist in the list. The resolve problem module iteratively searches through all the relevant knowledge groups and forms a list of all the concepts that are present in those groups. Subsequently, the resolve problem module marks all those concepts and their associated CFs (stored with those concepts) as relevant knowledge. The mode is set as *learning from existing knowledge*. Consequently, the system learns the given problem by utilizing all the relevant knowledge, i.e. automatically uses CFs and concepts in the classifiers. However, if the resolve problem module could not find any relevant knowledge group in the feedback from LHSM. The

Algorithm 5: Strategy adopted by the RHSM system to identify the candidate concepts that can confidently solve a problem (cf. Fig.5.2).

Data: The environment and problem configurations
Result: The candidate concepts that can confidently solve the given problem

```
Identify candidate concepts();   % Generate a list of concepts that have a length equal to the length of the current problem instance.
for Each Identified Concept do
    Preliminary Test();   % Perform preliminary test for each candidate concept by utilizing 1000 problem instances.
    Compute Confidence();   % Confidence (prediction accuracy) of each concept to solve the problem.
    if Concept Confidence < Threshold then
            % The confidence is below 90%.
        Remove Candidate Concept();   % Remove the candidate concept from the candidate list.
    end
end
for Each Candidate Concept do
    Intermediate Test();   % Perform intermediate level test for each candidate concept by utilizing (5000×length of the given problem instance) problem instances.
    Compute Confidence();   % Confidence of each concept to solve the problem.
    if Concept Confidence > Threshold then
            % The confidence is above 90%.
        Save Confidence Value();   % Save the confidence value for the candidate concept.
    else
        Remove Candidate Concept();   % Remove the candidate concept from the candidate list.
    end
end
Return Candidate Concepts And Confidence Values();
```

mode is set as *new learning from scratch*. In such a scenario, the system behaves as an ordinary LCS and learns the given problem from a tabula rasa. The overall pseudo-code of the novel strategy adopted by the lateralized system to solve a problem is given in Algorithm 6.

It is important to note that as each subsequent problem is learned the concept, length (number of unique features addressed), and associated CFs are stored in the knowledge pool. The related CFs can be used separately from the concepts to seed feature learning. A CF stores the relationship between features and targets. The individual CFs are used together in the condition and action of rules. Consequently, it enables niches to form and heterogeneity to occur. These rules are encapsulated in concepts, which store the solution for a part of the problem or whole problem. Moreover, a concept may be a higher-level of knowledge w.r.t. one problem, while being also a lower level of knowledge w.r.t. another problem.

5.2.3 Learning Methodology

The idea of lateralization could be implemented by utilizing any EML based system, where Learning Classifier Systems (LCSs) are selected due to their niche-based algorithm and built-in support for heterogeneity (i.e. different rules co-exist in the same population). These two important features make LCSs an ideal candidate to support lateralization, which extends the concepts of niches and heterogeneity to the different levels of a problem's composition. The learning methodology of the lateralized system is developed by utilizing the framework of accuracy-based LCSs, i.e. Wilson's XCS [251]. The property of XCS to generate a complete and accurate solution enables the lateralized system to capture all the BBKs required for a concept. The learned concepts and associated CFs can be used at different levels of abstraction. Moreover, the use of CFs provides improved expressivity and scalability as compared to the ternary alphabet [15]. These characteristics of CFs assist the lateralized system to efficiently handle diverse knowledge. The lateralized system enhances the state-of-

Algorithm 6: Overall strategy adopted by the lateralized system to solve a problem (cf. Fig.5.2).

Data: The environment and problem configurations.

Result: The given problem is identified as no learning needed, new learning from existing knowledge, or new learning from scratch.

```
Initialize the parameter settings and global variables;
Knowledge Identification
    RHSM();        %Algorithm 1.
    LHSM();        %Algorithm 2.
Resolve Problem
    Analyze RHSM Output();        %Iteratively search through the list
     of candidate concepts and check their confidence values.
    if Concept Confidence = 100% then
            %Confidence value 100 (or above a given threshhold).
        Run in Exploit Mode Only();
        Generate Inhibit Signal();        %Generates an inhibit signal so
         that LHSM cease and further analysis on the feedback from
         LHSM is stopped.
        Set No learning is Needed();
    else
        Generate Excite Signal();
    end
    Analyze LHSM Output();
    if Relevant Knowledge Groups Exist then
            %Iteratively search through all the relevant knowledge
         groups.
        Form List of Concepts();        %Forms a list of all the concepts
         that are present in the relevant knowledge groups.
        Mark Relevant Knowledge();        %Marks all those concepts
         and their associated CFs (stored with those concepts) as a
         relevant knowledge.
        Set Learning From Existing Knowledge();
    else
        Set Learning From Scratch();
    end
```

the-art CF-based XCS [236] in the following modules:

Condition and Action of Classifier

State-of-the-art CF-based systems have CFs either in the condition or in the action of the classifier [51, 15, 236]. The novel lateralized system has CFs in both the condition and the action of the classifier. To the best of our knowledge, this is the first time that CFs are included in both. The inclusion of concepts in the CFs and then CFs in both condition and action of the classifier enables the lateralized system to have a complex and heterogeneous representation of knowledge, reduce search space, generate compact rules, and have a lateralized population of rules.

Code Fragment Enhancement

The lateralized system enhances the CFs such that the leaf nodes have the ability to randomly select concepts or other CFs or environment variables (see Fig. 5.3). This inclusion of concepts (blocks of knowledge) in the CF enables the classifier to independently resolve a large part of the problem.

Lateralized Population

Two contributions are inherent in the population of rules evolved by the lateralized system, i.e. (i) CFs include concepts from different levels and domains in the knowledge pool. Thus, (ii) the set of rules that are evolved to solve a problem have diverse CFs from the learned BBKs. Consequently, the system incorporates diverse knowledge that enables the inclusion of a population of rules at different levels of abstraction. Thus the population of rules is termed as a lateralized population.

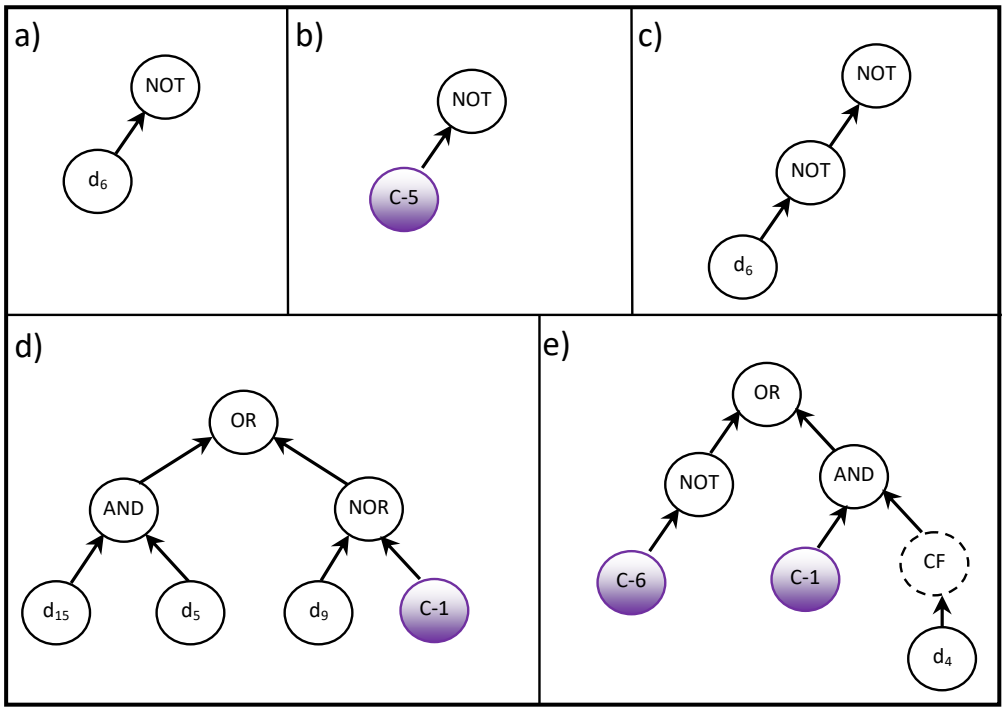


Figure 5.3: A schematic illustration of different CFs. D_1, D_2, – D_n are the input condition values. AND, OR, NAND, NOR, are the functions. C_1, C_2, –, C_6 are concepts. Note the double negative in c. which is a small amount of bloat.

5.3 Walk-Through of the Algorithms

This section presents a walk-through the algorithms of the lateralized system with illustrative examples. Let us suppose that the system has already learned seven concepts, i.e. C_1 (3-bit Parity), C_2 (3-bit Majority-on), C_3 (4-bit Parity), C_4 (5-bit Parity), C_5 (6-bit Parity), C_6 (6-bit Mux), and C_7 (7-bit Parity). The learned knowledge (CFs and concepts) is stored in the knowledge pool. The examples of learned concepts, CFs, and rules are shown in Fig. 5.1, Fig. 5.3, and Fig. 5.4, respectively.

(i)	If Not(D_6)	→ C-5
(ii)	If Not(Not D_6)	→ Not(C-5)

Figure 5.4: A schematic illustration of simple rules.

The system is tasked to learn 18-bit HMux problem such that the lower layer consists of 3-bit Parity problem and upper layer consists of 6-bit Mux problem, see Fig. 5.5.

The LHSM checks the length of each learned concept stored in the knowledge pool, e.g. the lengths of C_1, C_2, C_3, C_4, C_5, C_6, and C_7, are 3, 3, 4, 5, 6, 6, and 7, respectively. If the length of a concept is a factor of the length of the given problem (18), it can be selected as a constituent-concept, e.g. the lengths of concepts C_1, C_2, C_5, C_6 are factor of 18, they are selected as a constituent-concepts; whereas, the lengths of concepts C_3, C_4, C_7 are not the factor of 18, they are ignored. Suppose that initially the LHSM selects C_1 as a lower layer candidate knowledge. Subsequently, the LHSM figured out that the selected C_1 concept has length 3 and it could be repeated 6 times to completely cover any given instance of 18-bit HMux problem. The LHSM splits the given problem instance into 6 chunks such that each chunk has length 3. The LHSM iteratively solve each chunk by utilizing C_1 in exploit mode. The results are concatenated and they form a 6-bit upper layer problem instance.

Now the LHSM again checks the length of each learned concept stored in the knowledge pool. If the length of a concept is equal to the length of the upper layer problem instance, it can be selected as a holistic-concept w.r.t. the lower layer constituent-concept, e.g. the lengths of concepts C_5 and C_6 are 6, they are selected as holistic-concepts w.r.t. C_1, whereas, all others are ignored. The LHSM generates two candidate knowledge groups (CKGs) by pairing C_1 with each holistic-concept (C_5 and C_6), i.e. (CKG-i) lower layer constituent-concept C_1 and upper layer holistic-concept

C_5, and (CKG-ii) lower layer constituent-concept C_1 and upper layer holistic-concept C_6. This process is repeated for the remaining constituent-concepts, i.e. for C_2, C_5, and C_6.

In this initial lateralized system, only one concept may be repeated as a constituent for a large problem. In subsequent versions of the system, it would be possible to allow the combinations of different concepts to be utilized as constituents for a large problem. This may result in a combinatorial explosion, which could slow-down the system. Moreover, it is hard, even for a human, to handle such a heterogeneous split-up of a large problem.

The LHSM performs a preliminary test on all the CKGs to identify the relevant knowledge and discard the irrelevant knowledge. For this purpose, the LHSM evaluates each CKG for 1000 problem instances of the 18-bit HMux problem. During the evaluation process of a CKG, if the final output of an environmental instance generated by the CKG is different from the ground truth value. The LHSM stops the evaluation process for that CKG and discards it. Otherwise, the LHSM considers that CKG as relevant knowledge.

Preliminary test for CKG-i: Let us suppose that the LHSM receives the 18-bit HMux problem instance 101100101110010100 with ground truth 0. The LHSM splits the given problem instance into the chunks of length 3, equal to the length of lower layer C_1, i.e. 101, 100, 101, 110, 010, and 100. Subsequently, the LHSM utilizes constituent-concept C_1 (3-bit Parity concept), in exploit mode, on each chunk as a separate problem instance. The results of C_1 for the chunks are $0, 1, 0, 0, 1,$ and 1 respectively. These results are concatenated to form an upper layer problem instance, i.e. 010011. The LHSM utilizes the upper layer holistic-concept C_5 (6-bit Parity concept) on 010011 in exploit mode and generates a result 1. This result is different from the ground-truth. Consequently, the LHSM stops the evaluation process for CKG(i) and discards it.

Preliminary test for CKG-ii: Let us suppose that the LHSM gets the 18-

bit HMux problem instance 101100101110010100 with ground truth 0. The LHSM splits the given problem instance into the chunks of length 3, equal to the length of lower layer C_1, i.e. 101, 100, 101, 110, 010, and 100. Subsequently, the LHSM utilizes constituent-concept C_1 (3-bit Parity concept), in exploit mode, on each chunk as a separate problem instance. The results of C_1 for the chunks are $0, 1, 0, 0, 1,$ and 1 respectively. These results are concatenated to form an upper layer problem instance, i.e. 010011. The LHSM utilizes the upper layer holistic-concept C_6 (6-bit Mux concept) on 010011 and receives a result 0. This result is the same as that of the ground-truth. The LHSM gets another instance of an 18-bit HMux problem and repeats the above-mentioned procedure. If the result generated by CKG(ii) and the ground-truth are the same for 1000 random instances of 18-bit HMux problem, the CKG(ii) is considered as relevant knowledge. Otherwise, it will be discarded. In noisy problems, an error threshold (CF ε_0 [x]) could be included.

The above mentioned preliminary test is applied to all the CKGs. The CKGs that successfully completed the preliminary test are marked (a flag is set) as relevant knowledge and they are passed to the resolve problem technique. The walk-through of the remaining components is very simple and straight-forward, hence is not presented.

5.4 Experimental Design

5.4.1 Problem Domains

This chapter seeks to show the effectiveness of the lateralization approach for classification problems to obtain a proof-of-concept. This can be achieved by conducting a range of experiments on well-known complex Boolean problems. Boolean problems can exhibit heterogeneity and epistasis which are characteristics known to cause problems in classification techniques as these can not make the assumption linear separable. They also pos-

sess identifiable components of a solution, rather than the problem of the solution, that are transferable to other problems which helps to evaluate transfer learning abilities, i.e. the ability to identify and transfer important parts of knowledge. Boolean problems have exactly known solutions so that the exactness of the produced solution can be interrogated, e.g. Mux problems, Parity problems, Majority-on problems, and Carry problems. The scalability of the lateralized approach was investigated by utilizing more complex derived versions of these problems such as hierarchical multiplexer problems and high scale Parity problems (i.e. beyond the common 7-bit problem). Hierarchical problems have an additional layer of complexity, so they are hard to resolve as they have low sparsity and hierarchical distribution of knowledge [51, 244].

Boolean problems have measurable search space, dependency structure, and distributed niches (subsolutions) [244]. Although Boolean problems can be considered as 'toy' benchmark problems, the heterogeneity and epistasis characteristics make them analogous to real-world problems, such as finance, bioinformatics, and behavior modeling. Hence, many state-of-the-art systems have been evaluated by using Boolean problems [316, 317, 15, 278, 51].

The Mux problems are multi-modal and have epistasis characteristics, i.e. address bits are critical as they determine the importance of data bits [244, 259]. It is hard to make any useful generalization for Parity problem domains in the standard ternary alphabet of XCS [244]. To the best of our knowledge, 2-bit to 7-bit Parity problems have been used in the literature as the interaction between all bits is important, i.e. no redundant features, which can be discarded/removed. Recently, 8-bit Parity problems were used to test the effectiveness of the evolutionary multi-task learning system [317]. Whereas, the lateralized system is to address much higher scale problems, e.g. 16-bit Parity problem. The solutions of the Majority-on problems have strongly overlapping rules, which make it difficult to learn complete solutions. The Carry problems are considered as niche imbal-

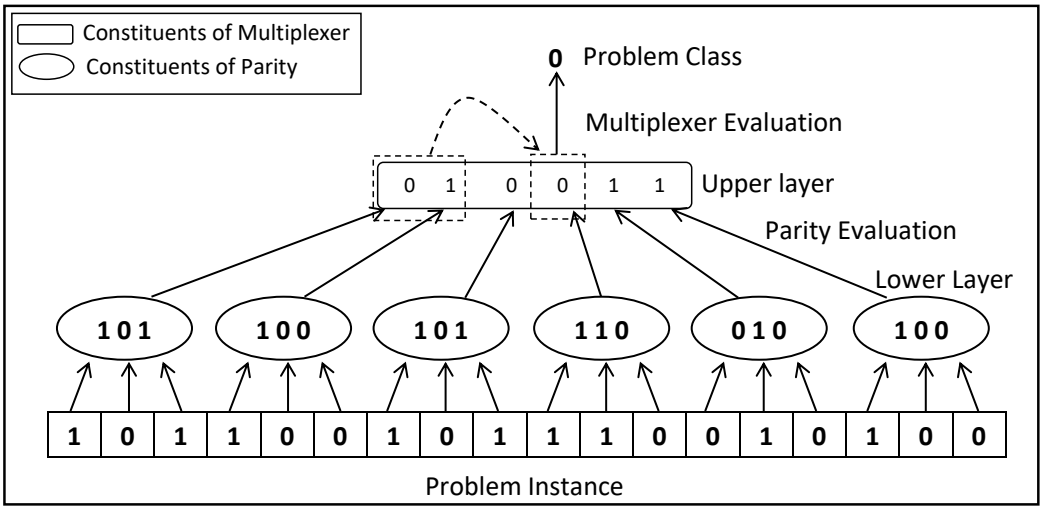


Figure 5.5: An instance of 18-bit hierarchical Mux problem. The lower layer consists of 3-bit Parity problems, whereas the upper layer is a 6-bit Mux problem.

ance problems and the solutions to these problems have strongly overlapping rules [244].

Hierarchical problems have an additional layer of complexity, so they are hard to resolve as they have low sparsity and hierarchical distribution of knowledge [51, 244]. Therefore, these problems are the best candidates with which to evaluate the effectiveness of the lateralized system. The hierarchical Boolean problems, presented here, consist of two layers. The lower layer is composed of multiple instances of a specific Boolean problem. The evaluation and integration of the lower layer generate an instance of the upper layer. The upper layer is another Boolean problem. For example, an 18-bit hierarchical problem may be composed of the following layers (i) lower layer based on a 3-bit Parity problem; and (ii) upper layer based on the 6-bit Mux problem. An example instance of an 18-bit HMux is shown in Fig. 5.5.

5.4.2 Learning Order

The order with which sub-problems are presented to an EML system plays an important role in learning [51, 15, 236]. Initially, the lateralized system could be trained with a diverse set of sub-problems. The learned knowledge is stored in the knowledge pool. The lateralized system can automatically (without human involvement) identify the relevant required building blocks of knowledge from the knowledge pool (if they exist). Instead of trying each block of learned knowledge, the lateralized system adopts an efficient strategy to identify the relevant BBKs from the knowledge pool. To identify higher-level relevant knowledge, the RHSM only evaluates the learned concepts that have the same length as that of the given problem instance. If this higher knowledge is not present, the LHSM analyses only those concepts that could be repeated to form a complete problem instance. Moreover, if the required knowledge is completely absent, the system considers it a novel problem and learns from a tabula rasa. In this case, the lateralized system behaves similarly to a conventional LCS.

For all the experiments, the lateralized system and CF-based LCS [15] (conventional LCS does not utilize sub-problems) are trained with the same set of relevant sub-problems in the same order. Moreover, a separate set of experiments for the lateralized system is conducted to calculate the computational overhead due to extraneous sub-problems (see Section 5.6.1).

5.4.3 Experimental Setup

The lateralized system uses the LCS parameter values that have been commonly used in the literature [244, 318]. These values are: learning rate $\beta = 0.2$; prediction error threshold $\varepsilon_0 = 10$; fitness fall-off rate $\alpha = 0.1$; fitness exponent $\nu = 5$; GA threshold $\theta_{GA} = 25$; crossover probability $\chi = 0.8$; mutation probability $\mu = 0.04$; deletion threshold $\theta_{del} = 20$; deletion fraction $\delta = 0.1$; subsumption threshold $\theta_{sub} = 20$; don't care prob-

Table 5.1: Population Size

Problem	Population Size
2-bit Parity	200
3-bit Parity	300
4-bit Parity	400
5-bit Parity	500
6-bit Parity	1000
7-bit Parity	2000
8-bit Parity	2500
9-bit Parity	3000
10-bit Parity	3500
11-bit Parity	4000
12-bit Parity	4500
13-bit Parity	5000
14-bit Parity	5500
15-bit Parity	6000
16-bit Parity	6500
6-bit Mux	500
11-bit Mux	1000
20-bit Mux	2000
37-bit Mux	5000
18-bit Hierarchical Mux	20000

ability $= 0.33$; fitness reduction $= 0.1$; prediction error reduction $= 0.25$; prediction reward $= 1000$. Additional parameter settings for XCS with Bayesian optimization algorithm (XCS/BOA) [314] are: error BOA 400; $\theta_{BOA} = 20$; maximum parents 4; minimum population 0; local population 10; MCMC updates 18. Moreover, all the experiments have been repeated 30 times and the values presented here are obtained by averaging the results of those experiments. Furthermore, the population size used for each

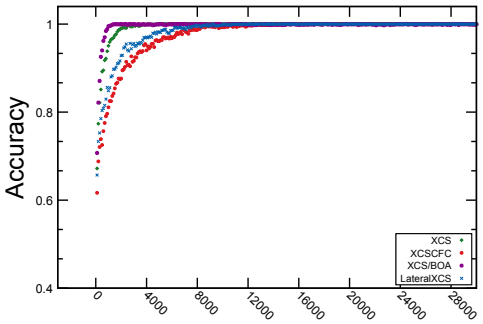
experiment is presented in the Table 5.1.

5.5 Experiments

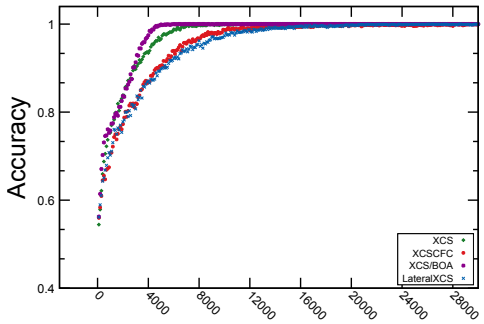
The lateralized system is bootstrapped with five hard-coded Boolean functions i.e., AND, OR, NAND, NOT, NOR. The system does not learn these basic functions. These functions are given to the lateralized system as they are commonly supplied to the EML systems in the literature [15, 236]. Initially, the lateralized system learns simple problems such as the 3-bit Parity problem, 3-bit Carry problem, and 6-bit Mux problem. Subsequently, the system is trained with higher-level problems such as 6-bit Carry, 11-bit Mux, and 18-bit HMux problems. During the learning of these problems, the system is tasked to identify and utilize the relevant BBKs from previously learned knowledge. Consequently, the system efficiently learns new problems and acquires the ability to resolve higher-level complex problems. The experimental results generated by the lateralized system (LateralXCS) are compared with the results generated by the conventional LCS (XCS), CF-based LCS (XCSCFC) [236], XCS/BOA [314], conventional GP, LLGP [276], and CGP [279] techniques.

5.5.1 Multiplexer Problems

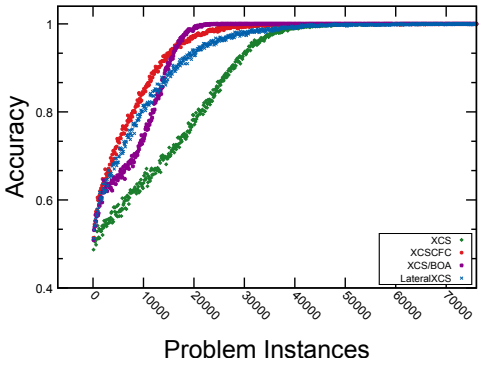
The first set of experiments was conducted by using Mux problems to obtain a proof-of-concept of the lateralized system. All the LCS systems (XCS, XCSCFC, XCS/BOA, and LateralXCS) managed to learn 6-bit, 11-bit, and 20-bit Mux problems. For these problems, the learning pace of XCS/BOA is better than all other systems. The first problem presented to the systems was a 6-bit Mux problem. Both XCSCFC and LateralXCS learned this problem from scratch. Consequently, the learning pace of XCS/BOA is higher than XCSCFC and LateralXCS. Moreover, the overhead of XCSCFC is more than LateralXCS, see Fig-5.6a. Subsequently,



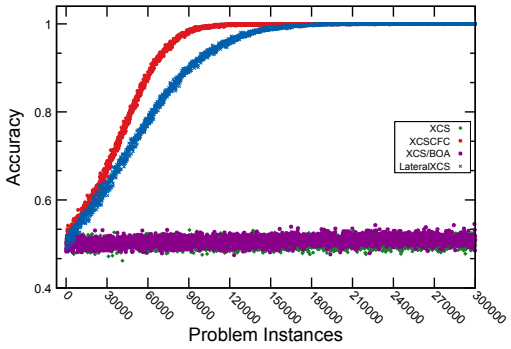
(a) 6-bit Mux problem



(b) 11-bit Mux problem problem



(c) 20-bit Mux problem problem



(d) 37-bit Mux problem

Figure 5.6: Experimental results of Mux problems using conventional LCS (XCS), CF-based XCS (XCSCFC), XCS/BOA, and Lateralized XCS (LateralXCS)

the systems were trained to learn an 11-bit Mux problem. During this learning, both XCSCFC and LaterXCS utilized learned knowledge from the 6-bit Mux problem. The learning pace of XCSCFC and LaterXCS is almost similar but lagging behind XCS and XCS/BOA, see Fig-5.6b. The next problem presented to the systems was a 20-bit Mux problem. The learning pace of XCS/BOA is better than all other systems. Both XCSCFC and LateralXCS utilized the learned knowledge from 6-bit and 11-bit Mux problems. Consequently, their learning pace is higher than XCS, see Fig-5.6c.

The experimental results show that both XCS and XCS/BOA were unable to learn 37-bit Mux problems, whereas, XCSCFC and LateralXCS successfully learned the problem by utilizing $200,000$ problem instances, see Fig. 5.6d. The learning pace of the LateralXCS is slower than XCSCFC from $30,000$ to $120,000$ problem instances. This occurs because the LateralXCS has to identify the suitable building blocks from the pool of learned knowledge. The XCSCFC swiftly achieved 99.8% accuracy but struggled to learn the problem completely. However, the LateralXCS achieved 100% accuracy by utilizing $300,000$ problem instances. The average accuracy and standard deviation, for the last hundred runs, of LateralXCS and XCSCFC are 100 ± 0.00 and $99.99 \pm 9.52985e^{-05}$, respectively.

5.5.2 Parity problems

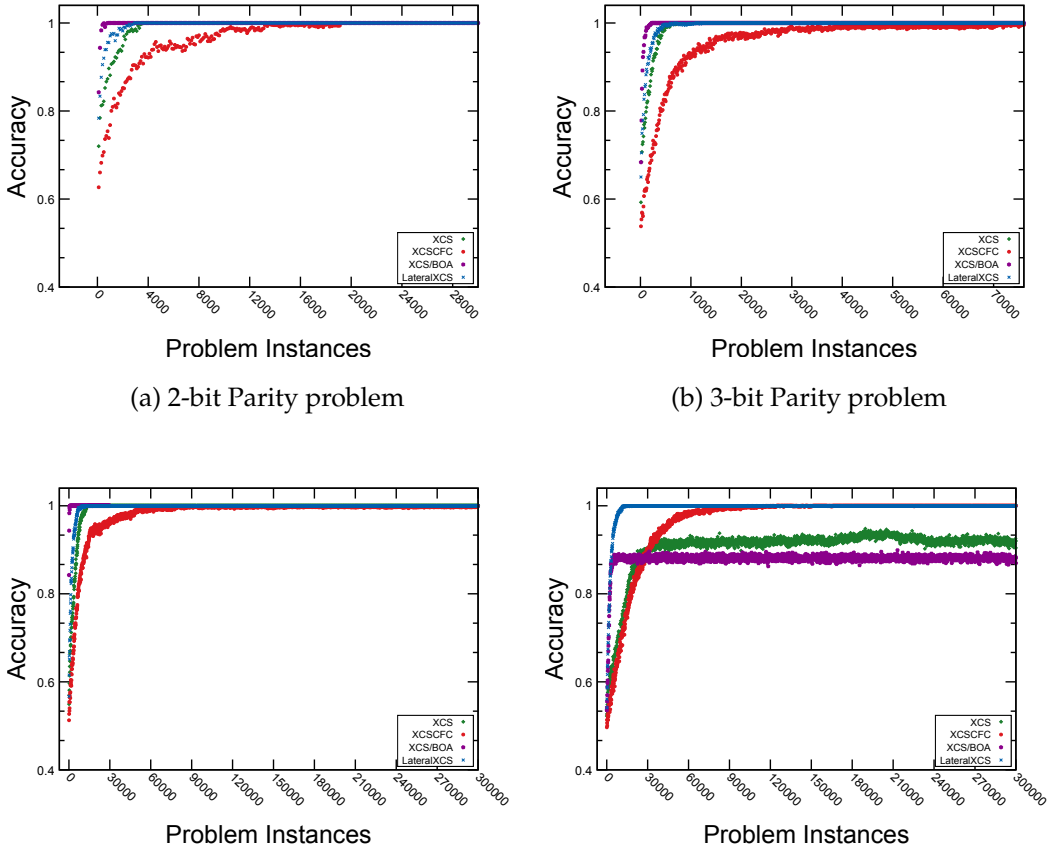
All the LCS systems (XCS, XCSCFC, XCS/BOA, and LateralXCS) managed to learn 2-bit, 3-bit, and 4-bit Parity problems. For these problems, the learning pace of XCS/BOA is better than all other systems. However, for 5-bit, 6-bit, and 7-bit Parity problems, XCS and XCS/BOA are not able to learn the problems completely, whereas, XCSCFC lags behind the LateralXCS. The first problem presented to the systems was a 2-bit Parity problem. Both XCSCFC and LateralXCS learned this problem from scratch. The utilization for CFs in the condition as well as in the action of a classifier empowers the LateralXCS to efficiently form accurate and maximum

general rules. Consequently, LateralXCS outperform XCS and XCSCFC, see Fig-5.7a. Subsequently, the systems were trained to learn 3-bit, 4-bit, 5-bit, and 6-bit parity problems, respectively. During the learning process, both XCSCFC and LaterXCS utilized learned knowledge from previous sub-problems and managed to obtain 100% accuracy for all parity problems. Moreover, LateralXCS outperformed XCS and XCSCFC in all parity problems, see Fig. 5.7.

The experimental results of the 7-bit Parity problem are shown in Fig. 5.8b. The LateralXCS successfully identified and reused the relevant building blocks of learned knowledge. Consequently, the system is able to efficiently learn the 7-bit Parity problem by utilizing only $15,000$ problem instances. In contrast, XCS, XCS/BOA, and XCSCFC learned 70%, 83% and 100% 7-bit Parity problem by utilizing $250,000$ problem instances, respectively. The interpretation of rules generated by LateralXCS shows that the Lateralized system automatically considers a higher-level Parity problem as a two-bit Parity problem. Three features of LateralXCS play a critical role in this regard, i.e., (i) ability to identify and utilize relevant learned *concepts*, (ii) ability to address a problem at different levels of abstraction, and (iii) ability to apply CFs at the condition as well as the action of classifiers.

The Wilcoxon signed-rank test was applied to statistically compare LateralXCS with XCSCFC (see Table 5.2). The test was conducted on the results of the 100 problem instances after the lateralized system achieved a performance accuracy of 100%. The second and third columns contain the average performance along with standard deviation. All P-values are less than 0.00001, which is evidence that the performance improvement of LateralXCS is statistically significant.

The XCS and GP systems are based on different evolutionary techniques and apply different strategies to address a problem. The main objective of the work presented here is not to develop a system that outperforms the GP-based system. However, their performance is compared to



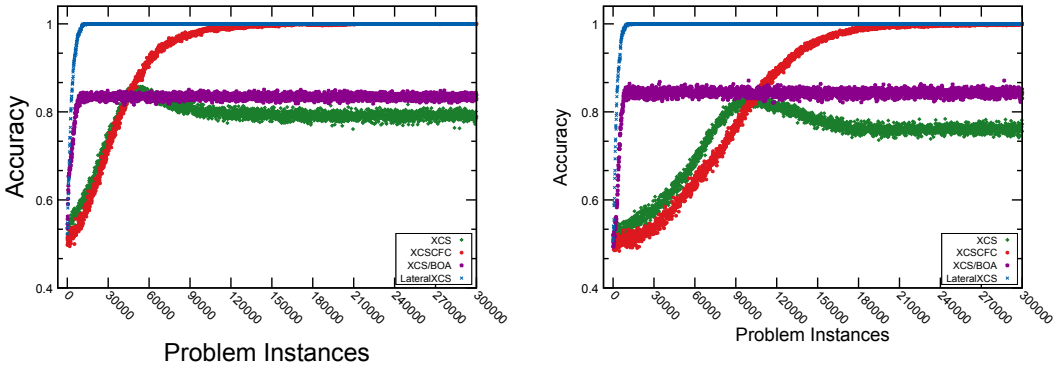
(a) 2-bit Parity problem

(b) 3-bit Parity problem

(c) 4-bit Parity problem

(d) 5-bit Parity problem

Figure 5.7: Experimental results of Parity problems using XCS, XCSCFC, XCS/BOA, and LateralXCS.



(a) 6-bit Parity problem

(b) 7-bit Parity problem

Figure 5.8: Experimental results of Parity problems using XCS, XCSCFC, XCS/BOA, and LateralXCS.

Table 5.2: Wilcoxon signed-rank test

Problem Domain	XCSCFC	LateralXCS	Z-Value	P-Value
7-bit Parity	65.71 ± 1.38	100.00 ± 0.00	-8.6818	<0.00001
18-bit hierarchical Mux	96.30 ± 3.40	100.00 ± 0.00	-8.6818	<0.00001

Table 5.3: Performance comparison with different GP systems

Problem Domain	GP	LLGP	CGP	Lateralized XCS
7-bit Parity	60.09	76.31	52.47	100.00
18-bit hierarchical Mux	89.09	88.89	54.28	100.00

show the effectiveness of the lateralized approach. The same set of Parity problem experiments was conducted by using GP, LLGP, and CGP systems. Each experiment was repeated for 30 times with 50 generations and 1024 population. The experimental results of the 7-bit Parity problem are shown in Table 5.3. None of the GP-based systems were able to completely learn the 7-bit Parity problem, whereas the LateralXCS achieved 100% accuracy by utilizing the equivalent number of problem instances.

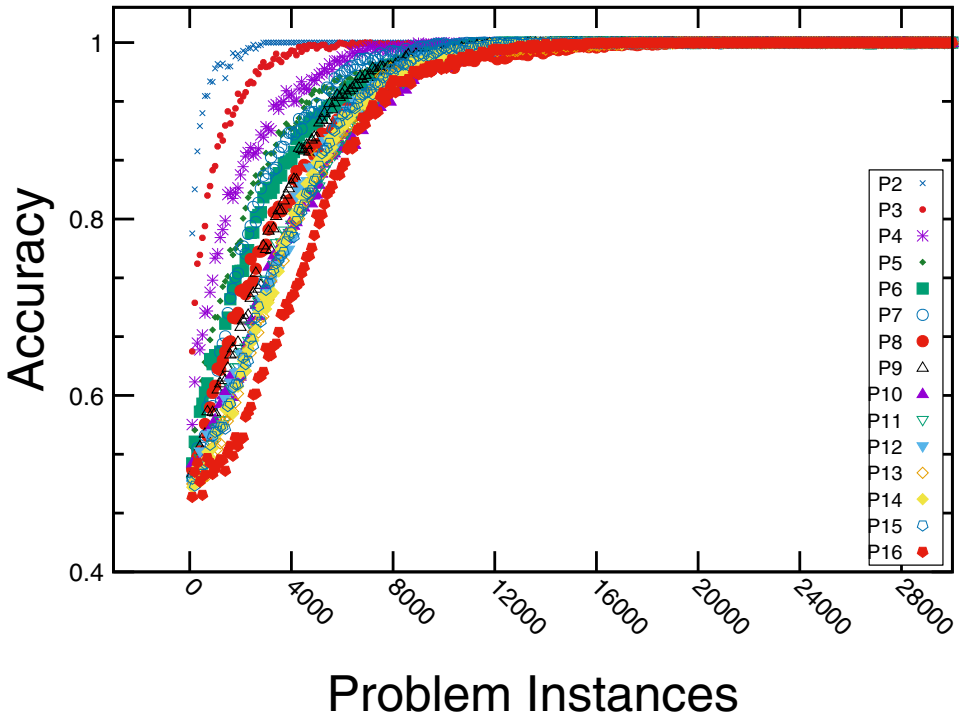


Figure 5.9: Experimental results of 2-bit to 16-bit Parity problems using LateralXCS.

The effectiveness of the LateralXCS also tested by utilizing higher-level Parity problems. The experimental results of 2-bit to 16-bit Parity problems using LateralXCS are shown in Fig.5.9. The LateralXCS efficiently achieved a performance accuracy of 100% by utilizing less than 20000 problem instances per problem. The performance scalability of the LateralXCS is explained in Section 5.6.2.

The accuracy achieved by XCS, XCSCFC, and LateralXCS after utilizing 50000 problem instances during the learning of 2-bit to 16-bit Parity problems is shown Fig.5.10. XCS could not usefully learn beyond the 7-bit Parity problem due to the lack of generalization in the ternary alphabet.

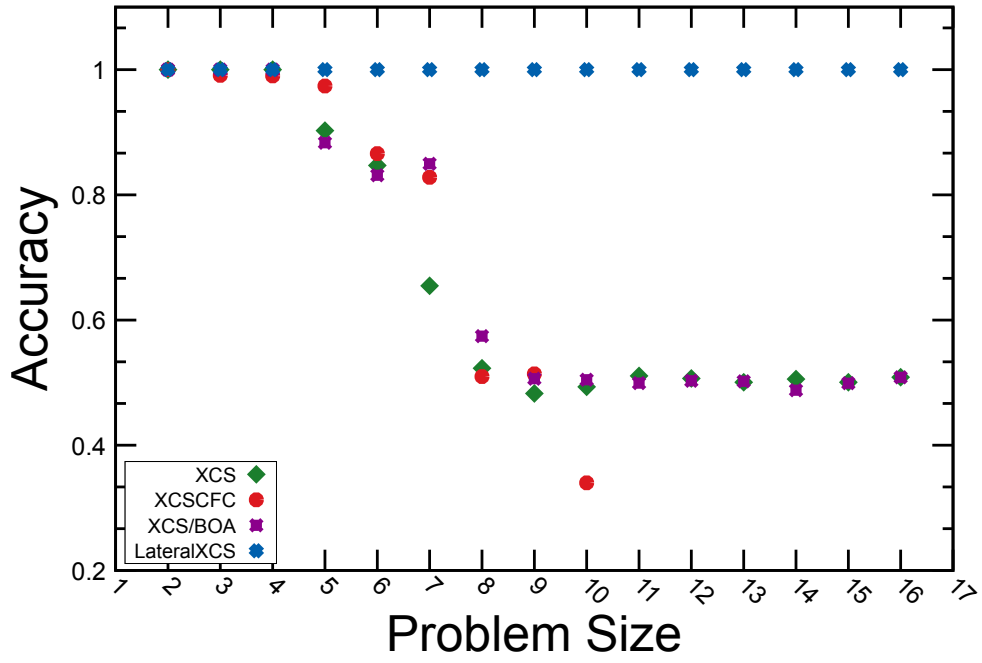


Figure 5.10: Accuracy of XCS, XCSCFC, and LateralXCS after utilizing 50000 problem instances of 2-bit to 16-bit Parity problems.

The accuracy of XCSCFC gradually decrease from 100% to 34% for 2-bit Parity problems to 10-bit Parity problems, due to such a lower accuracy, XCSCFC stopped generating fit rules to match the test instances. Consequently, XCSCFC could not be applied to the remaining higher-level Parity problems due to its dependency on previously learned fitter rules. In contrast, the LateralXCS achieved 100% accuracy for all the given 2-bit to 16-bit Parity problems.

5.5.3 Hierarchical Problems

Two sets of experiments for hierarchical problems were conducted, i.e., hierarchical Mux problems and hierarchical Carry problems. In 18-bit HMux problems, the lower layer consists of the 3-bit Parity problems and the upper layer consists of the 6-bit Mux problem. Initially, both XCSCFC and LateralXCS were trained with 3-bit parity and 6-bit Mux problems. Subsequently, all the systems were tasked to learn the 18-bit HMux problem. The experimental results of the 18-bit HMux problem are shown in Fig. 5.12. Both the XCS and state-of-the-art XCSCFC were unable to efficiently learn the hidden patterns of the given problem, whereas XCS/BOA could not usefully learn the hierarchical problem. It is observed that XCS/BOA can learn these problems by using a large population size. In contrast, the LateralXCS learned the hidden patterns and was able to organize the relevant BBKs in a useful way. Consequently, the LateralXCS learned the required rules to solve the problem, achieving the performance accuracy of 100%. The number of instances ($\approx 600,000$) utilized by the LateralXCS to learn the 18-bit HMux problem is low with respect to the complexity of the problem, i.e., low sparsity and hierarchical distribution of knowledge.

Two sets of experiments for hierarchical problems are conducted, i.e. hierarchical Mux problems and hierarchical Carry problems. In 18-bit hierarchical Carry problems, the lower layer consists of the 3-bit Majority-on problem and the upper layer consists of the 3-bit Carry problem. Initially, both XCSCFC and LateralXCS were trained with 3-bit Majority-on and 3-bit Carry problems. Subsequently, all the systems were tasked to learn an 18-bit hierarchical Carry problem. All the systems learned the problem in a similar way with almost the same learning pace. The experimental results of the 18-bit hierarchical Carry problem are shown in Fig. 5.11.

In 18-bit HMux problems, the lower layer consists of the 3-bit Parity problems and the upper layer consists of the 6-bit Mux problem. Initially, both XCSCFC and LateralXCS were trained with 3-bit parity and 6-bit Mux problems. Subsequently, all the systems were tasked to learn the 18-bit

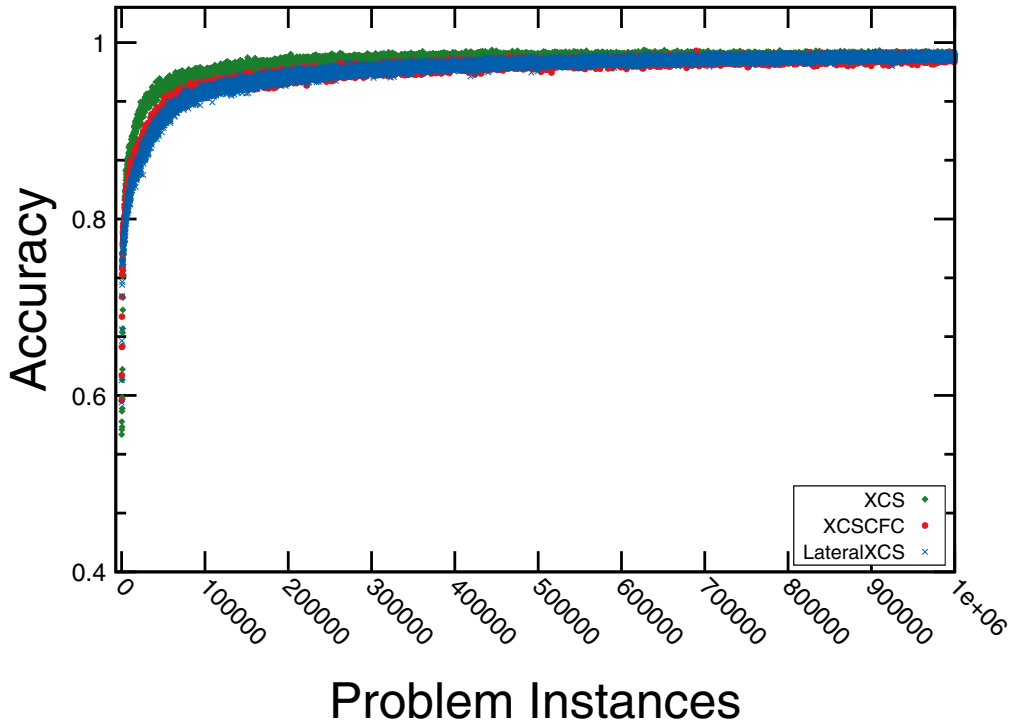


Figure 5.11: Experimental results of 18-bit hierarchical Carry problem using XCS, XCSCFC, and LateralXCS.

HMux problem. The experimental results of the 18-bit HMux problem are shown in Fig. 5.12. Both the XCS and state-of-the-art XCSCFC were unable to efficiently learn the hidden patterns of the given problem, whereas XCS/BOA could not usefully learn the hierarchical problem. It is observed that XCS/BOA can learn these problems by using a large population size. In contrast, the LateralXCS learned the hidden patterns and was able to organize the relevant BBKs in a useful way. Consequently, the LateralXCS learned the required rules to solve the problem, achieving performance accuracy of 100%. The number of instances ($\approx$ 600000) utilized by the LateralXCS to learn the 18-bit HMux problem is low with respect to the

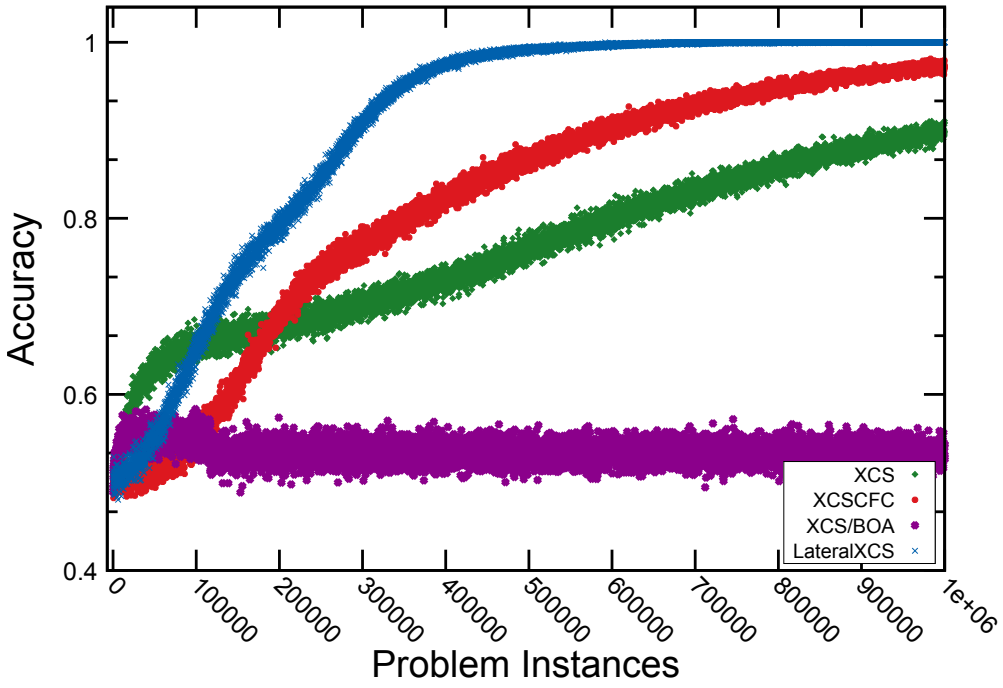


Figure 5.12: Experimental results of 18-bit hierarchical Mux problem using XCS, XCSCFC, XCS/BOA, and LateralXCS.

complexity of the problem, i.e. low sparsity and hierarchical distribution of knowledge.

The Wilcoxon signed-rank test was applied in a similar way as explained for Parity problem experiments (see Table 5.2). The P-values are less than 0.00001. The experimental results of the 18-bit HMux problem for GP-based systems are shown in Table 5.3. It shows that none of the GP-based systems achieved performance accuracy of 100%, whereas the LateralXCS achieved 100% performance accuracy by utilizing an equivalent number of problem instances.

Random Problem	Most Relevant First	Least Relevant First
2-bit Parity	2-bit Parity	2-bit Parity
3-bit Parity	3-bit Parity	3-bit Parity
6-bit Mux	6-bit Mux	6-bit Mux
3-bit MajOn	3-bit MajOn	4-bit Parity
4-bit Parity	6-bit Parity	5-bit Parity
5-bit Parity	6-bit Carry	5-bit MajOn
6-bit Parity	9-bit Parity	7-bit MajOn
5-bit MajOn	9-bit MajOn	7-bit Parity
6-bit Carry	4-bit Parity	8-bit Parity
7-bit MajOn	5-bit Parity	3-bit MajOn
7-bit Parity	5-bit MajOn	6-bit Parity
8-bit Parity	7-bit MajOn	6-bit Carry
9-bit Parity	7-bit Parity	9-bit Parity
9-bit MajOn	8-bit Parity	9-bit MajOn

Figure 5.13: Sequence of learned knowledge steps. Blue: required sub-problems. Purple: most relevant sub-problems. Yellow: least relevant sub-problems.

5.6 Experimental Analysis

This section provides an analysis of the experiments which were conducted to evaluate the performance of the novel lateralized system. It was achieved by computing the overhead of irrelevant problems and interpreting the decisions made by the novel system.

5.6.1 Overhead of Irrelevant Sub-problems to LateralXCS

The lateralized system can automatically (without human-in-the-loop) identify the relevant required building blocks of knowledge from the learned knowledge pool. Three sets of experiments were conducted to fairly compute the overhead (extra) problem instances utilized by the LateralXCS to

identify these relevant BBKs, i.e. (i) random order of presenting problems, (ii) most relevant problems first, and (iii) least relevant problems first. In all these experiments, the LateralXCS was tasked to identify the relevant BBKs that are required for learning the 18-bit HMux problem. The necessary sub-problems were learnt in order the most relevant of their sub-problems, e.g. 3-bit Parity needs the 2-bit Parity.

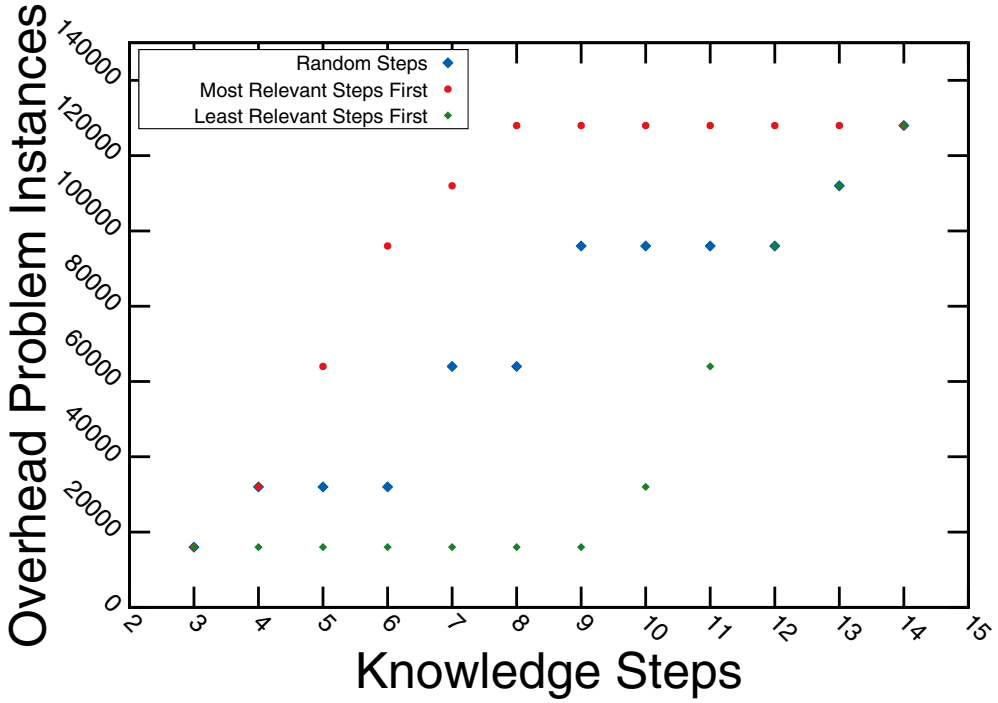


Figure 5.14: Overhead problem instances utilized by LateralXCS during the learning of 18-bit hierarchical Mux problems.

First, the LateralXCS was trained with a diverse set of problems in a random sequence of knowledge steps, i.e. the system may or may not consider the newly learned problem a relevant sub-problem for the learning of 18-bit HMux problem. Second, the LateralXCS was trained with a

problem set in which the most relevant problems were presented before irrelevant problems. Third, the LateralXCS was trained with a problem set in which the irrelevant problems were presented before the relevant problems at each scale. The learning sequence of problems is presented in Fig.5.13.

The overhead (extra) problem instances utilized by the LateralXCS to identify the relevant BBKs during the learning of 18-bit HMux problems are presented in Fig.5.14. The overhead cost depends on the relevancy of the learned problem concerning the given problem, e.g. both 6-bit Mux and 6-bit Parity problems are considered as potentially relevant sub-problems for learning the 18-bit HMux problem. After utilizing preliminary problem instances, the system identified 6-bit Mux as relevant and 6-bit Parity as irrelevant sub-problems, whereas, LateralXCS considers 5-bit MajorityOn and 5-bit Parity as irrelevant sub-problems without utilizing any problem instances during the learning of the 18-bit HMux problem. The number of combinations tried by the LateralXCS with the addition of a new problem along with the overhead cost is presented in the Table 5.4. At worst 128000 problem instances are needed to find the relevant BBKs for 18-bit HMux when the system has learned 14 diverse knowledge steps that form 16 candidate combinations.

5.6.2 Interpretation of Decisions

The decision-making process of the LateralXCS is interpretable. Close observation of the rules generated by the LateralXCS reveals that the system successfully identified and efficiently utilized the relevant BBKs from the pool of learned knowledge.

The LateralXCS efficiently learned the 7-bit Parity problem by utilizing a small number of problem instances as compared to other EML systems, see Fig. 5.8b. The learned concept of 7-bit Parity problem consists of 203 rules, 69 condition-CFs, and 116 action-CFs. An example rule from the learned concept of the 7-bit Parity problem is shown in Fig. 5.15. This is

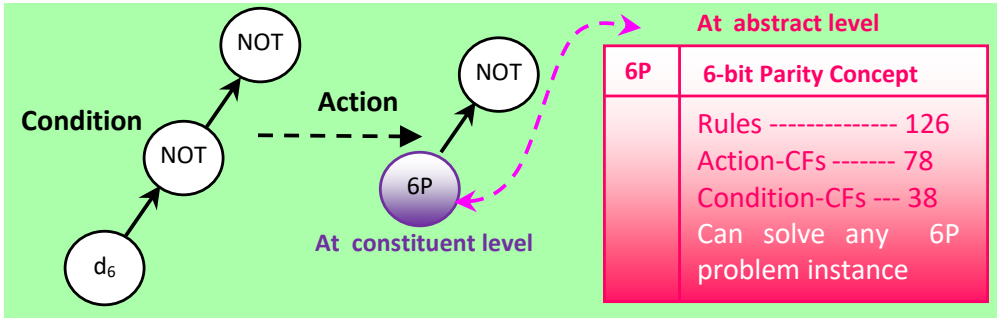


Figure 5.15: An example rule (R1) from 7-bit Parity problem (d_6 represents condition bit #6 and 6P represents 6-bit Parity concept). Numerosity 25, Experience: 121465, Accuracy 1, Prediction Error: 0, Prediction: 1000, Fitness: 0.095, Specificness 1

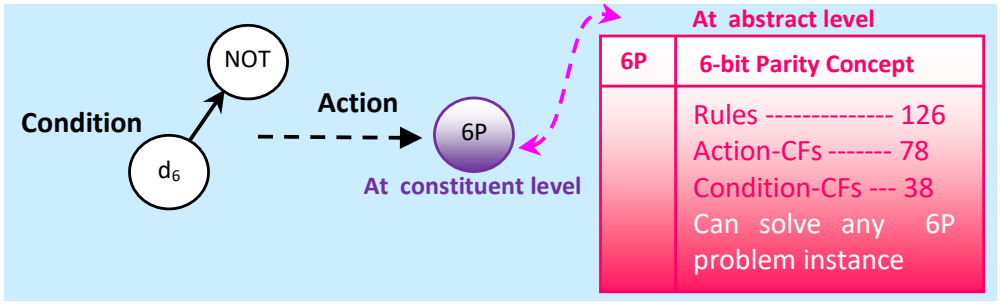


Figure 5.16: An example rule (R2) from 7-bit Parity problem (d_6 represents condition bit #6 and 6P represents 6-bit Parity concept). Numerosity 6, Experience: 111038, Accuracy 1, Prediction Error: 0, Prediction: 1000, Fitness: 0.023, Specificness 1

Table 5.4: Combinations and Overhead of learned knowledge steps

Knowledge Steps	Random Problems	Combinations	Overhead
1	2-bit Parity	0	0
2	3-bit Parity	0	0
3	6-bit Mux	2	16000
4	3-bit MajOn	4	32000
5	4-bit Parity	4	32000
6	5-bit Parity	4	32000
7	6-bit Parity	8	64000
8	5-bit MajOn	8	64000
9	6-bit Carry	12	96000
10	7-bit MajOn	12	96000
11	7-bit Parity	12	96000
12	8-bit Parity	12	96000
13	9-bit Parity	14	112000
14	9-bit MajOn	16	128000

the most experienced (iterations 121465) and accurate rule with high numerosity (25) and low specificity (1). The only CF in the condition has three elements, i.e. condition bit #6, operator NOT, operator NOT. The condition CF is "not of not of condition bit #6", i.e. $NOT(NOT(D_6))$. The CF in the action part has two elements, i.e. 6-bit Parity concept and operator NOT. The action CF is "NOT of 6-bit Parity". A rule matches the given problem instance if all of the CFs in its condition generate value '1'. According to this principle, the above-mentioned rule matches all the problem instances that have value '1' at the 7th bit (i.e. D_6). The resultant response of the system is the opposite of the 6-bit Parity concept.

Another important rule from the 7-bit Parity problem is shown in Fig. 5.16. This is also an experienced (iteration 111038) and accurate rule with numerosity 6, and specificity 1. The only CF in the condition part has two

D_0	D_1	D_2	D_3	D_4	D_5	D_6	P6	P7	Rule1 if ~(~D_6) -> P7=~P6		Rule2 If ~D_6 -> P7 = P6	
									Applicable	Output	Applicable	Output
0	0	0	0	0	0	0	0	0	×	-	✓	0
0	0	0	0	0	0	1	0	1	✓	1	×	-
0	0	0	0	0	1	0	1	1	×	-	✓	1
0	0	0	0	0	1	1	1	0	✓	0	×	-
0	0	0	0	1	0	0	1	1	×	-	✓	1
-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-
1	1	1	1	1	1	0	0	0	×	-	✓	0
1	1	1	1	1	1	1	0	1	✓	1	×	-

Figure 5.17: Logical interpretation of two experienced and accurate 7-bit Parity problem rules (see rule R1 Fig.5.15 and rule R2 Fig.5.16). Here D represents the condition bits and P represents Parity concepts.

elements, i.e. condition bit #6 and NOT operator. This CF is "not of condition bit #6", i.e. $NOT(D_6)$. The CF in the action part has only one element that is "6-bit Parity". This rule matches all the problem instances that have value '0' at the 7th bit. The resultant response of the system is the same as that of the 6-bit Parity concept. These two rules cover all the instances of the 7-bit Parity problem. By evolving these compact rules, the Lateralized system effectively converted the 7-bit Parity problem into a two-bit Parity problem, as shown in Fig. 5.17. Consequently, the LateralXCS efficiently learned the 7-bit Parity problem by utilizing a very small number of problem instances. Therefore, it is plausible that LateralXCS can solve any scale n-bit Parity problem given successively scaled problems.

Hierarchical Boolean problems are challenging due to an additional layer of complexity, low sparsity, and hierarchical distribution of knowledge. It is necessary to apply heterogeneous BBKs to resolve a hierarchical Mux problem. The learned concept of 18-bit HMux problem consists of 3202 rules, 2347 condition-CFs, and 679 action-CFs. An example rule from

Rule	
Condition	D_3, D_4, D_5, D_{10}, D_{17}, CF_{1218D}
Action	CF_{08D36}
Features	
Numerosity 12, Accuracy 1, Fitness: 0.181, Prediction Error: 0, Prediction: 1000, Experience: 7700, Specificness 6	

Figure 5.18: An example rule from 18-bit hierarchical Mux problem.

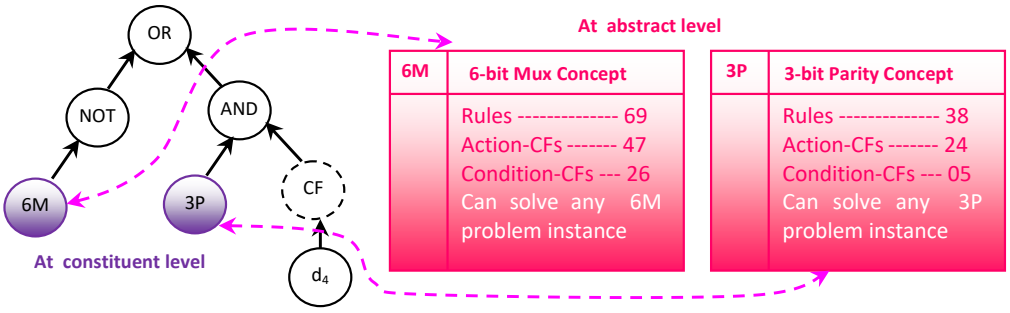


Figure 5.19: Tree representation for CF Id: 1218D. 6M, 3P, and d_4 represent 6-bit Mux concept, 3-bit Parity concept, and condition bit #4, respectively

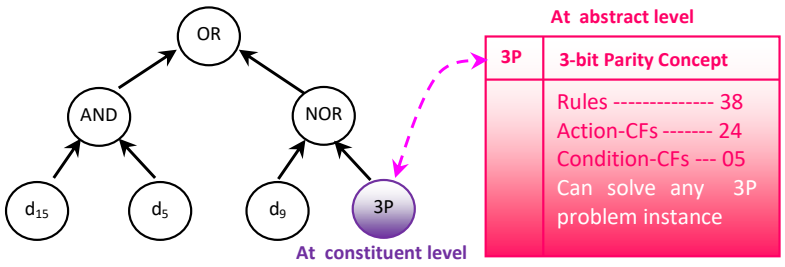


Figure 5.20: Tree representation for action CF Id: 08D36. 3P, and d_5, d_9, d_{15} represent 3-bit Parity concept, and condition bits #5, #9, #15, respectively

18-bit HMux is shown in the Fig. 5.18. This is an experienced (iterations 7700) and accurate rule with numerosity 12, and specificity 6. There are 6 CFs in the condition of the rule. The tree representations for condition CF (Id:$1218D$) and action CF (Id:$08D36$) are shown in Fig. 5.19 and Fig. 5.20, respectively. A close observation of this rule reveals that, along with other BBKs, it has 6-bit Mux and 3-bit Parity concepts in its condition and 3-bit Parity in action. These concepts act at the constituent level when they are inside the CFs to resolve hidden problem instances. In contrast, they act at the holistic level when tasked to independently resolve any respective problem instance of 6-bit Mux problem or 3-bit Parity problem. This shows that the system has the ability to identify and utilize the critical BBKs, from the learned knowledge pool, to resolve the hidden layers of the problem.

5.7 Discussion

This work is designed to solve problems where a presented instance of a problem can be constructed from instances of sub-problems. It can solve base problems, which then act as future BBKs, albeit this entails additional computational overhead. The system considers (addresses) the problem at two different viewpoints (constituent-level and holistic-level) simulta-

neously. As the problem scales and the knowledge pool grows, there will be more and more candidate constituent sub-problems, which does slow down the behavior. However, the system has the ability to identify (without human-in-the-loop) the relevant sub-problems for a large problem that has constituents parts. Consequently, it scales much more quickly than systems that do not consider sub-problems, as demonstrated by resolving hierarchical Mux problems.

Boolean problems can exhibit heterogeneity and epistasis with identifiable components of a problem that are transferable to other problems. These qualities make Boolean problems an ideal test set to obtain a proof-of-concept and show the effectiveness of the lateralized approach. The novel system was not much better than existing algorithms for solving simple problems. The lateralized system with modular learning was anticipated to be suited to hierarchical problems, where it performed very well. But it is not over-fitted or only suited to such engineered problems as demonstrated by the Parity results where the problem is not observably (or constructed to be) hierarchical and yet the lateralised system still outperformed state-of-the-art existing systems. The majority of the Boolean problems are single-step and may be considered as supervised learning. However, the experiments test the novel system using reinforcement learning to support the plausibility of applying the lateralized approach to multi-step problems, e.g. path planning.

5.8 Chapter Summary

The main objective in this chapter was to develop a novel system to obtain the proof-of-concept of the lateralized approach. The novel system successfully applied lateralization and modular learning at different levels of abstraction to resolve complex Boolean problems. Considering the same problem at different levels of abstraction (i.e. constituent level and holistic level) enables the novel system to reframe a complex problem as

a simple problem and efficiently resolve it. For example, the novel system addressed the *n*-bit Parity problem as a two-bit problem by utilizing the learned concept of the (*n*-1)-bit Parity problem and the one additional condition bit #n. Moreover, the experimental results demonstrated that the lateralized system has the ability to identify and utilize the relevant BBKs to efficiently learn the distribution of knowledge in hierarchical Boolean problems.

The developed lateralized system successfully provided the proof-of-concept but it does not show the effectiveness of the lateralized approach in resolving real-world problems. Hence, the next chapter aims at developing lateralized system for computer vision problems.

6

Lateralized Learning for Robustness Against Adversarial Attacks in a Visual Classification System

Deep learning is an important field of machine learning. It is playing a critical role in a variety of applications ranging from self-driving cars to security and surveillance. However, deep networks have deep flaws. For example, they are highly vulnerable to adversarial attacks. One reason may be the homogeneous nature of their knowledge representation, which allows a single disruptive pattern to cause miss-classification. The biological lateral framework allows heterogeneous, modular learning at different levels of abstraction, enabling different representations of the same object.

A lateralized framework for artificial intelligence systems has been created, which has been verified on 'toy', albeit complex,

Boolean problems. This chapter verifies the effectiveness of the lateralized approach on real-world computer vision tasks that alternative state-of-the-art techniques have struggled to address fully. Two novel lateralized systems are developed to show that the lateralized approach can be scaled and not limited to learning classifier systems. The first lateralized system is developed to address binary-class image classification tasks, whereas, the second lateralized system is an improved version of the first implementation to address multi-class (200 *classes) image classification tasks.*

The results of image classification tasks show that the lateralized systems efficiently learn hierarchical distributions of knowledge, demonstrating performance that is similar to (or better than) other state-of-the-art deep systems as it reasons using multiple representations. Crucially, the first system outperformed all the state-of-the-art deep models for the classification of normal and adversarial images by $0.43\% - 2.56\%$ *and* $2.15\% - 25.84\%$*, respectively; whereas, the second system outperformed all the state-of-the-art deep models for the classification of normal and adversarial images by* $19.05\% - 41.02\%$ *and* $1.36\% - 49.22\%$*, respectively. Lateralisation enabled the novel systems to exhibit robustness beyond previous work, this advocates for the creation of data sets that facilitate lateralization, i.e. have components of objects and the objects themselves to be learned specifically or in an end-to-end manner.*

6.1 Introduction

A lateralized framework for artificial intelligence systems has been created. The proof-of-concept was successfully obtained by adapting this

framework to create lateralized AI system for complex Boolean problems. The next step is to show that the underlying lateralized framework can be adapted to create lateralized AI systems for real-world computer vision problems that include uncertainty, noise, and irrelevant and redundant data. Such a real-world problem could show the advantages of the novel lateralized approach against state-of-the-art approaches.

Deep learning (DL) is a methodology of progressively extracting higher-level features from the raw input by applying multiple layers of artificial neural networks [287]. A broad range of DL-based systems has been developed. These systems are playing critical roles in a variety of applications ranging from self-driving cars to security and surveillance [2, 3]. However, deep networks are highly vulnerable to adversarial attacks [48, 52][1]. Even a small (imperceptible to a human) perturbation to an image can fool many deep networks resulting in the wrong prediction made with high confidence [53, 54].

One reason for poor robustness against adversarial attacks is the reliance on homogeneous knowledge representation. Thus, a single, targeted pattern can disrupt classification performance. Homogeneous systems work well when the relationship between features and action (target) is linearly separable. Moreover, deep networks encourage linear behavior for efficient learning. However, most adversarial attacks exploit this hallmark to fool deep networks [52, 55, 56].

Existing approaches provide only a partial solution to this problem. Adversarial training techniques reduce over-fitting by regularizing the deep networks, which improves robustness against adversarial attacks. However, these techniques are considered non-adaptive due to their dependency on already existing adversarial data. Moreover, the need for adversarial training results in an increase in the training data size and expensive network architecture [55, 290]. Furthermore, it has been reported

[1] An adversarial attack is a technique that attempts to fool AI models by generating deceptive input images [53, 54] (see Chapter 2).

that adversarial-trained networks can again be fooled by creating novel adversarial patterns [291].

Compression-based techniques are another approach that has been investigated as a defense against adversarial attacks. The results suggest that compression alone is inadequate to provide an effective defense [292, 293, 294]. Moreover, it is hard to find appropriate compression for a data set. Smaller compressions are unable to handle adversarial perturbations, whereas, larger compressions decrease the classification accuracy of clean images. Modification of the deep networks is yet another area where efforts have been made to improve adversarial robustness. However, it has been reported that the majority of these methods are either unable to provide an effective defense or too complex so require a very large number of training instances [295, 296, 297].

6.1.1 Chapter Objectives

The main objective reported in this chapter is to create two novel lateralized systems for computer vision problems, by adapting the developed lateralized framework. These novel systems will have the ability to provide robust solutions against adversarial attacks by applying lateralization and modular learning at different levels of abstraction. To achieve this objective, the following sub-objectives are set:

(i) Create a lateralized system that can simultaneously process a single visual input at different levels of abstraction, i.e. at the constituent level and the holistic level.

(ii) Represent knowledge in a heterogeneous manner. Different knowledge components are utilized or re-utilized at different levels of abstraction, i.e. a holistic knowledge component at one level can be re-utilized as a constituent knowledge component at a higher level of abstraction. Different system components coordinate to reuse the learned knowledge at different levels of abstraction.

(iii) Enable communication between different system components through inhibition and excitation signals to efficiently resolve the problem and avoid extraneous computations.

6.1.2 Chapter Organisation

The remainder of this chapter is organized as follows. Section 6.2 presents a lateralized system for binary-class image classification tasks. It shows how a homogeneous system can be split into a lateralized system for visual classification tasks. It presents the critical components and architecture of the novel system based on the lateralized framework created in Chapter 4.2. An example task of classifying cats vs dogs, in photographic images, is used to show the potential of the lateralized approach. The experimental setup, data sets, and data preparation are explained in Section 6.2.2. Section 6.2.3 presents the work done to evaluate the effectiveness of the developed system. The robustness of the developed system against adversarial attacks and interpretation of decisions are presented in Section 6.2.4. Section 6.3 presents a lateralized system for multi-class image classification tasks. It shows how the developed lateralized system can be improved to handle multi-class image classification tasks. An example task of classifying birds (200 classes), in photographic images, is used to show the scalability of the lateralized approach. Section 6.3.2 describes the experimental setup, data set, and data preparation. The work done to evaluate the effectiveness of the developed system is presented in Section 6.3.3. Section 6.4 highlights the strengths, limitations, and drawbacks of the lateralized approach for CV tasks. Finally, the chapter summary is presented in Section 6.5.

6.2 Lateralized System for Binary-Class Image Classification

6.2.1 Lateralized System

The overall classification process of the lateralized system can be divided into two main phases, i.e. the context phase and the attention phase. The context phase handles simple images, whereas, the attention phase address noisy and corrupt images. The novel lateralized system is created by adapting the lateralized framework such that both phases (context phase and attention phase) implement left-half (constituent level) and right-half (holistic level) functionality. The implementation of this functionality for both phases enables lateralization at multiple levels of abstraction. Moreover, excite and inhibit signals are generated by the context phase that assists the novel system to efficiently solve the given problem. Finally, the feedback from both phases is analyzed to resolve the problem. A schematic depiction of the strategies developed for the novel system is shown in Fig. 6.1.

Context Phase

The context phase consists of multiple deep networks. Some of these deep networks are used to generate constituent level predictions, i.e. about individual parts of the objects to be identified. The remaining deep networks are used to generate a holistic level prediction, i.e. the big picture. This strategy not only allows the novel system to consider the given task at the constituent level and holistic level simultaneously but also enables it to move away from end-to-end learning so as to generate important feature groups. Here cat vs dog classification is used as an illustrative example. The prediction consists of the probability that the image (or prat) belongs to each candidate class. Subsequently, these prediction values are used to

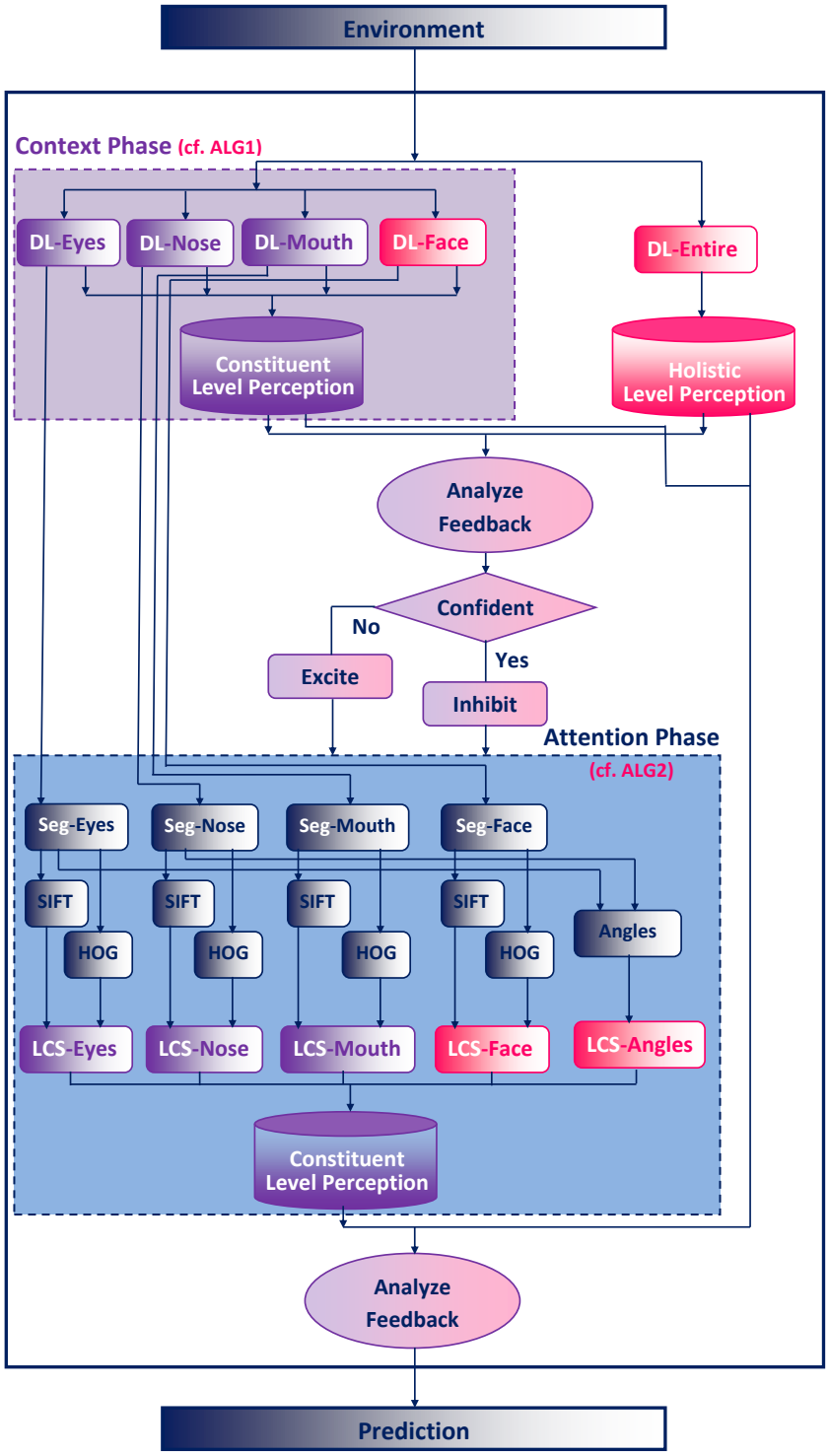


Figure 6.1: A schematic depiction of the strategies developed to achieve cognitive inspired functionality in the binary-class lateralized system. (color key: constituent, holistic, and mix knowledge proceedings are represented by purple-white, pink-white, and light purple-pink gradients, respectively)

generate two types of perceptions, i.e constituent level perception (CLP) and holistic level perception (HLP). The context phase consists of deep networks, i.e. five in this case. Three of these deep networks are used to generate predictions about the constituent eyes, nose, and mouth, whereas two deep networks are used to generate configural predictions about the face and overall image. Here, the prediction contains two values, i.e. the likelihood the image belongs to class '*one*', and the likelihood it belongs to class '*two*'.

Initially, the absolute difference between the probabilities of class '*one*' and class '*two*' is computed for a part (or whole image). If the probability of class '*one*' is greater than the probability of class '*two*', the resultant difference value is multiplied by '-1' (to change the sign), otherwise, it is left as is. Subsequently, the resultant value is divided by 100 to normalize it between -1 and 1. Finally, this final value is considered as a prediction vote for the part (or the whole). This process is repeated to compute the prediction vote for each part (eyes, nose, mouth), and big picture (face, whole image). The overall CLP vote is computed by adding the prediction votes for eyes, nose, mouth, and face. In contrast, the HLP vote consists of the whole image prediction vote only. It is important to note here that if a deep network is unable to predict any part (due to noise or adversarial attack), the default prediction value of 0 is used. The CLP and HLP information are shared with the system.

The system analyzes the feedback from CLP and HLP. If CLP and HLP support each other, i.e. both the CLP vote and HLP vote have the same sign, the system is confident to classify the given image correctly. Subsequently, the system adds both votes and generates an inhibit signal to the attention phase so that it can cease working. Finally, the system makes a final prediction as class '*one*' or class '*two*' if the final votes are positive or negative, respectively. However, if CLP and HLP do not support each other (i.e. CLP vote and HLP vote have different signs) or CLP is confused (i.e. CLP vote is 0), the system cannot confidently classify the given im-

age correctly. Subsequently, the system generates an excite signal to the attention phase and waits for the reply. The pseudo-code of the strategy adopted by the context phase to generate a prediction is given in Algorithm 7.

Attention Phase

The Attention phase consists of multiple learning classifier systems (LCSs). Some of these LCSs are used to generate constituent level predictions, i.e. about individual parts, whereas the remaining LCSs are used to generate a holistic level prediction, i.e. big picture. For this work, a total of five LCSs are used to generate predictions for the attention phase. Three of these systems are used to generate CLPs about eyes, nose, and mouth, whereas one LCS is used to generate an HLP (big picture) about the face. Moreover, another LCS is used to generate an HLP (big picture) based on the angles of a triangle consisting of two eyes and nose vertices. It is postulated that the eyes of a cat are usually close to the nose, whereas the eyes of a dog are relatively far from the nose. Thus, the imaginary triangle consisting of the eyes and nose is a feature that may distinguish cats from dogs. This feature is to be used in the LCSs to generate a holistic level prediction about the given problem instance. The attention phase starts processing in parallel with the context phase. However, it stops immediately if it receives an inhibit signal from the context phase. The attention phase utilizes the deep models of the context phase to generate predictions about the bounding box (bbox) of eyes, nose, mouth, and face. Subsequently, it segments each part according to the bbox values.

Different types of features are computed for the segmented images. These features are selected based on the nature of the classification problem, effectiveness of the classification, and robustness against noisy data. For this work, the system computes three variants of scale-invariant feature transform (SIFT) features and three variants of histogram of oriented gradients (HOG) features for the segmented images (see Chapter 2). These

Algorithm 7: Strategy adopted by the context phase to generate a prediction (cf. Fig. 6.1).

Data: The data set and problem configurations
Result: Generate a prediction, inhibit or excite signal for attention phase

Initialize the global variable and parameter settings;
Compute Prediction Vote(); *% Compute prediction vote for a part (eyes, nose, mouth, face) or whole image.*
    Generate Prediction From Deep Model(); *% Generate prediction from the relevant deep model. The prediction returns two values, i.e. the probability to be a class 'one', and the probability to be a class 'two'.*
    Compute Absolute Difference(); *% Absolute difference between the probabilities of class 'one' and class 'two'.*
    if *Probability class 'one'* $>$ *Probability class 'two'* **then**
        Multiply Absolute Difference by $'-1'$;
    else
        Multiply Absolute Difference by $'+1'$;
    end
    Normalize Difference(); *% Divide by* 100 *to normalize between* -1 *and* 1.
Compute CLP Vote (); *% Add prediction votes of eyes, nose, mouth, and face to compute constituent level perception (CLP). It is important to note here that the vote for the missed (e.g. deep network is unable to identify due to noise or adversarial attack) part is* 0 *by-default.*
Compute HLP Vote (); *% Assign the whole image prediction votes as a holistic level perception (HLP).*
Analyse Feedback (); *%Analyze the feedback from CLP and HLP.*

```
14  if CLP vote sign equal to HLP vote sign then
15      % System is confident to classify the given image correctly.
16      Generate Inhibit Signal();      % generates an inhibit signal to
          the attention phase so that it stops working.
17      Final Votes = CLP vote + HLP vote;
18          if Final Votes < 0 then
19              prediction class 'one';
20          else
21              prediction class 'two';
22  else
23      Generate Excite Signal();       % generates an excite signal to
          the attention phase and wait for the reply.
```

variants are selected to add diversity. Note that the system is flexible enough to adopt any practical number. These features form the input instances (environment) for the LCS. Three LCSs models are used to generate constituent level predictions for eyes, nose, and mouth. The fourth LCS model is used to generate a prediction for the face (big picture). Moreover, an imaginary triangle is created by considering the center-points of bound boxes of two eyes and nose as vertices. The three angles of this triangle (big picture) are used as input instances (environment) for another LCS to generate a holistic level prediction. Instead of generating absolute prediction from the LCSs, i.e. class '*one*' or class '*two*', the votes for each class based on the prediction array is returned as a prediction probability. These returned votes are the prediction probability for class '*one*' or class '*two*' for each problem instance. Moreover, these prediction probabilities for class '*one*' are multiplied by -1, whereas the predictions for class '*two*' are left as is. Subsequently, the final CLP votes for LCSs are computed by adding the prediction probabilities of each part (eyes, nose, mouth), face, and angles. This final LCSs based CLP vote is returned to the system.

Algorithm 8: Strategy adopted by the attention phase to generate a prediction (cf. Fig. 6.1).

Data: The data set and problem configurations
Result: Generate final prediction

Initialize the global variable and parameter settings;
Check Inhibit Signal(); *% Check inhibit signal form context phase and stop immediately.*
Get BBox From Deep Models(); *% Get bbox prediction for eyes, nose, mouth, and face.*
Crop Img(); *% Segment each part based on the bbox values.*
Compute SIFT(); *% Compute three variants of SIFT features for each segmented image.*
Compute HOG(); *% Compute three variants of HOG features for each segmented image.*
Compute Angle(); *% Compute angles of an imaginary triangle consisting of two eyes and nose. The center of bbox for eyes and nose are taken as vertices.*
Get LCSs Prediction(); *% Get LCSs predictions for eyes, nose, mouth, face, and angle*
Compute LCS-CLP Vote (); *% Add LCSs prediction votes of eyes, nose, mouth, face, and to compute LCSs based CLP. It is important to note here that the vote for any missed part is* 0 *by-default.*
Share LCSs-CLP();
Analyse Feedback (); *%Analyze the feedback from LCSs-CLP, Context-CLP and Context-HLP.*
    if *Final Votes* < 0 **then**
        prediction class '*one*';
    else
        prediction class '*two*';
    end

Subsequently, the system computes the prediction votes for the image by adding votes from LCS-based CLP, context CLP, and context HLP. The lateralized system finally decides the problem instance as class '*one*' if the image prediction votes are greater than 0, otherwise, as class '*two*'. The pseudo-code of the strategy adopted by the attention phase to generate a prediction is given in Algorithm 8.

6.2.2 Experimental Design

This work seeks to show the robustness of a lateralized system against adversarial attacks. It can be achieved by conducting classification experiments on images data sets. The data sets need to have constituents level as well as holistic level ground truth information as it is required for the training of the constituent level prediction models and holistic level prediction models of the novel system. For this work, the experiments are conducted for cat vs dog classification.

Data Sets

This work uses publicly available cat and dog data sets that have been used by the research community. These data sets have constituents and holistic levels labeled data. The cat data set is taken from Kaggle competition [264]. It includes more than 9000 cat images along with ground truth files (9 points annotation of the head). The dog data set is taken from dlib (C++ library for ML) [265], which is a modified copy (modified missed annotations and loose bboxs) of the data used by Liu et al. [266]. It includes more than 8000 dog images along with the ground truth information (8 points annotation of the head).

Data Preparation: The lateralized system needs the information (bbox) about the eyes, nose, mouth, and face of an image to train deep models and LCSs. The ground truth files of the chosen data sets have annotations related to these parts but not the bboxs. The system utilizes these annota-

tions to generate the required information. For this purpose, two separate routines (one for cat and one for dog images) are developed that take the annotation file as input and generate corresponding bboxs for eyes, nose, mouth, and face. Moreover, these routines have the ability to handle rotated images and generate bboxs accordingly. Further explanation (logical description and algorithm) of these routines is not presented as independent of the lateralization.

Experimental Setup

The learning strategy for the context phase is developed by utilizing multiple deep models depending on the nature and complexity of the problem, e.g. five deep models are used here. These models can be based on any state-of-the-art pre-trained deep network. The models used here are based on pre-trained 50 layers residual networks (ResNet50 model), which are well-recognized and widely used deep networks [319]. The loss function, mean absolute error (MAE), is used to train the deep networks, whereas, the state-of-the-art Adam optimization algorithm (stochastic gradient descent optimizer) is used to update the network weights [320]. Moreover, the deep models are trained for 300 epochs such that the length of each epoch is 1000. It is important to note that ResNet50 model is also used to generate adversarial images to ensure that the adversarial attacks have the same underlying model. The learning strategy for the attention phase is developed by utilizing multiple (e.g. five here) accuracy based supervised LCSs, i.e. sUpervised Classifier Systems (UCSs) [246]. The configuration settings for LCSs are the same as used by the majority of the research community [252]. For this work I use following LCSs parameter values: Type of crossover ="two point"; Crossover probability $\chi = 0.8$; Genetic algorithm's threshold $\theta_{ga} = 25$; Subsumption threshold $\theta_{sub} = 20$; Minimum accuracy threshold $\epsilon_0 = 0.99$; Deletion threshold $\theta_{del} = 20$; Experience threshold $\theta_{exp} = 10$; Probability of mutating a linguistic term $\mu = 0.04$; Exponent of the fitness function $\upsilon = 10$; Tournament selection parame-

ter $\tau = 0.4$; Learning rate $\beta = 0.2$; Fraction of population mean fitness $\delta = 0.1$; Fall of rate in fitness function $\alpha = 0.1$; Real representation parameter $r_0 = 0.7$; Nominal representation parameter $= 0.4$; Fitness reduction $= 0.1$; Mutation parameter $m_0 = 0.5$; Population size $= 8000$; Number of iterations $= 500000$.

6.2.3 Experiments

For all the experiments, the combined data sets are randomly divided into 80% training and 20% testing images. The lateralized system is trained once using the training images. The adversarial attacks are applied to the testing images only. The effectiveness of the novel system is evaluated by utilizing the original testing images and then the adversarial images. During the evaluation process, an image is presented to the lateralized system. Initially, both the context and attention phases start processing simultaneously. The context phase computes CLP and HLP votes based on the predictions from deep models, whereas, the attention phase segments the images by utilizing the bbox information received from the deep models. Subsequently, it computes three variants of SIFT features by utilizing the following parameter values: patchSize: 64, 128, and 256; maxBinValue: 0.2, numOrientationBins: 8, and numSpatialBins: 2. Similarly, it computes three variants of HOG features by utilizing the following parameter values: image size: $64, 126$, and 256; pixels per cell: $32, 64$, and 128. All these features form the input environment instance for the LCSs. The attention phase computes CLP votes by utilizing the prediction probabilities of LCSs. Finally, the lateralized system receives all the CLPs and HLP votes and makes the final prediction accordingly.

Two variants of fast gradient sign method (FGSM) attacks are applied on the testing images to generate adversarial images, i.e. medium-level attack FGSM-M (epsilon (max norm) value 50) and strong attack FGSM-S (epsilon 150) [321]. Moreover, two variant of Iterative attacks are applied on the testing images to generate adversarial images, i.e. medium-level

Table 6.1: Classification Accuracy
Highest Accuracy is in bold.

	VGG	SqueezeNet	AlexNet	ResNet	LateralSys
OrigImgs	99.37	98.35	97.24	98.64	**99.80**
FGSM-M	94.68	85.91	90.25	89.30	**98.34**
FGSM-S	70.38	55.22	63.00	77.61	**81.06**
Itr-M	97.44	95.81	94.72	88.24	**99.59**
Itr-S	96.10	93.91	93.66	83.41	**99.30**

attack Itr-M (epsilon 18, alpha 1, and iterations 10), and strong attack Itr-S (epsilon 50, alpha 1, and iterations 10) [321]. The experimental results of the novel system are compared with the results generated from four state-of-the-art deep models, i.e. ResNet [319], AlexNet [212], VGG [322], and SqueezeNet [323].

The experimental results show that the lateralized system successfully exhibits robustness against all (four) adversarial attacks, see Table 6.1. Unexpectedly, the classification accuracy of the novel system is better than all other state-of-the-art deep models for the original test images (see Section 6.2.4). Moreover, it is evident from the experimental results that the lateralized system exhibits strong robustness against three types of adversarial attacks (FGSM-M, Itr-M, and Itr-S). Against all these attacks the classification accuracy of the novel system decreases by less than 1%, whereas, the accuracy of all other systems decreases between 5% to 26%. The novel system could not completely resist the FGSM-S attack and shows a classification accuracy of 81.06%. But all other deep models also perform worse. It is quite understandable because FGSM-S is a very strong adversarial attack that destroys the image contents badly. Despite this disruption, the classification accuracy of the novel system is better than all other models against FGSM-S adversarial images.

6.2.4 Experimental Analysis

The decision-making process of the novel system is interpretable. The role played by different system components during the decision-making process is explained by utilizing eight examples, four from cat images and four from dog images. Those examples are selected where the lateralized system's components are at odds with each other. However, the final decision of the lateralized system is correct. The decision-making process for these examples is explained in the following three cases.

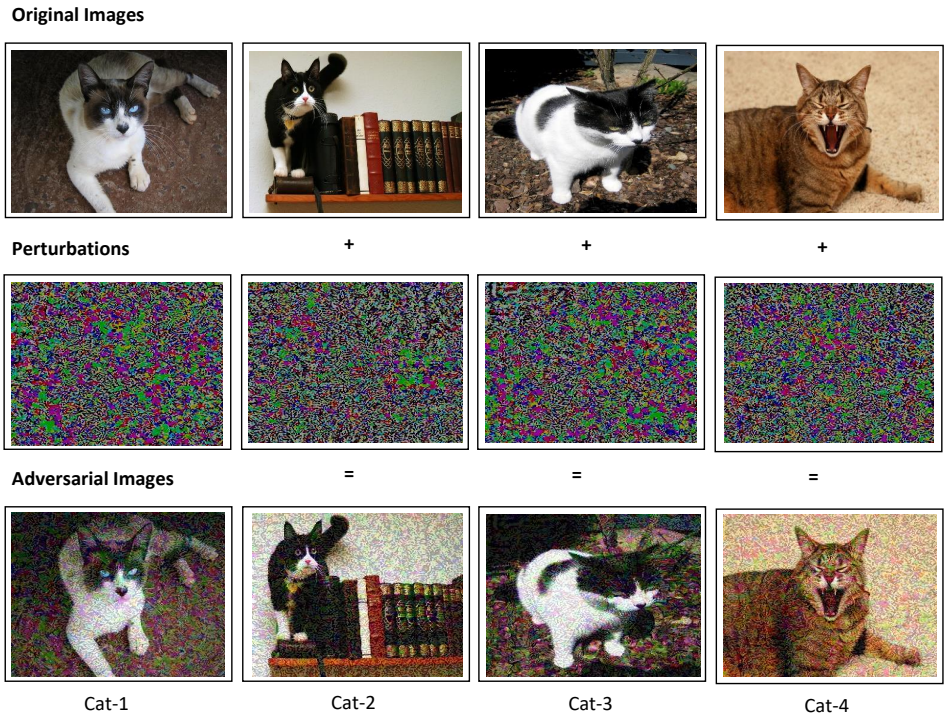


Figure 6.2: Example cat images. Original images are in the first row. The relevant perturbations are in the second row. The resultant adversarial images are in the third row.

Case-1: In this scenario, the deep model makes a correct HLP for the whole image with low confidence, whereas, the component deep mod-

els make a confused CLP due to opposite component predictions made with equal confidence. However, the LCSs model makes a correct prediction with high confidence which aligns with the holistic level prediction. Consequently, the lateralized system makes the correct decision with high confidence. For example, during the classification process of Cat-1, the holistic level deep model predicts that it has a 33.05% resemblance to a cat, whereas, the constituent level deep models predict that it is 99.00% cat mouth, and 99.00% dog mouth. Consequently, the CLP does not provide any useful input. However, the LCSs model predicts that it is 87.61% cat mouth, and 12.39% dog mouth. This prediction aligns with the holistic level prediction and empowers the novel system to make a correct cat prediction with high confidence, see Fig. 6.2 and Fig. 6.4. Similarly, for Cat-3 classification, the holistic level deep model predicts that it is a 46.29% cat. Whereas, the constituent level deep models predict that it is 100.00% cat nose, 100.00% dog nose, 99.00% cat mouth, and 99.00% dog mouth. Consequently, the CLP does not make any useful input. However, the LCSs model predicts that it is 78.67% cat nose, and 21.33% dog nose, 99.69% cat mouth, and 0.31% dog mouth. This prediction aligns with the holistic level prediction and empowers the novel system to make a correct cat prediction with high confidence, see Fig. 6.2 and Fig. 6.4.

Case-2: In this scenario, the deep model makes the wrong HLP for the whole image, whereas, the component deep models, as well as the LCSs model, make correct predictions for different components. Consequently, the lateralized system makes a correct final prediction. For example, during the classification process of Cat-4, the holistic level deep model predicts that it has a 1.06% resemblance to a dog, whereas, the constituent level deep models predict that it is 100.00% cat nose, and 99.00% cat mouth. Moreover, the LCSs model predicts that it is 96.02% cat nose, and 3.98% dog nose, 94.35% cat mouth, and 5.65% dog mouth. This prediction aligns with the component models predictions and empowers the novel system to make a correct cat prediction with high confidence, see Fig. 6.2 and

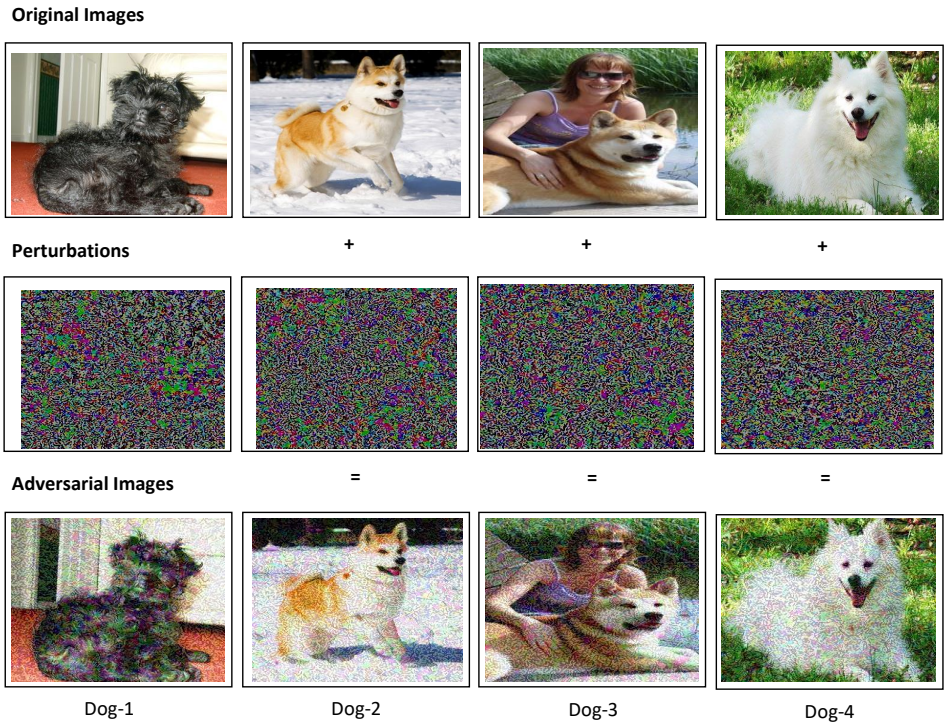


Figure 6.3: Example dog images. Original images are in the first row. The relevant perturbations are in the second row. The resultant adversarial images are in the third row.

Fig. 6.4. Similarly, for Dog-3 classification, the holistic level deep model predicts that it has a 13.93% resemblance to a cat, whereas, the constituent level deep models predict that it is 99.00% dog mouth. Moreover, the LCSs model predicts that it is 20.65% cat mouth, and 79.35% dog mouth. This prediction aligns with the component models prediction and empowers the novel system to make a correct dog prediction with high confidence, see Fig. 6.3 and Fig. 6.4.

Case-3: In this scenario, the deep model makes the wrong HLP for the whole image, the component deep models make correct predictions for different components, whereas, the LCSs model makes a partially con-

Image Name	Whole Image Prediction (DL Model)	Constituents Predictions (DL-Models)	Constituent Predictions (LCSs-Models)
Cat-1	33.05 % Cat	99% CM, 99% DM	87.61% CM, 12.39% DM
Cat-2	22.83% Dog	83% CE, 99% CN 99%CM	60.82% CE, 39.18% DE 90% CN, 10% DN, 30.39% CM, 69.61%DM
Cat-3	46.29% Cat	100% CN, 100% DN 99% CM, 99% DM	78.67% CN, 21.33% DN 99.69% CM, 0.31% DM
Cat-4	1.06% Dog	100% CN, 99% CM	96.02% CN, 3.98% DN 94.35% CM, 5.65% DM
Dog-1	64.79% Cat	99% DN	16.46% CN, 83.54% DN
Dog-2	21.04% Cat	99% DN, 94% DF	25.09% CN, 74.9% DN 73.99% CF, 26.01% DF
Dog-3	13.93% Cat	99% DM	20.65% CM, 79.35% DM
Dog-4	31.3% Cat	99% DE, 99% DN 100% DF	58.31% CE, 41.69% DE 30.13% CN, 69.87% DN 8.96% CF, 91.04% DF

Text Color: Correct Prediction Wrong Prediction Confused Prediction
Background Color: Case-1 Case-2 Case-3
Abbreviations: CE: Cat Eyes, CN: Cat Nose, CF: Cat Face, CM: Cat Mouth DE: Dog Eyes, DN: Dog Nose, DF: Dog Face, DM: Dog Mouth

Figure 6.4: Interpretation of decision-making process adopted by the lateralized system to classify adversarial images.

fused prediction, i.e. the majority of the predictions are correct but some are wrong. The correct predictions from the LCSs model along with the correct predictions from component deep models enable the lateralized system to make the correct final decision. For example, during the classification process of Cat-2, the holistic level deep model predicts that it has a 22.83% resemblance to a dog, whereas, the constituent level deep models predict that it is 83.00% cat eyes, 99.00% cat nose, and $+99.00\%$ cat mouth. Moreover, the LCSs model predicts that it is 60.82% cat eyes, 39.18% dog eyes, 90% cat nose, 10% dog nose, 30.39% cat mouth, and 69.61% dog mouth. Although the last prediction of the LCSs model is wrong, the other correct predictions from the LCSs model plus the correct predictions from component deep models outweigh the wrong predictions. Consequently, the novel system makes a correct cat prediction with high confidence. Similar behavior can be observed for the Dog-4 classification process, see Fig.

6.2, Fig. 6.3, and Fig. 6.4.

The current implementation of the lateralized system can handle only binary-class image classification tasks. However, the majority of the real-world problems are multi-class. Moreover, both the lateralized systems (i.e. for Boolean problems and binary-class image classification problems) have been developed by using LCSs. The lateralized approach is not limited to LCSs. An enhanced implementation of the lateralized system is presented in the next section that can handle multi-class image classification tasks and does not utilize the LCSs as underlying systems.

LCSs do not make any assumption about the linearity between the independent and dependent variables, which enables them to address epistatic/heterogeneous domains. However, as the number of classes grows in different problem domains, so does the number of possible variable interactions; eventually this makes evolutionary search impractical. However, the linearity assumptions reduces the search space, renders learning practical again (for a while) and works well for domains without epistic or heterogeneous patterns. Random forest (RF) is a state-of-the-art decision tree based ensemble learning technique that can directly handle high-dimensional data sets. RFs have been commonly used for regression and classification (binary/multi-class) problems [324, 325, 326].

6.3 Lateralized System for Multi-Class Image Classification

6.3.1 Lateralized System

The overall lateralized approach of the novel system is similar to the binary-class lateralized approach, see Section 6.2, except that the prediction can be generated by utilizing more constituent and holistic features and the underlying LCSs are replaced by random forest classifiers (see Fig. 6.5). The enhanced techniques of both phases are explained below.

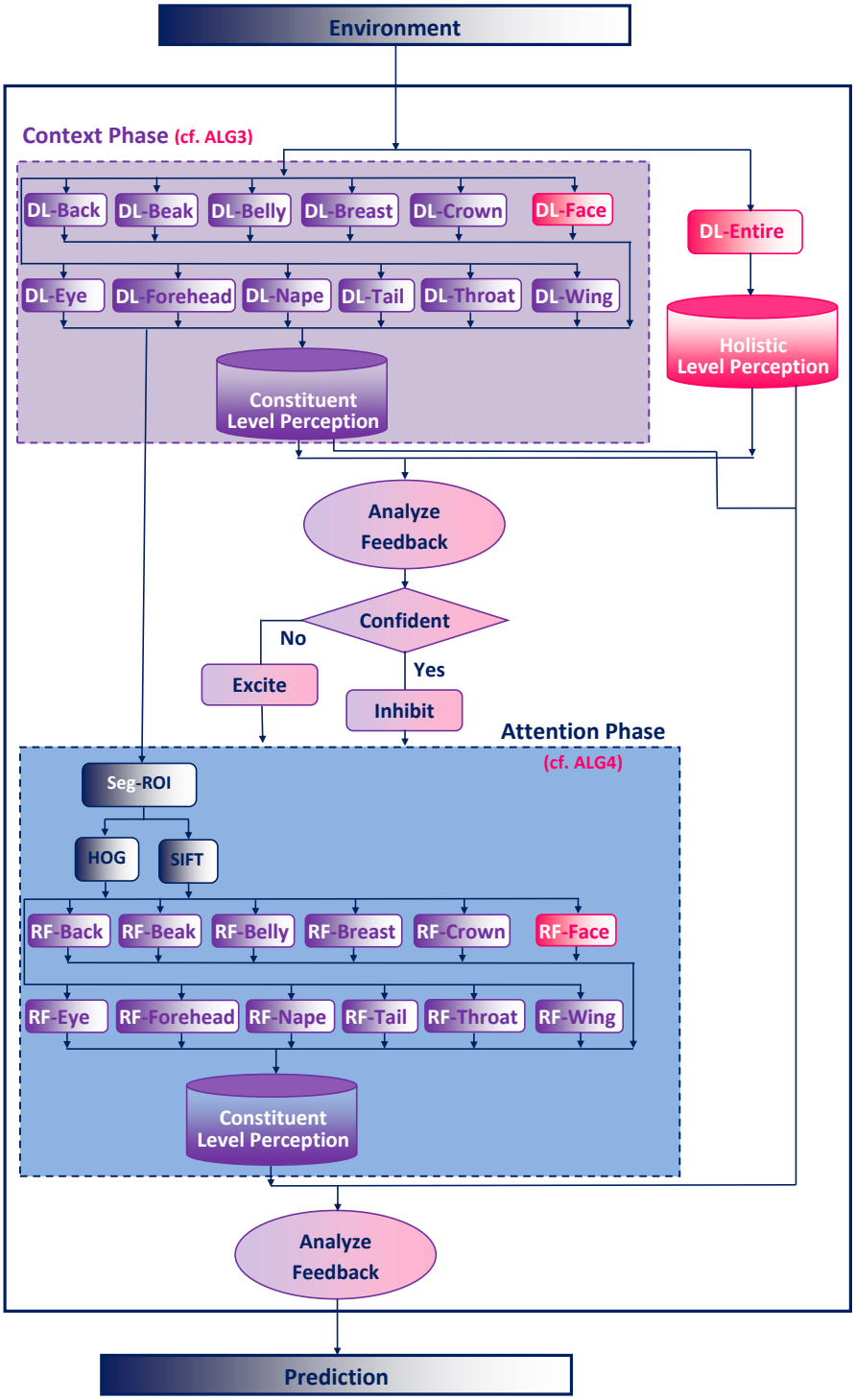


Figure 6.5: A schematic depiction of the strategies developed to achieve cognitive inspired functionality in the multi-class lateralized system. (color key: constituent, holistic, and mix knowledge proceedings are represented by purple-white, pink-white, and light purple-pink gradients, respectively)

Context Phase

The strategy developed for the context phase is improved by utilizing more deep networks to generate constituent and holistic features. Here birds classification is used as an illustrative example (see Chapter 3). The context phase consists of thirteen deep networks. Eleven of these deep networks are used to generate predictions about the constituent back, beak, belly, breast, crown, eye, forehead, nape, tail, throat, and wing, whereas two deep networks are used to generate configural predictions about the face and overall image. Here, the prediction is a probability that a part belongs to a candidate species (class).

Let $\mathcal{CM}_c$ is a constituent class matrix that contains the probability of an image belongs to a class. It has 200 entries (one entry for each class) and is initialized to 0. Each constituent deep model generates a prediction value for the given image, as shown in equation 6.1.

$$\mathcal{M}(img) = \begin{cases} C, & \text{class} \\ P, & \text{probability} \end{cases} \tag{6.1}$$

where $\mathcal{M}$ is a model that predict class C with probability P for a given image img. These prediction values are added in the constituent class matrix, as shown in equation 6.2.

$$\mathcal{CM}_c = \sum_{i=1}^{n} \sum_{\forall m} \mathcal{I} \times P \tag{6.2}$$

where n is the number of classes (200), m is the number of constituent models, and $\mathcal{I}$ is given below.

$$\mathcal{I} = \begin{cases} 1, & \text{C = i} \\ 0, & \text{Otherwise} \end{cases} \tag{6.3}$$

All the entries in the $\mathcal{CM}_c$ are normalized between 0 and 100. Finally, the probability of each class in the $\mathcal{CM}_c$ is compared and the class with

the highest probability is considered as a constituent level perception, as shown in equation 6.4.

$$\mathcal{CLP} = \max_{i \in [1,\ldots,n]} \mathcal{CM}_c(i) \tag{6.4}$$

In contrast, a deep model is used to obtain the holistic level prediction probabilities by utilizing the whole image. The resultant highest prediction probability is considered as a holistic level perception, as shown in equation 6.5.

$$\mathcal{HLP} = \max_{i \in [1,\ldots,n]} P(i) \tag{6.5}$$

where P is the prediction probability for class i, and n is the number of classes (200 in this case). It is important to note here that if a deep network is unable to predict any part (due to noise or adversarial attack), the default prediction value of 0 is used. The CLP and HLP information are shared with the system.

The system analyzes the feedback from CLP and HLP. If CLP and HLP support each other, i.e. both predict the same class, the system is confident to classify the given image correctly. Subsequently, generates an inhibit signal to the attention phase so that it can cease working. However, if CLP and HLP do not support each other (i.e. CLP and HLP predict different classes) or CLP is confused (i.e. more than one class has the maximum prediction value), the system cannot confidently classify the given image correctly. Subsequently, the system generates an excite signal to the attention phase and waits for the reply. The pseudo-code of the strategy adopted by the context phase to generate a prediction is given in Algorithm 9.

Attention Phase

The attention phase consists of thirteen random forest classifiers. Eleven of these RF are used to generate predictions about the constituent back, beak,

Algorithm 9: Strategy adopted by the context phase to generate a prediction (cf. Fig. 6.5).

Data: The data set and problem configurations
Result: Generate a prediction, inhibit or excite signal for attention phase

Initialize the global variable and parameter settings;
Compute CCM(); % *Compute Constituent Class Matrix (CCM).*
    Initialize CCM(); % *Initialize all the classes with* 0 *probability.*
    Generate Prediction From Deep Model(); % *Generate prediction probability for each part (back, beak, belly, breast, crown, eye, forehead, nape, tail, throat, wing,, face) or whole image.*
    Update CCM for each Deep Model(); % *Add all the prediction probabilities in the CCM.*
    Normalize CCM(); % *Normalize all the prediction values in the CCM between* 0 *and* 100.
Compute CLP(); % *Compare the prediction probabilities of all the classes in the CCM. The class with highest prediction probability is considered as a constituent level perception (CLP). It is important to note here that the prediction probability for the missed (e.g. deep network is unable to identify due to noise or adversarial attack) part is* 0 *by-default.*
Compute HLP(); % *The highest whole image prediction probability is considered as a holistic level perception (HLP).*
Analyse Feedback (); %*Analyze the feedback from CLP and HLP.*
    if *CLP and HLP Predict the Same Class* **then**
        % *System is confident to classify the given image correctly.*
        Generate Inhibit Signal(); % *generates an inhibit signal to the attention phase so that it stops working.*
        Generate Final Prediction();
    else
        Generate Excite Signal(); % *generates an excite signal to the attention phase and wait for the reply.*
    end

belly, breast, crown, eye, forehead, nape, tail, throat, and wing, whereas one RF is used to generate configural predictions about the face. Here, the prediction is a probability that a part belongs to a candidate species (class). The attention phase starts processing in parallel with the context phase. However, it stops immediately if it receives an inhibit signal from the context phase. The attention phase utilizes the deep models of the context phase to generate predictions about the bounding box (bbox) of the back, beak, belly, breast, crown, eye, forehead, nape, tail, throat, wing, and face. Subsequently, it segments each part according to the bbox values.

Similar to the attention phase of the binary-lateralized system, three variants of SIFT features and three variants of HOG features are computed from the segmented images (see Chapter 2). Note that the system is flexible enough to adopt any practical number. These features form the input instances (environment) for the RF classifiers. The RF classifiers generate prediction probabilities for each class. Another class matrix $\mathcal{CM}_a$ for attention phase is computed based on these probabilities by using equations 6.1 and 6.2. All the entries in the $\mathcal{CM}_a$ are normalized between 0 and 100. Finally, the CLP for the attention phase is computed by using equation 6.4. This final RF-based CLP value is returned to the system.

The system analyzes the deep model based CLP, RF-based CLP, and HLP. If two of these perceptions support the same class, the system favors that class and makes a prediction. Otherwise, the system computes another class matrix $\mathcal{CM}_f$ by adding the respective prediction probabilities of $\mathcal{CM}_c$, $\mathcal{CM}_a$, and holistic level prediction probabilities. Finally, the system compares the prediction probabilities of all the classes in $\mathcal{CM}_f$ and the resultant highest prediction probability is considered as the final prediction, as shown in equation 6.6. The pseudo-code of the strategy adopted by the attention phase to generate a prediction is given in Algorithm 10.

$$\mathcal{FP} = \max_{i \in [1,\ldots,n]} \mathcal{CM}_f(i) \tag{6.6}$$

Algorithm 10: Strategy adopted by the attention phase to generate a prediction (cf. Fig. 6.1).

Data: The data set and problem configurations
Result: Generate final prediction

```
Initialize the global variable and parameter settings;
Check Inhibit Signal();        % Check inhibit signal form context phase
  and stop immediately.
Get BBox From Deep Models();   % Get bbox prediction for each
  part (back, beak, belly, breast, crown, eye, forehead, nape, tail, throat,
  wing,, face).
Crop Img();        % Segment each part based on the bbox values.
Compute SIFT();    % Compute three variants of SIFT features for
  each segmented image.
Compute HOG();     % Compute three variants of HOG features for
  each segmented image.
Get RF Prediction();    % Get RF predictions for each part
Compute RF-CCM();       % Compute RF based Constituent Class
  Matrix (CCM).
Compute RF-CLP();
Share RF-CLP();
Analyse Feedback ();    %Analyze the feedback from RF-CLP,
  Context-CLP and Context-HLP.
  if Majority Favor Same Class then
      Generate Prediction();
  else
      Compute FCM();
          %Compute final prediction class matrix by adding the
        respective entries from CCM, RF-CCM, and holistic
        probabilities.
      Compute FP();     % The highest prediction probability in the
        FCM is considered as a final prediction.
  end
```

6.3.2 Experimental Design

This work seeks to show the scalability of lateralized approach. It can be achieved by conducting classification experiments on a multi-class images data set. The data set needs to have constituents level as well as holistic level ground truth information as it is required for the training of the constituent level prediction models and holistic level prediction models of the novel system. Note that such publicly available data sets are currently rare. For this work, the experiments are conducted for birds' species classification. The selected data set is publicly available and it has constituents and holistic levels labeled data.

Data Sets

This work uses publicly available birds data set (Caltech-UCSD Birds-200-2011) that has been used by the research community [267]. This dataset contains 11788 photographic images of 200 bird species. The ground-truth information about the parts and overall image is available. Each image contains 15 points annotation of the bird, i.e. (1) back, (2) beak, (3) belly, (4) breast, (5) crown, (6) forehead, (7) left eye, (8) left leg, (9) left wing, (10) nape, (11) right eye, (12) right leg, (13) right wing, (14) tail, (15) throat (see Chapter 3).

Data Preparation: The lateralized system needs the information (bbox) about the back, beak, belly, breast, crown, eye, forehead, nape, tail, throat, wing, and face of a bird in an image to train deep models and RF classifiers. The ground truth files of the chosen data sets have annotations related to these parts but not the bboxs. The system utilizes these annotations to generate the required information. For this purpose, separate routines are developed that take the annotation file as input and generate corresponding bboxs for each part and face. Moreover, these routines have the ability to handle rotated images and generate bboxs accordingly. Further explanation (logical description and algorithm) of these routines

is not presented as independent of the lateralization approach.

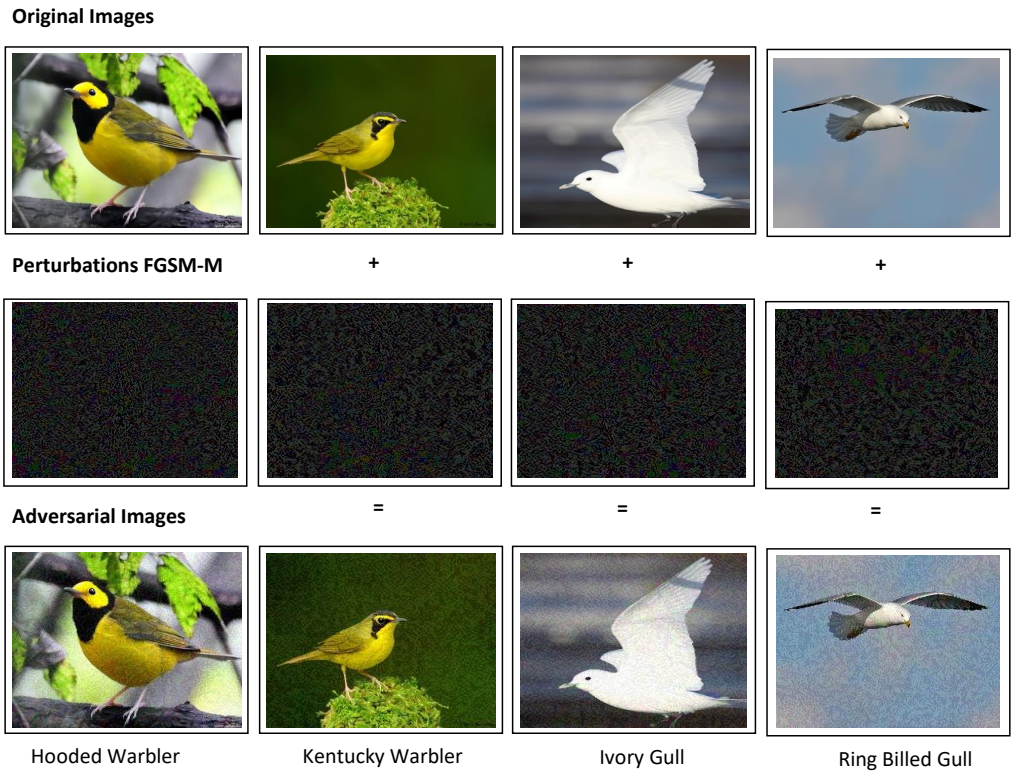
Experimental Setup

The learning strategy for the context phase is developed by utilizing multiple deep models depending on the nature and complexity of the problem, e.g. twelve deep models are used here. These models can be based on any state-of-the-art pre-trained deep network. The models used here are based on pre-trained 50 layers residual networks (ResNet50 model), which are well-recognized and widely used deep networks [319]. The loss function, mean absolute error (MAE), is used to train the deep networks, whereas, the state-of-the-art Adam optimization algorithm (stochastic gradient descent optimizer) is used to update the network weights [320]. Moreover, the deep models are trained for 300 epochs such the length of each epoch is 1000. It is important to note that ResNet50 model is also used to generate adversarial images to ensure that the adversarial attacks have the same underlying model. The learning strategy for the attention phase is developed by utilizing multiple (e.g. twelve here) RF classifiers. The RF classifiers are used with their default settings, i.e, no parameter tuning is performed. Moreover, the system is implemented using Scikit-learn libraries version 0.21.3 for Python version 3.7 [327].

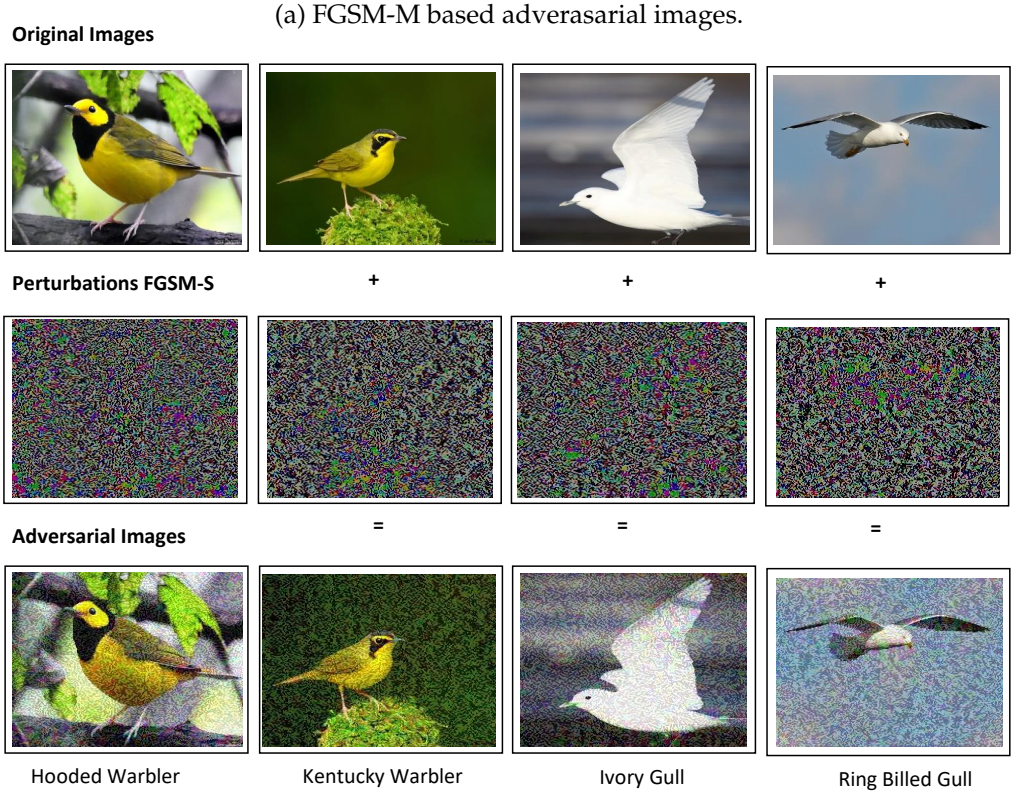
6.3.3 Experiments

For all the experiments, the 10-fold cross-validation technique is applied[2]. The lateralized system is trained once using the training images. The adversarial attacks are applied to the testing images only. The effectiveness of the novel system is evaluated by utilizing the original testing images and then the adversarial images. During the evaluation process, an image is presented to the lateralized system. Initially, both the context and

[2]The experimental results presented here are not the average of all the experiments. Some of the experiments are in progress and results will be updated later.

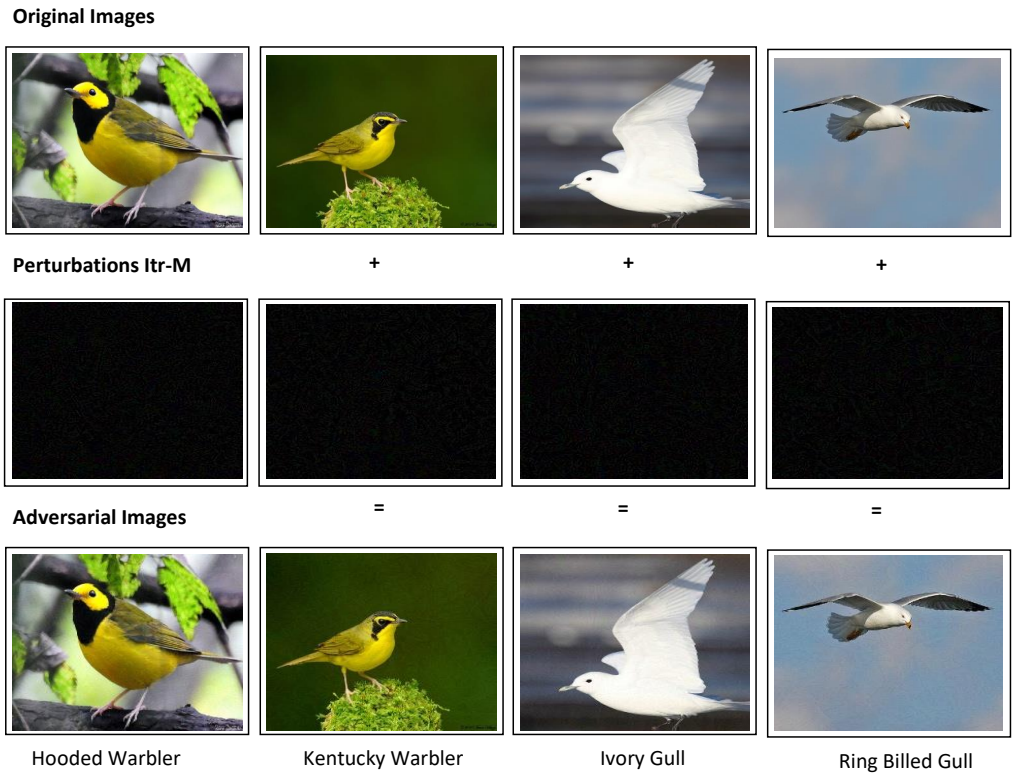


(a) FGSM-M based adverasarial images.

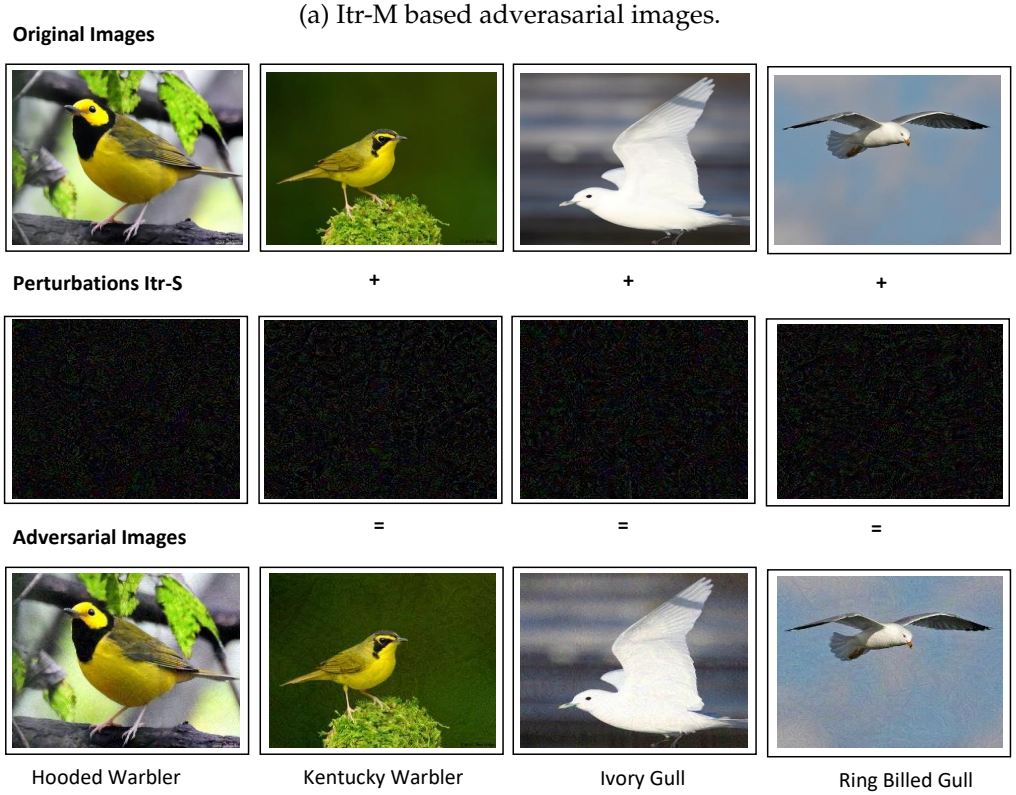


(b) FGSM-S based adversarial images

Figure 6.6: Example bird images of four different species. Original images are in the first row. The relevant adversarial perturbations are in the second row. The resultant adversarial images are in the third row.



(a) Itr-M based adverasarial images.



(b) Itr-S based adversarial images

Figure 6.7: Example bird images of four different species. Original images are in the first row. The relevant adversarial perturbations are in the second row. The resultant adversarial images are in the third row.

attention phases start processing simultaneously. The context phase computes CLP and HLP based on the predictions from deep models, whereas, the attention phase segments the images by utilizing the bbox information received from the deep models.

Three variants of SIFT features are computed by utilizing the following parameter values: patchSize: 64, 128, and 256; maxBinValue: 0.2, numOrientationBins: 8, and numSpatialBins: 2. Similarly, three variants of HOG features are computed by utilizing the following parameter values: image size: $64, 126$, and 256; pixels per cell: $32, 64$, and 128. All these features form the input environment instance for the RF classifiers. The attention phase computes CLP by utilizing the prediction probabilities of RF classifiers. Finally, the lateralized system receives all the CLPs and HLP and makes the final prediction accordingly.

Two variants of FGSM attacks are applied on the testing images to generate adversarial images, i.e. medium-level attack FGSM-M (epsilon (max norm) value 50) and strong attack FGSM-S (epsilon 150) [321]. Moreover, two variant of Iterative attacks are applied on the testing images to generate adversarial images, i.e. medium-level attack Itr-M (epsilon 18, alpha 1, and iterations 10), and strong attack Itr-S (epsilon 50, alpha 1, and iterations 10) [321]. Sample intact and adversarial images generated by applying FGSM-M, FGSM-S, Itr-S, and Itr-M are shown in Figs. 6.6a, and 6.6b, 6.7a, and 6.7b respectively. The experimental results of the novel system are compared with the results generated from four state-of-the-art deep models, i.e. ResNet [319], AlexNet [212], VGG [322], and SqueezeNet [323].

The experimental results demonstrate that the lateralized approach is scalable, see Table 6.2. The novel lateralized system outperformed all the state-of-the-art deep models for the classification of original test images by $19.05\% - 41.02\%$. Moreover, the novel system successfully exhibited robustness against three types of adversarial attacks (FGSM-M, Itr-M, and Itr-S). For all these attacks, the classification accuracy of the novel later-

Table 6.2: Classification Accuracy
Highest Accuracy is in bold.

	VGG	SqueezeNet	AlexNet	ResNet	LateralSys
OrigImgs	50.18	54.78	35.38	57.35	**76.40**
FGSM-M	15.83	19.40	15.85	15.08	**41.28**
FGSM-S	00.99	03.07	01.60	02.61	**04.43**
Itr-M	42.03	37.05	28.99	13.06	**61.17**
Itr-S	31.08	28.58	24.65	05.03	**54.25**

alized system is between $19.05\% - 41.02\%$ and 61.17%, whereas, the classification accuracy of all other systems is between $05.03\% - 41.02\%$ and 42.03%. The novel system could not completely resist the FGSM-S attack because it is a very strong adversarial attack that destroys the image contents badly. But all other deep models also perform worse. The novel lateralized system outperformed all the state-of-the-art deep models for the classification of adversarial images by $1.36\% - 49.22\%$.

6.4 Discussion

This work is designed to provide robust solutions for image classification problems against adversarial attacks. It is noted that the aim of this work is to create a system that exhibits natural robustness against adversarial attacks and not to devise another adversarial avoidance technique for a specific model or specific adversarial attack.

Although the novel architecture is not designed to model human (or other animal) vision, vertebrate brains do have complementary lateralised modules that represent objects at local (left) and global (right) levels. A detailed discussion about AI lateralization and vertebrate lateralization is presented in Chapter 8.

The novel lateralized system simultaneously considers (addresses) the

given problem instance at constituent and holistic levels. Moreover, a holistic level sub-problem in one representation may become a constituent level sub-problem in a higher-level representation. For example, during the context phase, processing the eyes, nose, and mouth are constituent level sub-problems, whereas the face is a holistic level sub-problem. However, during the final feedback analysis, the face becomes a constituent level sub-problem, whereas the whole image prediction is a holistic level sub-problem (see Section 6.2). This ability to address the problem at different scales empowers the novel system to successfully exhibit robustness against adversarial attacks. This is because an adversarial attack needs to successfully challenge both the constituents and holistic components of an image to fool the novel system.

The binary-class lateralized system applies an LCS-based strategy to resolve complex and ambiguous problems during the attention phase. The LCSs empowers the novel system to correctly classify images that are badly affected by the adversarial attacks. In the majority of the cases, the LCS models either predict the right class or collectively favor the right class decision, see Section 6.2.4. The built-in support for heterogeneity and niche-based algorithm of LCSs play a critical role in this regard. These two features of LCSs support lateralization which considers sub-problems (niches) at different levels of abstraction (heterogeneous). However, the improved version of the lateralized system shows that the lateralized approach is scalable and not limited to the LCSs. Instead of LCSs, it is the lateralized architecture that provide robustness against adversarial attacks. Note: not end-to-end learning, needs fine-grained labelled data (or sub-data sets), and RF make linearity assumptions to handle large number of classes practically.

The decision-making process of the novel system is interpretable, e.g. although the Dog-3 image has some resemblance to a cat (13.93%) it is classified as a dog because it has 99.00% dog mouth, see Section 6.2.4. The system, therefore, makes a step toward explainable artificial intelligence.

The novel system is an ensemble-like system, in that it resolves different components of a problem to make a final decision. However, the ability to consider the same sub-problem at different levels of abstraction and the use of excite/inhibit signals to activate/deactivate system components make it a lateralized system rather than an ensemble system.

6.5 Chapter Summary

The main objective in this chapter was to create a novel lateralized system for visual classification tasks. Lateralization was successfully applied to create a classification system that is robust against adversarial attacks. The ability to consider the same problem instance at different levels of abstraction (i.e. constituent level and holistic level) empowers the lateralized system to correctly classify the adversarial images. The novel system can handle simple problem instances at the context phase, whereas, more attention is automatically given to the noisy and corrupt problem instances based the feedback from the context phase. This strategy empowers the novel lateralized system to make correct decisions for badly corrupted images where either the constituent predictions are confused or the holistic prediction favor the wrong class. The experimental results demonstrate that the lateralized system successfully exhibits robustness against adversarial attacks. The novel binary-class system outperformed all the state-of-the-art deep models for the classification of normal and adversarial images by $0.43\% - 2.56\%$ and $2.15\% - 25.84\%$, respectively, whereas, the novel multi-class system outperformed all the state-of-the-art deep models for the classification of normal and adversarial images by $19.05\% - 41.02\%$ and $1.36\% - 49.22\%$, respectively.

Not only is much greater computer power needed to address more classes, but data sets with constituents features need to be created. In the next chapter, the idea of lateralization will be applied to resolve aliasing states in maze problems. These are multi-step problems that approximate

those in real life. The lateralized approach will enable the artificial agent to consider the maze problem at different levels of abstraction (constituent level or local viewpoint and holistic level or global viewpoint). The local viewpoint assists the artificial agent to make local decisions, whereas, the world (holistic) viewpoint assists the agent to resolve aliasing states at different locations of the maze. The experimental test will verify whether or not the lateralized system can resolve aliasing states in complex maze problems.

7

Frame-of-Reference based Learning: Overcoming Perceptual Aliasing in Multi-Step Decision Making Tasks

Perceptual aliasing challenges reinforcement learning agents. They struggle to learn stable policies through failing to identify and disambiguate perceptually identical states in the environment that require different actions to reach a goal. As the agent often has only a local frame-of-reference it cannot represent the global environment. Frame-of-reference based learning is a feature of vertebrate intelligence that allows multiple simultaneous representations of an environment at different levels of abstraction. This enables the resolution of patterns that are madeup of patterns that are madeup of features.

The developed lateralized framework can be adapted to resolve perceptual aliasing in multi-step decision making tasks. The evolutionary computation technique of learning classifier systems has now shown promise in learning nested patterns in single-step domains. Thus this work aims to develop a novel lateralized learning classifier system inspired by frame-of-reference in vertebrate brains for learning stable policies in non-Markov multi-step domains.

Considering aliased states at a constituent level enables the novel system to place them appropriately in holistic level policies. Experimental results show that the novel system effectively solves complex aliasing patterns in non-Markov environments that have been challenging to artificial agents. For example, the novel system utilizes only 6.5, 3.71, *and* 3.22 *steps to resolve Maze10, Littman57, and Woods102, respectively. Handling input at different levels of abstraction, i.e. frame-of-reference, simultaneously within a learning classifier system counters the problems of perceptual aliasing.*

7.1 Introduction

Navigation can be considered as a multi-step path planning problem. As a biological agent navigates its way through an environment toward a goal, it must use cues from the local environment to dictate the next step, while maintaining an updated spatial map of where that environment lies within the navigatable world. Perceptual aliasing occurs when identical local environments (e.g., a T-junction) are repeated at multiple locations within the world, meaning that the agent must know both the local cues and the location within the world map in order to know how to proceed.

Perceptual aliasing is a long-standing problem for artificial agents in

applying reinforcement learning (RL) to many multi-step tasks [10, 16, 57, 58, 59, 14, 60]. Here the agent is assumed to perceive its local environment without access to the global world view, which it needs to construct through interaction with the environment. Aliasing occurs when the agent's internal representation confounds the external world states, i.e. the agent's current perception is unable to distinguish environmental states which appear identical but require different actions [57]. In such a scenario, the reinforcement for the environment instructs the agent to take a specific action in a given state. Unfortunately, when the agent encounters an aliased state, it persists in taking the same action, which will now be reinforced differently. This inconsistency prevents the learning of stable policies, especially for multi-step tasks [328]. Perceptual aliasing, therefore, diminishes the effectiveness of reinforcement learning [58] and hinders its application to real-world problems [16].

RL agents can handle simple environments that do not have aliased states, e.g. Markov environments; however, they struggle in environments that have aliased states, e.g. non-Markov environments. Perceptual aliasing occurs only in non-Markov environments. A large number of approaches have been investigated to handle these perceptual aliasing problems [10, 57, 59, 14, 60, 11, 272, 273, 298, 299, 300, 302, 303, 306, 307, 308, 311]. These techniques can solve simple non-Markov environments but can not optimally resolve the majority of complex non-Markov environments. The limitations of these alternative techniques are presented in Chapter 3.

One factor that may have hampered progress in learning in non-Markov environments is the reliance on a local frame-of-reference (FoR) only, i.e. local viewpoint of the environment based on an agent's immediate perception. Hence the agent does not consider the environment at a higher level of abstraction (big-picture), which would allow it to uniquely identify aliased states. Consequently, aliased steps in a policy[1] are stored with

[1] A policy, like a route, can be considered as a large pattern prescribing state transitions

the same weight as non-aliased steps. It is asserted that an aliased state is a small pattern that gets repeated in an environment, which makes it difficult to identify where any specific instance occurs in the global environment.

These patterns can be combined with other patterns (aliased or not) to form higher level patterns and so forth. Eventually, each pattern, either in itself or part of a higher level pattern, is unique. Thus, non-Markov environments entail patterns that form hierarchical patterns, such as multiple aliased states at different positions in the environment. These aliased states can be identified uniquely (i.e. turned to non-aliased states) by considering an environment at different levels of abstraction simultaneously. Conventional RL systems struggle to capture such complex structures.

A hypothesized solution is to develop a learning system inspired by biological FoRs. In biological intelligence, an FoR is used to represent an environment from a viewpoint, e.g. a local viewpoint (from the agent's perspective) or world viewpoint (the complete map) [27, 28], see Chapter 2. FoRs enable vertebrate (and many invertebrate) brains to process the same information at multiple levels of abstraction [24, 26]. Moreover, FoRs are utilized to generate a grid map of the environment [148]. This grid map is utilized by the brain's coordinate mechanism for spatial navigation. The concept is that an artificial agent localizes itself by utilizing a local FoR. If it fails to uniquely identify its state (i.e. it is in an aliased state), it will utilize a higher level FoR. The agent will keep adding to its FoRs until it disambiguates the aliased states, which is then considered the world viewpoint.

Thus, a system is needed that can store representations of a state at different levels of abstraction and learn how these can be formed into a hierarchy to describe the patterns in a problem. The representation of a state at multiple levels of abstraction produces heterogeneous knowledge. Lateralization is a type of heterogeneity. The developed lateralized frame-

from a starting to the goal state, (see Section 7.2.2).

work can be adapted to develop a heterogeneous features based system that can consider an environmental instance at different levels of abstraction. An evolutionary machine learning (EML) system is capable of detailed learning of individual features, and abstract learning of the patterns of features (see Chapter 5). It is hypothesized that incorporating FoRs into an evolutionary machine learning system could allow it to overcome current limitations and learn to solve problems in non-Markov environments. In contrast to conventional systems that do not differentiate between detailed and abstract learning, the novel EML system is anticipated to solve problems in non-Markov environments by representing knowledge in both constituent and holistic frames of reference. The learning agent automatically identifies the level of abstraction that is required to successfully turn an aliased state into a non-aliased state. Finally, inspired by the brains' grid map, an adjacent states map (ASM) can be created for an environment. This ASM is utilized by the agent to differentiate aliased states based on the neighboring states. The EML technique to be used is Learning Classifier Systems (LCSs)[2] as they store learned knowledge in $if<state>then<action>$ rules. The states need to be stored in a format that links them through actions, which are termed code-paths (CPs) here. CPs form building blocks of knowledge[3] which are useful in themselves, but crucially can be constructed together to form higher level (more abstract) BBKs. This enables the system to function at appropriate levels of abstractions. The rules can provide elementary knowledge (or local viewpoint), which is needed to form the constituent level blocks of knowledge (CPs). Simultaneously, abstract knowledge (or the world viewpoint) can be formed by combining these CPs into holistic blocks of knowledge ((sub)policies of differing length, similar to routes). A CP will have the ability to accurately handle a non-aliased state as in an ordinary rule. Mul-

[2]LCSs have been used as a preferred research tool to evolve solutions for a wide range of maze problems for the last 30 years [309, 272, 241, 59, 329, 60].

[3]As previously, a building block of knowledge is a unit of knowledge that is transferable and can be used or reused to solve a part of a problem or the whole problem.

tiple CPs, policies, and the ASM will have the ability to provide a world viewpoint at higher levels of abstraction, which can be used to handle an aliased state. This process will essentially turn an aliased state at the constituent level into a non-aliased state at the holistic level. The agent can then create the optimal policy to reach the goal.

A schematic illustration of a conventional approach and novel (FoRs based) approach is shown in Fig. 7.1. Each state is represented by a colored circle. The multiple instances of the same color represent aliased versions of a state. The policies are represented by ellipses. A conventional EML approach (left side) relies only on local FoRs (local viewpoint) and considers individual features and niches in a homogeneous manner, i.e. all the states are treated the same, hence it does not generate unique patterns. The novel approach (right side) utilizes multiple FoRs and splits a complex problem into constituent and holistic knowledge. The local viewpoint (LV) identifies states at the constituent level; the world viewpoint (WV) places them appropriately in policies at a holistic level to generate unique patterns.

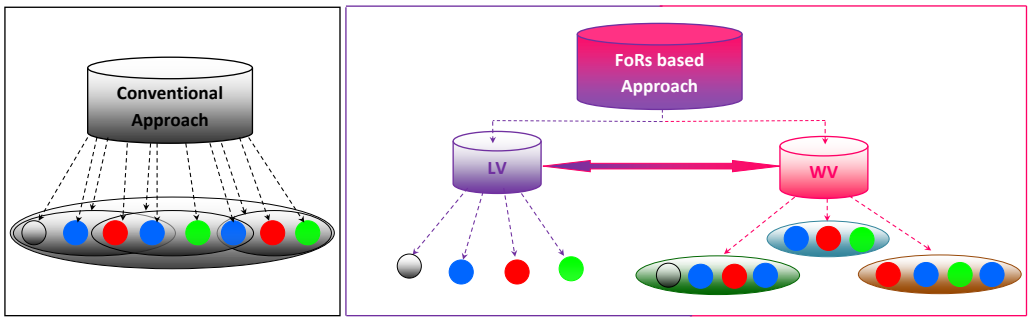


Figure 7.1: A schematic illustration of a conventional approach and novel (FoRs based) approach.

7.1.1 Chapter Objectives

The main objective reported in this chapter is to create a novel FoRs based system, inspired by the principles of animals' navigation, for decision making in state-transition learning. The system is to provide optimal solutions for non-Markov environments by utilizing FoRs that enable heterogeneous knowledge representations at different levels of abstraction. To achieve this objective, the following sub-objectives are set:

(i) Create a novel FoRs based system that has the ability to process a single input at different levels of abstractions to provide multiple environmental views, i.e. a local viewpoint (constituent knowledge) and a world viewpoint (holistic knowledge, complete map) of the same state.

(ii) Create a heterogeneous representation of knowledge, i.e. CPs and policies. This knowledge will be utilized or re-utilized at different levels of abstraction to generate constituent representations and holistic representations, that will allow interpretation of learned policies (see Section 7.5).

(iii) Integrate different blocks of knowledge (CPs) at different levels of abstraction to generate an unambiguous representation of knowledge. The resultant knowledge will be used to disambiguate complex patterns of aliased states, which will enable the learning of stable policies.

(iv) Create a strategy to activate/deactivate (sub)policies such that the agent can reach the goal state by using the minimum number of steps.

7.1.2 Chapter Organisation

The remainder of this chapter is organized as follows. Section 7.2 describes how a FoRs-based system can be created. It explains the critical components and architecture of the novel system. Markov and non-Markov environments are used to evaluate the developed system. The experimental setup is presented in Section 7.3. The effectiveness of the novel approach to optimally solve non-Markov environments is examined in Section 7.4. The interpretation of learned policies is presented in Section 7.5. Section 7.6 highlights the strength, drawbacks, and limitations of the novel approach. Finally, the chapter summary is presented in Section 7.7.

7.2 Frame-of-Reference based System

This work develops a FoRs based system for decision making to resolve non-Markov environments. We first introduce two novel components that are used to achieve heterogeneous knowledge representation at different levels of abstraction, i.e. the code-path (constituent knowledge), and the policy of code-paths (holistic knowledge). These techniques are assisted by a novel adjacent states map strategy that provides a snapshot of the environment. Subsequently, the utilization of code-paths and their policies for the identification and disambiguation of aliased states is described. Finally, the overall strategy adopted by the novel system to resolve non-Markov environments is presented.

7.2.1 Code-paths

A code-path is a GP-like tree (similar to a code fragment [15]) that encodes state-action-state sequences. Its format is a binary tree with depth up to two. Consequently, a CP can have a maximum of seven nodes, i.e. four states linked by three actions, see Fig. 7.2. This limit is set to keep tree size bounded to avoid intractable learning problems. A CP acts as a con-

stituent level BBK such that a single-step CP provides an egocentric (local) viewpoint, whereas, a multi-step CP or multiple CPs provide an allocentric viewpoint of the environment.

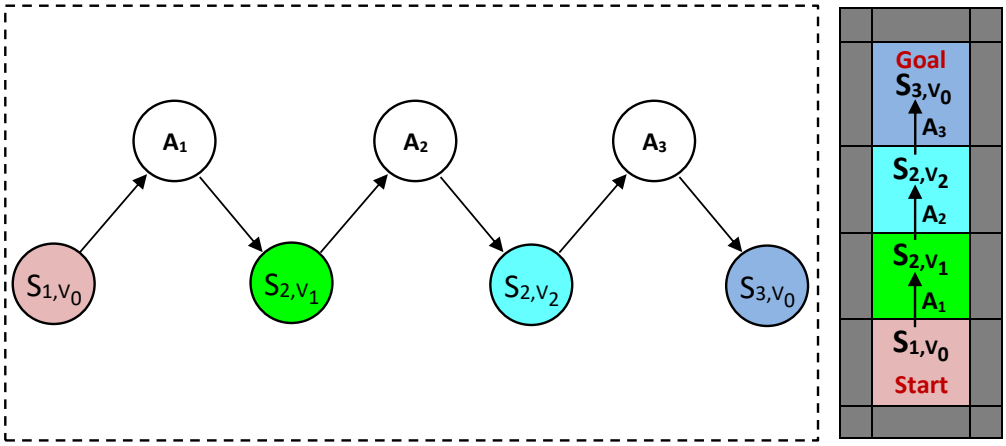


Figure 7.2: A sample maze (right) and the corresponding code path (left) when an agent moves from the start state (10000000 encoded – conventionally clockwise in a grid from the top; with 1 open, 0 blocked) to the goal state (00001000). With LCS perceived states S_{2,V_1} $(10001000, 1)$ and S_{2,V_2} $(10001000, 2)$ are non-aliased versions of the aliased state S_{2,V_0} (10001000).

A state is an environmental input instance and a version is its unique identity. All states have a default version of 0. The agent disambiguates aliased states *(S)* by assigning them different versions *(V)*. For example, the states S_1 and S_2 are two different egocentric states (i.e. different observations on the local scale). S_{1,V_0}, S_{2,V_1}, and S_{2,V_2} are three different allocentric states (i.e. different from a world viewpoint); V_0 indicates a state with no known aliasing; $V_1, V_2, \ldots, V_n$ indicate aliased versions of the state S_i.

Let $\mathcal{S}$ be a set of N_s states, $\mathcal{S} = \{S_i\}_{i=1}^{N_s}$, then we use "$S_{i,V_j}$" to refer to the j^{th} version (V_j) of the i^{th} state (i.e. S_i). Thus, the pool of states with their versions is represented as $\mathcal{SV} = \{S_{i,V_j}\}_{\forall i,j}$. Now, let $\mathcal{A}$ be a set of N_A actions, $\mathcal{A} = \{A_k\}_{k=1}^{N_A}$, then a CP can be defined as a state-action function

that walks through states using actions:

$$CP : \mathcal{SV} \times \mathcal{A} \longrightarrow \mathcal{SV}. \tag{7.1}$$

In practice, a CP is an alternate sequence of states and actions starting from a start state (say S_{l,V_m}) and ends at an end state (say S_{p,V_q}). For example, the CP in Fig. 7.2 can be represented by the sequence:

$CP =< S_{1,V_0}, A_1, S_{2,V_1}, A_2, S_{2,V_2}, A_3, S_{3,V_0} >$

Whenever an artificial agent moves from one state to another state during training, three nodes of the corresponding CP are created/updated. For example, when an agent currently in the state S_{1,V_0} executes an action A_1 and moves to another state S_{2,V_1}, the corresponding CP will insert/update these three entries in its nodes, respectively. A sample maze and the corresponding CP representing an agent that moves upwards from the start state to the goal state is shown in Fig. 7.2.

7.2.2 Policies

A policy, similar to a route in animals' navigation, is a pattern that prescribes state transitions from a start state to the goal state. Here, a policy is comprised of multiple CPs that are used by the agent while moving from a start position to the goal. Moreover, a policy has two associated attributes $-$ the number of steps used to reach the goal state (or steps), and the number of times the policy successfully guided the agent to the goal state (or experience). Fig. 7.3 illustrates two policies (red and yellow dotted lines), and corresponding CPs. The first policy, P_r, (red) consists of CP-1 and CP-3 which can lead the agent to the goal state by using 4 steps. The second policy,P_y, (yellow) consists of CP-2 and CP-3 which can lead the agent to the goal state by using 5 steps.

Let $\mathcal{CP}$ be a set of N_{cp} code paths, $\mathcal{CP} = \{CP_k\}_{k=1}^{N_{cp}}$, then the policy, P, can be defined as a subset of $\mathcal{CP}$. For example, in Fig. 7.3, there are three code paths, $\mathcal{CP} = \{CP_1, CP_2, CP_3\}$. The policy, P_r, consists of two

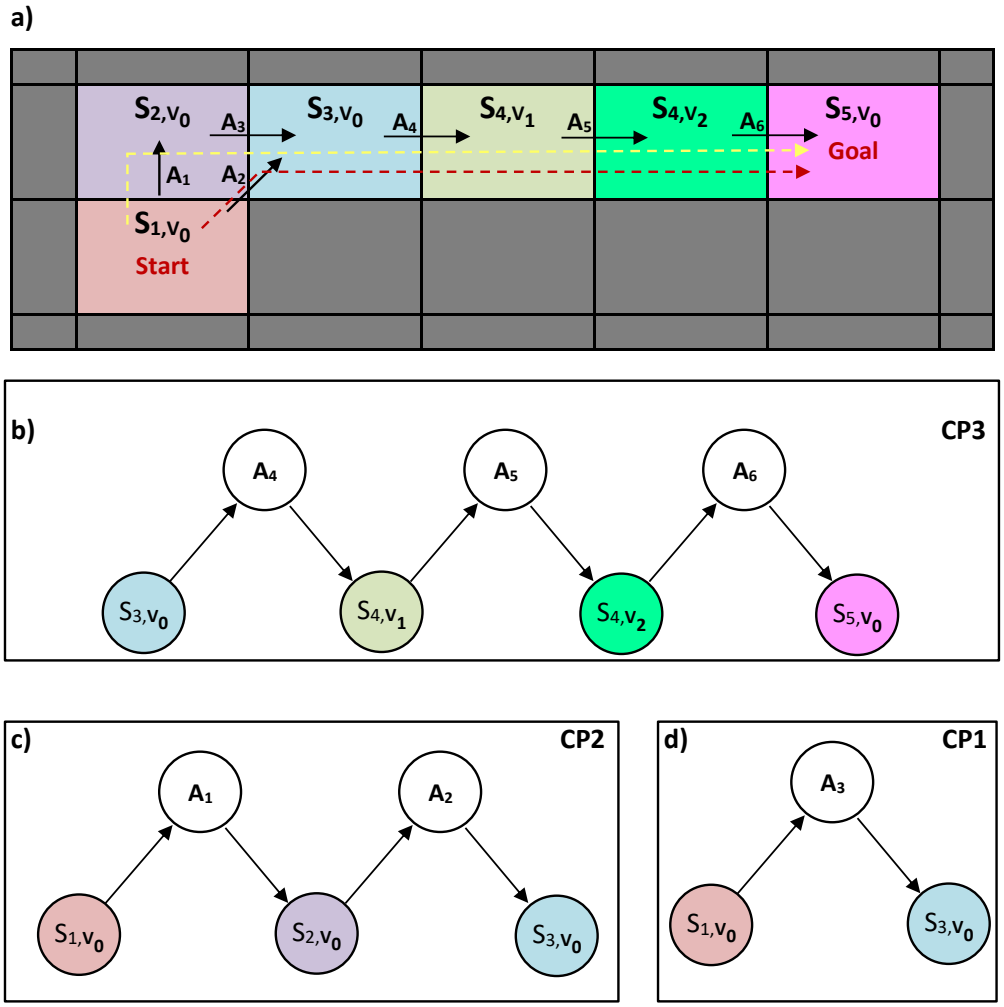


Figure 7.3: a) A sample maze and two policies. The first policy (red dotted line) consists of CP-1 and CP-3. The second policy (yellow dotted line) consists of CP-2 and CP-3. b) A three-step CP (CP-3) from S_{3,V_0} to S_{5,V_0}. c) A two-step CP (CP-2) from S_{1,V_0} to S_{3,V_0}. d) A single-step CP (CP-1) from S_{1,V_0} to S_{3,V_0}. A_i can take any action value between 0 and 8.

CPs and it is represented as $P_r = \{CP_1, CP_2\}$. Similarly, the policy, P_y, is represented as $P_y = \{CP_2, CP_3\}$. In general, the set of all N_p possible polices is represented as $\mathcal{P} = \{P_l\}_{l=1}^{N_p}$, where the goal is to find the optimal policy, $P_{\hat{l}}$, which has the lowest cost, i.e. $cost(P_{\hat{l}}) \leq P_l, \forall l = 1, \ldots N_p$. Such a cost is computed based on two associated attributes, i.e. the number of steps and experience.

The policy is a holistic level knowledge representation, which provides an allocentric viewpoint (world viewpoint) of an environment. It is created during the explore mode in two situations: (i) when an agent successfully reaches the goal, or (ii) when an evolutionary process is triggered. In the first scenario, the agent logs the path that is used to reach the goal. Loops are removed from the path before creating a policy. In order to remove a loop, the whole path is traversed such that if the current state (and version) already exists in the path, the horizontal and vertical distances are computed to determine whether it is the same state or a new aliased state (distance computation is presented in Section 7.2.4 Case-5). If the resultant horizontal and vertical distances are zero then it is the same state, that is, the agent has completed a loop, and so the states visited by the agent during this loop are removed from the path.

In the second scenario (ii), an evolutionary process is triggered when the average number of time-steps since the last crossover is greater than θ_{GA}. Two new policies are created by applying a crossover technique on two existing policies such that both policies have a common state (with the same version) in their paths. The first policy is randomly selected from the pool of existing policies. A second policy is randomly selected until one with a common state with the first policy is found. If compatible policies are found, the common state between these policies acts as a crossover point such that the new child policies are created by combining the opposite halves from the parent policies, otherwise no new policies are created.

A_0		A_1		A_2		A_3		A_4		A_5		A_6		A_7		ID	
0	0	S_2	V_2	0	0	0	0	0	0	0	0	0	0	0	0	S_1	V_1
0	0	0	0	0	0	0	0	0	0	S_1	V_1	0	0	0	0	S_2	V_2

Figure 7.4: An example section of an adjacent states map.

7.2.3 Adjacent States Map

The ASM contains information about the neighboring states within the environment. It is a sparse, dynamic map that is developed/updated at runtime while the agent explores the environment. The number of rows of the ASM is equal to the number of states visited by the agent. The columns of the ASM are equal to the number of possible actions that can be taken by the agent, plus the last column (ID) that is reserved for naming each state. Each non-heading cell contains a state-version tuple. For this work, as is common in maze navigation tasks, an agent can execute eight actions (separated by $45°$) to move to a neighboring state where the action is without noise. Whenever an agent moves from one state to another, all the corresponding entries (adjacent states) in the ASM are created/updated. For example, an agent moves from a state S_{1,V_1} to another state S_{2,V_2} by executing an action A_1. Two entries in the ASM are created representing (i) state S_{2,V_2} at column A_1 in a row against state-id S_{1,V_1}, and (ii) state S_{1,V_1} at column A_5 in a row against state-id S_{2,V_2}. It is important to note that the agent can move back to the original state by executing a flipped (opposite) action, e.g. A_5 is a flipped action of A_1. A section of the ASM, corresponding to this movement, is shown in Fig. 7.4.

7.2.4 Aliasing Identification and Disambiguation

The most challenging step in learning a non-Markov environment is identifying and disambiguating the aliased states, as local knowledge provides

conflicting information. Learning is achieved by comparing and integrating different levels of CPs. The single-step CPs represent the egocentric viewpoint, whereas multi-step CPs represent the allocentric viewpoint. Multi-step CPs and policies are used, along with the ASM, to generate the world viewpoint (complete map) of the environment. During this process, new CPs are created and incompatible CPs are updated or deleted. There are five distinct cases that can lead to the identification of an aliased state.

Case-1

In a specific state the agent effects a previous action but *transitions to an unexpected state*. For example, the agent moves from state S_{1,V_0} to state S_{3,V_0} by executing an action A_1, but another CP already exists, moving from state S_{1,V_0} to state S_{2,V_0} by executing the same action A_1. Thus, state S_{1,V_0} is marked in the ASM as an aliased state and is disambiguated into two states, here termed state S_{1,V_1} and state S_{1,V_2}. Consequently, the agent is only confident about its transitions due to the actions A_1 and flipped-A_1 (i.e. A_5). The transitions due to all other actions now become ambiguous because the agent cannot discern the correct non-aliased version (S_{1,V_1} or S_{1,V_2}).

The agent updates the unambiguous knowledge and deletes the ambiguous knowledge. For this purpose, all the CPs with entries "$S_{1,V_0} \xrightarrow{\text{A1}} S_{2,V_0}$" are updated with "$S_{1,V_1} \xrightarrow{\text{A1}} S_{2,V_0}$". Similarly, the CPs with flipped action (A_5) entries "$S_{2,V_0} \xrightarrow{\text{A5}} S_{1,V_0}$" are updated with "$S_{2,V_0} \xrightarrow{\text{A5}} S_{1,V_1}$". The corresponding entries in the ASM and multi-step CPs are also updated. Moreover, all the entries in CPs from S_{1,V_0} that transition to another state by executing an action other than A_1 are deleted as they are no longer reliable. Similarly, the flipped entries from another state to S_{1,V_0} with an action other than A_5 are deleted. The corresponding information is also deleted from the ASM and multi-step CPs. Consequently, the agent updates learning and, most importantly, learns to forget. Finally, two new CPs for the

move "$S_{1,V_2} \xrightarrow{\text{A1}} S_{3,V_0}$" and "$S_{3,V_0} \xrightarrow{\text{A5}} S_{1,V_2}$" are created. For this newly disambiguated aliased state S_{1,V_2}, a new row in the ASM is created and corresponding information is added to that row.

Case-2

The agent ends in the same state from the same action, but *transitioned from an unexpected starting state*. For example, the agent moves from a state S_{3,V_0} to another state S_{2,V_0} by executing an action A_2 but another CP already exists that transitions from state S_{1,V_0} to state S_{2,V_0} by executing the same action A_2. The state S_{2,V_0} is marked as an aliased state and is disambiguated into two states, here termed state S_{2,V_1} and state S_{2,V_2}. Subsequently, all the CPs with entries "$S_{1,V_0} \xrightarrow{\text{A2}} S_{2,V_0}$" are updated with "$S_{1,V_0} \xrightarrow{\text{A2}} S_{2,V_1}$". Similarly, the CPs with flipped action "$S_{2,V_0} \xrightarrow{\text{A6}} S_{1,V_0}$" are updated with "$S_{2,V_1} \xrightarrow{\text{A6}} S_{1,V_0}$". The corresponding entries in the ASM and multi-step CPs are also updated. Moreover, all the entries in CPs from any state to S_{2,V_0} by executing an action other than A_2 are deleted as unreliable. Similarly, the flipped entries from S_{2,V_0} to another state with an action other than A_6 are deleted. The corresponding information is also deleted from the ASM and multi-step CPs. Consequently, the agent updates learning and learns to forget. Finally, two new CPs for the move "$S_{3,V_0} \xrightarrow{\text{A2}} S_{2,V_2}$" and "$S_{2,V_2} \xrightarrow{\text{A6}} S_{3,V_0}$" are created. For this newly disambiguated aliased state S_{2,V_2}, a new row in the ASM is created and corresponding information is added to that row.

Case-3

The agent does not effect the previous action, but *a new action that transitions to the same state*. For example, the agent moves from state S_{2,V_0} to another state S_{1,V_0} by executing an action A_3 but another CP already exists from state S_{2,V_0} to state S_{1,V_0} by executing a different action $\neq A_3$. The state S_{1,V_0} is marked as an aliased state and is disambiguated into

two states, here termed state S_{1,V_1} and state S_{1,V_2}. Subsequently, all the CPs with entries "$S_{2,V_0} \xrightarrow{\neq A3} S_{1,V_0}$" are updated with "$S_{2,V_0} \xrightarrow{\neq A3} S_{1,V_1}$". Similarly, the CPs with flipped action "$S_{1,V_0} \xrightarrow{\neq A7} S_{2,V_0}$" are updated with "$S_{1,V_1} \xrightarrow{\neq A7} S_{2,V_0}$". The corresponding entries in the ASM and multi-step CPs are also updated. Moreover, all the entries in CPs from S_{1,V_0} to another state by executing an action other than $\neq A7$ are deleted. Similarly, the flipped entries from another state to S_{1,V_0} with an action other than $\neq A3$ are deleted. The corresponding information is also deleted from the ASM and multi-step CPs. Consequently, the agent updates learning and learns to forget. Finally, two new CPs for the move "$S_{2,V_0} \xrightarrow{\text{A3}} S_{1,V_2}$" and "$S_{1,V_2} \xrightarrow{\text{A7}} S_{2,V_0}$" are created. For this newly disambiguated aliased state S_{1,V_2}, a new row in the ASM is created and corresponding information is added to that row.

Case-4

The agent effects an action and transitions to a new state; however, a CP already exists that *effects a flipped action from the new state that transitions to a different state*. For example, the agent moves from state S_{1,V_0} to another state S_{2,V_0} by executing an action A_4 such that no other CP already exists from state S_{1,V_0} to any other state by executing the same action A_4. Subsequently, two CPs for this move need to be created, i.e. "$S_{1,V_0} \xrightarrow{\text{A4}} S_{2,V_0}$" and "$S_{2,V_0} \xrightarrow{\text{A0}} S_{1,V_0}$". During the creation of CP with flipped action, i.e. "$S_{2,V_0} \xrightarrow{\text{A0}} S_{1,V_0}$", it is found that a CP already exists from state S_{2,V_0} to another state S_{3,V_0} by executing the same action A_0. The state S_{2,V_0} is marked as an aliased state and is disambiguated into two states, here termed state S_{2,V_1} and state S_{2,V_2}. Subsequently, all the CPs with entries "$S_{2,V_0} \xrightarrow{\text{A0}} S_{3,V_0}$" are updated with "$S_{2,V_2} \xrightarrow{\text{A0}} S_{3,V_0}$". Similarly, the CPs with flipped action "$S_{3,V_0} \xrightarrow{\text{A4}} S_{2,V_0}$" are updated with "$S_{3,V_0} \xrightarrow{\text{A4}} S_{2,V_1}$". The corresponding entries in the ASM and multi-step CPs are also updated. Moreover, all the entries in CPs from S_{2,V_0} to another state by executing an action other than A_0 are deleted. Similarly, the flipped entries from other states to S_{2,V_0}

	A_0	A_1	A_2	A_3	A_4	A_5	A_6	A_7
Angle	0	45	90	135	180	225	270	315
X-Axis	0	1	1	1	0	-1	-1	-1
Y-Axis	1	1	0	-1	-1	-1	0	1

Figure 7.5: The horizontal and vertical distances against actions. The action A_0 is at 0° and each subsequent action is separated by 45°.

with an action other than A_4 are deleted. The corresponding information is also deleted from the ASM and multi-step CPs. Consequently, the agent updates learning and learns to forget. Finally, two new CPs for the move "$S_{1,V_0} \xrightarrow{\text{A4}} S_{2,V_2}$" and "$S_{2,V_2} \xrightarrow{\text{A0}} S_{1,V_0}$" are created. For this newly disambiguated aliased state S_{2,V_2}, a new row in the ASM is created and corresponding information is added to that row.

Case-5

An aliased state already exists in a path. This case is executed only if the agent cannot identify an aliased state by utilizing any of the above-mentioned cases. The agent logs its path during the navigation. If the current state (and version) already exists in the path, the agent computes the horizontal and vertical distance to determine whether it is in the same state (i.e. identifies a loop) or a new aliased state. For this work, the action A_0 is at 0° and each subsequent action is separated by 45°. The distance values against all the actions are presented in Fig. 7.5. To compute the distance, the following procedure is applied. For each action executed by the agent, it moves to a new state, a value ($+1, -1$ or 0) is added to the horizontal and vertical distances. For example, against action A_0 the values 0 and 1 are added to the *x*-axis and *y*-axis, respectively. These values are added for all the states between the current state and the already visited state by traversing through the path. Finally, if the agent determines that it has traveled a non-zero horizontal or vertical distance, it marks its state as a

new aliased state and disambiguates it by assigning a new version.

7.2.5 Predict Aliased Version

The prediction of the correct aliased version (V) of a state(S) is an important step in resolving non-Markov environments. The agent disambiguates aliased states by assigning them unique versions, as an aliased state with a unique version behaves as a non-aliased state. The agent is confident about its position in the environment if it is in a non-aliased state or an aliased state with the correct version. Consequently, the agent moves to the next state by executing the appropriate action that leads to the optimal path to the goal. The techniques adopted by the agent to predict the aliased version are explained below.

Case-1

The *agent is confident about its position in the environment*, i.e. either it is in a non-aliased state or an aliased state with the correct version. When the agent moves from a confirmed state to an aliased state, it extracts the correct version of the aliased state from the CPs. For this purpose, if a CP exists that has the same initial $state - version \xrightarrow{\text{action}} state$ entries, the version of the current aliased state is set from that CP.

Case-2

The *agent is not confident about its position in the environment*, i.e. either a random starting point is an aliased state or there exist multiple CPs with this same state but different versions. Consequently, the agent is in an aliased state with a default version 0. When the agent moves from such a state to another aliased state, it applies one of the following techniques to predict the aliased version.

First, if the current path of the agent has more than two states, it finds matching multi-step CPs by comparing only the states (not versions) and

actions with the last three moves (because a CP can support at most three moves; this number could be extended at the cost of additional computations). If such a CP exists, the agent considers the aliased state version of that CP as a candidate version. To verify that candidate version, the agent searches for the flipped CP, i.e. from the current state to the previous state. Subsequently, the agent counts all the CPs that have the same flipped action but result in different states. If this number is equal to the total number of aliased versions for that state, the agent flags the prediction as correct. Otherwise, the candidate version is discarded.

If the first technique does not find the aliased version, the agent identifies the flipped CPs from the current state to the previous state without comparing the versions. If the candidate CPs have multiple versions for the current aliased state they are discarded. Otherwise, the agent finds all the CPs that have the same flipped action but different states. If the number of CPs found is equal to the total number of aliased versions for that state, the agent marks the prediction as correct. Otherwise, it is discarded. If the agent fails to make a correct prediction, the default version 0 is used, which means that the agent is not yet confident about its position in the environment.

Case-3

This case is executed if the *agent cannot predict an aliased state version* by utilizing the above-mentioned cases. The agent attempts to predict the aliased version by utilizing the information from the ASM. The agent makes a list of the current states (without comparing the versions) that have the previous state in their neighbors at the flipped action. These are considered candidate states. Subsequently, the agent compares the shared neighborhood of each candidate state with the previous state. If a state has the same neighboring states as the shared neighborhood, that state is flagged as a final candidate state. At the end of this process, if there is only one final candidate state, the version of that state is considered a correct aliased

version. Otherwise, the agent is not confident about its position in the environment, and the default version 0 is used.

7.2.6 Overall Strategy

EC generates the constituent rules using CFs to encode the conditions of the state and actions. Many encodable rules are not valuable, e.g. rules where actions collide with walls, so EC search improves efficiency and reduces storage. Moreover, EC is used to utilize these constituent rules to evolve CP-based policies in the form of state-action-state tuples. Again this removes redundancy/irrelevant conditions that would be kept by exhaustive search. A schematic depiction of the overall strategy developed to achieve the cognitive inspired functionality in the FoRs-based system is shown in Fig. 7.6. The general state-action-reward scheme of the novel system is similar to the standard multi-step reinforcement learning scheme in LCSs [303, 14]. The learning methodology of the novel FoRs system is developed by utilizing the framework of accuracy-based LCSs, i.e. Wilson's XCS [251]. Code fragments assist the novel system to link the environmental features through functional nodes. A disjunctive normal form of CFs constitute a rule, which encapsulates how well CFs link together to provide an egocentric viewpoint of the environment. As in the standard LCSs, here the rules are created by three methods, i.e. covering, crossover, and mutation. These rules are combined in a population, which enables specific niches of the problem to be combined together to solve different parts of the problem domain. The FoRs system departs from a conventional LCS in the novel use of state versions within the conditions of the classifier rules are enhanced with a state-version encoding. In order to appear in the match set, the rules need to match the condition, including the version, of the state.

At the start of the learning process, all the states (aliased and non-aliased) have version 0. The agent is randomly placed in a state in the environment. The agent obtains the version of the current state by ap-

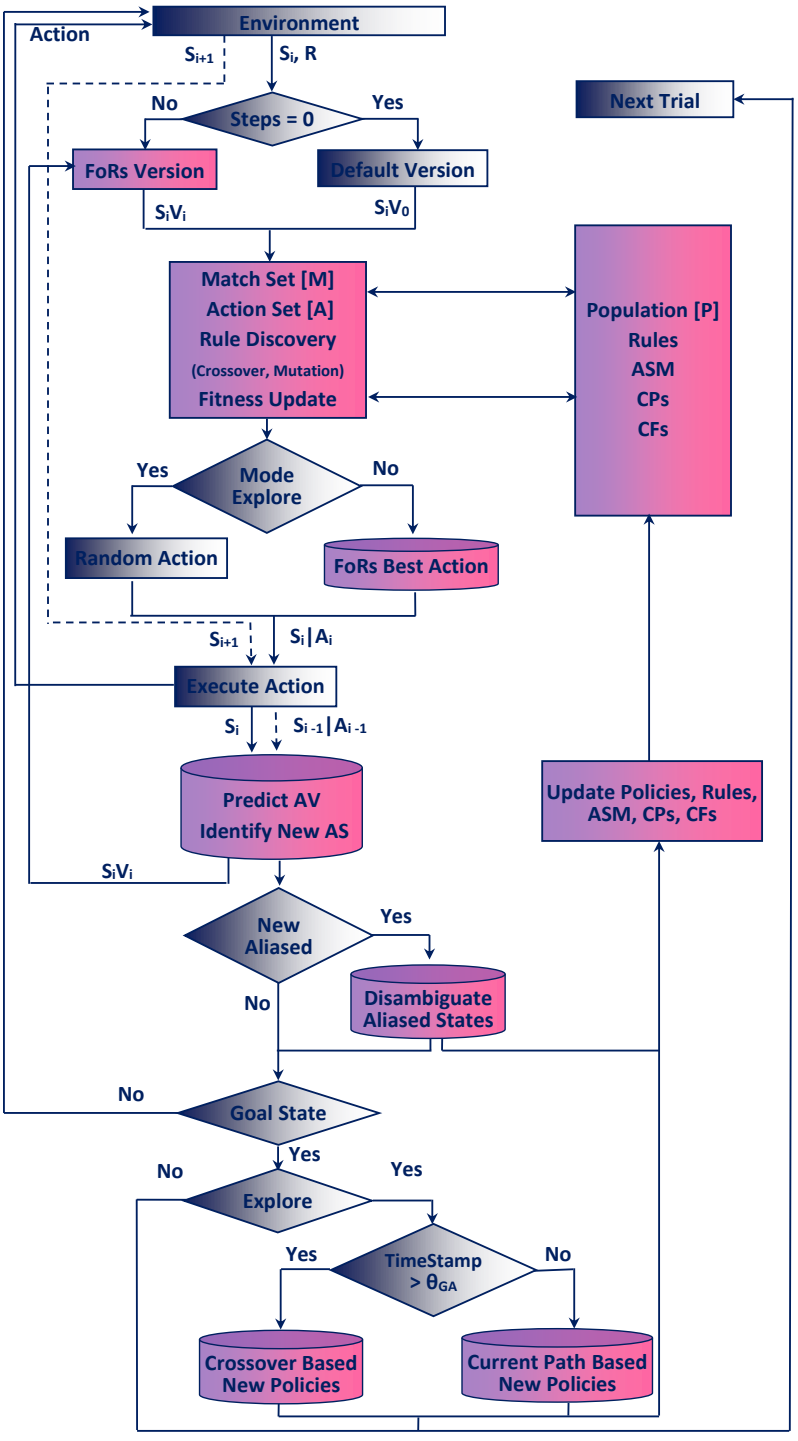


Figure 7.6: A schematic depiction of the overall strategy developed to achieve cognitively inspired functionality. (color key: constituent, holistic, and mix knowledge proceedings are represented by purple-white, pink-white, and light purple-pink gradients, respectively)

plying methods presented in Section 7.2.5. Subsequently, the agent determines whether it is an aliased state or not. If it is an aliased state, the agent disambiguates the aliased states by assigning them unique versions. The methods adopted by the agent for the identification and disambiguation of aliased states are presented in Section 7.2.4.

The learning process of the agent is divided into explore and exploit modes, similar to the standard LCS process. The agent logs the path while navigating through the environment. During the explore mode, the system attempts to create a new policy if one does not exist for the current path. A new policy can be created in two situations: (i) when an agent successfully reaches the goal, or (ii) when an evolutionary process is triggered. The method to create a new policy in the first situation is presented below, whereas, the conditions to trigger the evolutionary process to subsequently create new policies is presented in Section 7.2.2. A new policy is created if the agent successfully reaches the goal state before utilizing the maximum allowed steps, which is set with domain knowledge. For this purpose, the loops are removed from the path. Subsequently, the path is virtually traversed in reverse order, i.e. from the goal state to the start state. For each step, the ambiguous aliased versions (i.e. 0) are updated with the correct versions. New CPs are created if they do not exist, by applying the strategies explained in Sections 7.2.4 and 7.2.5. New aliased versions may be created to disambiguate the aliased states from the path. During this process, if no new aliased versions are created and no ambiguous version is left in the path, the new policy is created if it does not already exist. Moreover, the experience attribute of the new policy is initialized to zero and incremented by one each time the policy assists the agent to successfully reach the goal state.

The reverse traversing of the path enables the novel system to connect the isolated blocks of aliased states. If the agent cannot establish a connection (by finding or creating CPs) between the states of the path by utilizing the above-mentioned methods, the agent attempts to find blocks

of aliased states that match the path at the point of missing connection (i.e. when no CP exists). Utilizing the ASM, such blocks are identified through analysing the multi-step CPs. If a block is found that matches the current path and is isolated from other neighboring states, then it is connected with the broken state by creating a new CP. An example of such a scenario is presented in Section 7.5.

During exploit mode, the system activates a policy that assists the agent to take the best action. In order to select the most appropriate policy, the system must identify the correct version of the state. Therefore, for aliased states, the agent predicts the most likely aliased version by utilizing the strategies explained in Section 7.2.5. However, if the agent is unable to predict the version, the system randomly selects one. Subsequently, the agent tries to identify the best policy for the selected state-version. Each state may have more than one policy. The novel system selects a valid policy that has the smallest value for the step attribute. That policy can lead the agent to the goal state by utilizing the minimum number of steps. Finally, the agent activates the selected policy (cf. roll-out).

The agent determines the best action using two different strategies: the action set, and the active policy. If two actions are the same, the policy is marked as true by setting a flag (termed "cognate"), which can take value [true, false]. The agent is confident about its decision and executes the action. However, if the actions are different, then the agent prefers the action provided by the policy and marks the cognate flag for the policy as false. Consequently, if the agent moves to a different state predicted by the policy or is unable to move, the policy is marked as malign by setting a flag (termed "malign"), which can take value [true, false], and the agent picks another best policy with respect to the current state. The malign policies will not be selected again for the current multi-step run. At the end of the multi-step run, all the cognate and malign flags are cleared. A walk-through of this novel approach is presented in Section 7.5.

7.3 Experimental Design

This work seeks to demonstrate the effectiveness of the FoRs-based approach to address the perceptual aliasing problem in RL agents while solving multi-step tasks in non-Markov environments. Maze problems are used as a test paradigm to investigate how agents learn state-action transitions in multi-step environments. Mazes have been used in a wide variety of navigation based research from cognitive neuroscience to artificial intelligence [27, 268, 269, 270, 123, 14, 244, 271, 272, 273], as they approximately simulate real-world navigation problems. Mazes have a structure that allows experimenters to easily control and trace the behavior of an agent during the learning process. They offer a wide range of complex environments that artificial agents struggle to solve. This includes complex non-Markov mazes that are characterized by heterogeneity in action probability in a given state and clusters of such aliased states. As these characteristics make maze problems effective to evaluate the novel approach, a wide range of mazes are used as the test domain. These mazes include both deterministic and non-Markov environments, all aliasing types (I, II, and III), and a broad range of complexity $(1 - 251)$ [309, 272]. The majority of these mazes and the related woods environments have been used in state-of-the-art studies [309, 303, 14, 272, 330]. A brief introduction of a maze environment and the mazes that are used in this work, for readers unfamiliar with these environments, are presented in Chapter 3.

7.3.1 Experimental Setup

The novel system uses the XCS configuration settings that have been commonly used in XCS studies [303, 14]. The parameter values are: crossover probability $\chi = 0.8$; GA threshold $\theta_{GA} = 25$; mutation probability $\mu = 0.04$; learning rate $\beta = 0.2$; deletion fraction $\delta = 0.1$; deletion threshold $\theta_{del} = 20$; prediction error threshold $\varepsilon_0 = 10$; fitness exponent $\nu = 5$; fitness fall-off rate $\alpha = 0.1$; fitness reduction $= 0.1$; don't care probability $= 0.33$; sub-

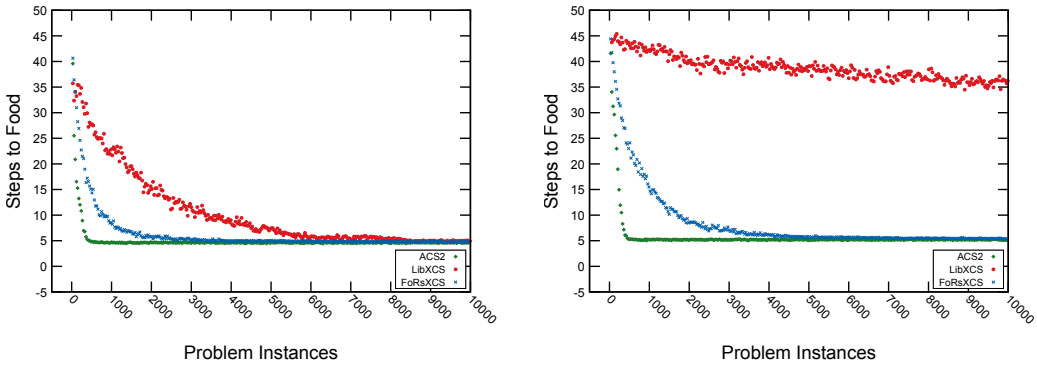
sumption threshold $\theta_{sub} = 20$; prediction reward $= 1000$; prediction error reduction $= 0.25$. The agent is randomly placed at an empty position within an environment at the start of a trial. The agent is allowed to reach the goal by utilizing a maximum of 50 steps. All the results presented here are taken from the average of 30 runs.

The experimental results of the novel system (FoRsXCS) are compared with the experimental results of seven well-known benchmark systems, i.e. BACS2[309], BACS3 [309], XCSLib [331], ACS2 [241], XCSM [303], AgentP [272], and deep recurrent Q-network (DRQN) [332, 333]. DRQN is a well-known deep learning-based system that applies a connectionist strategy to solve POMDPs environments. We were able to reproduce the experimental performance for XCSLib, ACS2, and DRQN. The results for BACS2 and BACS3 have been kindly shared by the authors. However, for the other systems (XCSM and AgentP), I have used the results reported in the respective studies.

7.4 Experiments

The first set of experiments was conducted for deterministic mazes (Maze5 and Maze6) to provide proof-of-concept for the developed FoRsXCS system. For the Maze5, ACS2 and DRQN outperformed FoRsXCS by 0.18 and 0.13 steps, respectively. But for Maze6, ACS2 outperformed FoRsXCS by 0.25 steps but the performance of FoRsXCS is better than all other systems including DRQN. It is noted that the learning rate of the novel system is slower than ACS2 (see Fig. 7.7). This is understandable because the novel system has to create heterogeneous BBKs to create the local and world viewpoints of the environment. Thus the FoRXCS may be computationally inefficient for solving simple deterministic mazes.

The second set of experiments was conducted for non-Markov mazes to evaluate the effectiveness of the novel approach. A comparison of experimental results with different state-of-the-art studies is presented in Ta-



(a) Maze5 (b) Maze6

Figure 7.7: Experimental results of Maze5 and Maze6 using ACS2, XCSLib, and FoRs-based XCS (FoRsXCS)

ble 7.1. The experimental results show that the novel FoRsXCS system either outperformed all other systems or showed similar behavior in solving all the non-Markov mazes except Maze7. For Maze7, AgentP and BACS3 outperformed the novel system by 0.03 and 0.01 steps, respectively.

The experimental results show that the novel FoRsXCS system effectively solves complex aliasing patterns in mazes that have been most challenging to artificial agents. For example, the novel system utilizes 6.51, 3.71, and 3.22 steps to resolve Maze10, Littman57, and Woods102, respectively. In contrast, none of the existing systems behave effectively in all these mazes. The minimum steps required by the best existing systems are 7.87 (AgentP), 4.52 (BACS2), and 3.30 (AgentP) to solve Maze10, Littman57, and Woods102, respectively. It is important to note that the well-known DRQN successfully solves deterministic and non-Markov mazes but the novel FoRsXCS outperformed DRQN in all the mazes. The reasons for the performance efficiency of the novel FoRsXCS are explained in Section 7.5. The varied learning pace of the novel system for different mazes is presented in the supplementary material (see Section S-IV).

The Wilcoxon signed-rank test was applied to statistically compare FoRsXCS with DRQN (see Table 7.2). The test was conducted on the results of the last 100 trials. The second and third columns contain the average performance along with standard deviation. FoRXCS statistically out-performed DRQN on all mazes, all $p's < 0.00001$, which is evidence that the performance advantage of FoRsXCS is statistically significant.

The novel FoRsXCS can utilize multiple viewpoints at different levels of abstraction, depending on the complexity of aliased patterns in the environment. This functionality adds extra computational cost. Although it is not straightforward to compare the computational cost for different systems due to operating system constraints, these systems can be compared based on the average processing time required for an agent to take a step in an environment. The average single-step processing times, computed by using Maze7, for FoRsXCS and XCSLib are 326.37μSec and 74.56μSec, respectively. The processing time for the novel FoRsXCS is 4.4 times longer than the XCSLib. However, this cost is justified because the FoRsXCS needs on average 4.3 steps to successfully reach the goal in Maze7, whereas the XCSLib needs 31 steps. Thus, XCSLib utilizes 7.2 times more steps as compared to FoRsXCS. This shows that the overall computational cost of the XCSLib is greater than that of the FoRsXCS. Furthermore, the FoRsXCS based agent successfully reached the goal in all trials, whereas the XCSLib based agent did not always reach the goal.

7.5 Experimental Analysis

The learning process of the FoRs based system is interpretable. Close observation of CPs and the ASM reveals that the novel system successfully identified and disambiguated complex patterns of aliased states by utilizing the relevant BBKs at different levels of abstraction. Consequently, the novel system translated a non-Markov environment into a Markov environment.

Table 7.1: Performance accuracy of LCS agents
Lower is better (Best performance is in bold).

	XCSM	AgentP	BACS2	BACS3	XCSLib	ACS2	DRQN	FoRsXCS
Maze5	-	-	-	-	5.18 ± 0.31	**4.61 ± 0.06**	4.66 ± 0.18	4.79 ± 0.07
Maze6	-	-	-	-	36.48 ± 0.87	**5.20 ± 0.05**	8.07 ± 1.12	5.45 ± 0.06
Maze7	10.0	**4.3**	4.34 ± 0.07	4.32 ± 0.06	31.73 ± 0.75	26.20 ± 1.30	4.45 ± 0.18	4.33 ± 0.05
Maze10	35.0	7.87 ± 7.43	26.09 ± 17.50	8.17 ± 1.36	40.44 ± 0.64	36.27 ± 0.81	9.75 ± 0.75	**6.51 ± 0.21**
MazeB	-	-	4.0 ± 0.17	4.09 ± 0.17	4.31 ± 0.05	4.73 ± 0.19	3.80 ± 0.13	**3.51 ± 0.04**
MazeD	-	-	3.0 ± 0.20	2.87 ± 0.11	2.88 ± 0.03	2.77 ± 0.03	2.83 ± 0.15	**2.74 ± 0.03**
MazeF1	-	-	-	-	1.80 ± 0.0	1.80 ± 0.02	1.80 ± 0.08	**1.79 ± 0.02**
MazeF2	-	-	-	-	2.50 ± 0.0	**2.49 ± 3.37**	3.01 ± 0.13	**2.49 ± 0.03**
MazeF3	-	-	-	-	**3.37 ± 0.01**	**3.37 ± 0.04**	3.62 ± 0.14	**3.37 ± 0.04**
MazeF4	-	4.5	4.51 ± 0.09	**4.49 ± 0.09**	33.63 ± 0.84	28.16 ± 1.12	5.06 ± 0.16	**4.49 ± 0.06**
Littman57	-	4.82 ± 5.41	4.52 ± 0.33	4.79 ± 0.49	11.95 ± 0.74	15.84 ± 1.35	5.60 ± 0.34	**3.71 ± 0.08**
Littman89	-	3.8	4.29 ± 0.21	4.29 ± 0.19	6.07 ± 0.25	8.34 ± 0.60	6.13 ± 0.29	**3.78 ± 0.05**
Woods101	3.2	**2.9**	3.03 ± 0.08	3.02 ± 0.07	8.04 ± 0.56	8.48 ± 0.35	3.31 ± 0.13	**2.90 ± 0.02**
Woods101-$\frac{1}{2}$	-	**3.1**	3.23 ± 0.20	3.20 ± 0.15	29.94 ± 0.60	22.95 ± 0.75	3.73 ± 0.15	**3.10 ± 0.02**
Woods102	3.5	3.3	3.73 ± 0.24	3.85 ± 0.23	28.97 ± 0.98	13.07 ± 0.50	3.49 ± 0.11	**3.22 ± 0.03**

Table 7.2: Wilcoxon signed-rank test

Problem Domain	DRQN	FoRsXCS	Z-Value	P-Value
Littman57	5.60 ± 0.34	3.71 ± 0.08	-5.5109	<0.00001
Maze10	9.75 ± 0.75	6.51 ± 0.21	-5.5109	<0.00001
Woods102	3.49 ± 0.11	3.22 ± 0.03	-5.5109	<0.00001

Littman57 provides a non-Markov environment with moderate aliasing complexity. The maze and its aliased states (each state has a unique color) are shown in Fig. 7.8. The learning sequence of the novel system to resolve Littman57 is shown in Fig. 7.9. At the start of the learning process, all the states have default version 0. Initially, the agent was randomly placed in state S_8. It executed action $A7$ to transit to state S_3. Consequently, the the agent created CPs which provide connections between "S_{8,V_0} & S_{3,V_0}", i.e. $S_{8,V_0} \xrightarrow{\text{A7}} S_{3,V_0}$, $S_{3,V_0} \xrightarrow{\text{A3}} S_{8,V_0}$. Subsequently, the agent executed action $A6$ to transit to state S_2. Consequently, the agent created CPs which provide connections between "S_{3,V_0} & S_{2,V_0}", i.e. $S_{3,V_0} \xrightarrow{\text{A6}} S_{2,V_0}$, $S_{2,V_0} \xrightarrow{\text{A2}} S_{3,V_0}$. These CPs form constituent BBKs and provide the egocentric view. Subsequently, the agent created two holistic level CPs, which provide connections among "S_{8,V_0}, S_{3,V_0} and S_{2,V_0}", i.e. $S_{8,V_0} \xrightarrow{\text{A7}} S_{3,V_0} \xrightarrow{\text{A6}} S_{2,V_0}$ and $S_{2,V_0} \xrightarrow{\text{A2}} S_{3,V_0} \xrightarrow{\text{A3}} S_{8,V_0}$. These CPs form holistic BBKs and provide an abstract view. However, there are two such blocks in the maze, represented by purple and red dotted lines. The agent could not differentiate between these blocks. The relevant information is updated in the ASM, see Fig. 7.9-a.

The agent moved from S_{2,V_0} to S_{3,V_0} by executing an action $A2$. There is no change/update in BBKs because a CP already exists for this move, i.e. $S_{2,V_0} \xrightarrow{\text{A2}} S_{3,V_0}$. In the next step, the agent moved from S_{3,V_0} to S_{2,V_0} by executing an action $A2$. The agent successfully identified that the current state S_{2,V_0} is an aliased state because of the existing CP $S_{3,V_0} \xrightarrow{\text{A6}} S_{2,V_0}$ (i.e. holistic view). Consequently, the agent disambiguated the aliased state

S_{2,V_0} into two unique states, i.e. S_{2,V_1}, and S_{2,V_2}. The corresponding BBKs, with same or flipped actions, are updated with S_{2,V_1}, e.g. $S_{3,V_0} \xrightarrow{A6} S_{2,V_1}$. Two new constituent level CPs are created, i.e. $S_{3,V_0} \xrightarrow{A2} S_{2,V_2}$, and $S_{2,V_2} \xrightarrow{A6} S_{3,V_0}$. The holistic level CPs after this move are, $S_{2,V_1} \xrightarrow{A2} S_{3,V_0} \xrightarrow{A2} S_{2,V_2}$, and $S_{2,V_2} \xrightarrow{A6} S_{3,V_0} \xrightarrow{A6} S_{2,V_1}$. It is important to note that the agent may consider the central state S_2 as V_1 or V_2 depending on the associated block. Moreover, the state S_{8,V_0} is in the common neighborhood of states S_{3,V_0} & S_{2,V_2} and a CP exists from S_{3,V_0} to S_{8,V_0}, i.e. $S_{3,V_0} \xrightarrow{A3} S_{8,V_0}$. Consequently, a connection is created from state S_{2,V_2} to S_{8,V_0} in the ASM, see Fig. 7.9-b.

0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	1	0	1	0	1	0	F	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0	0	0	0	0	0
0	S_0	S_1	S_2	S_3	S_2	S_3	S_2	S_4	S_5	S_6	S_7	0
0	0	0	S_8	0	S_8	0	S_8	0	S_8	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0

S_0=00100000 S_1=00110010 S_2=00101010 S_3=00110110 S_4=001F0110
S_5=0010F010 S_6=00100F10 S_7=00100010 S_8=11000001

Figure 7.8: Maze Littman57, 1 empty, 0 blocked, F food/goal.

The agent moved from S_{2,V_2} to S_{3,V_0} by executing an action $A2$. The agent successfully identified that the current state S_{3,V_0} is an aliased state because of the existing CPs $S_{2,V_2} \xrightarrow{A6} S_{3,V_0}$ and $S_{3,V_0} \xrightarrow{A6} S_{2,V_1}$ (i.e. holistic view). Consequently, the agent disambiguated the aliased state S_{3,V_0}

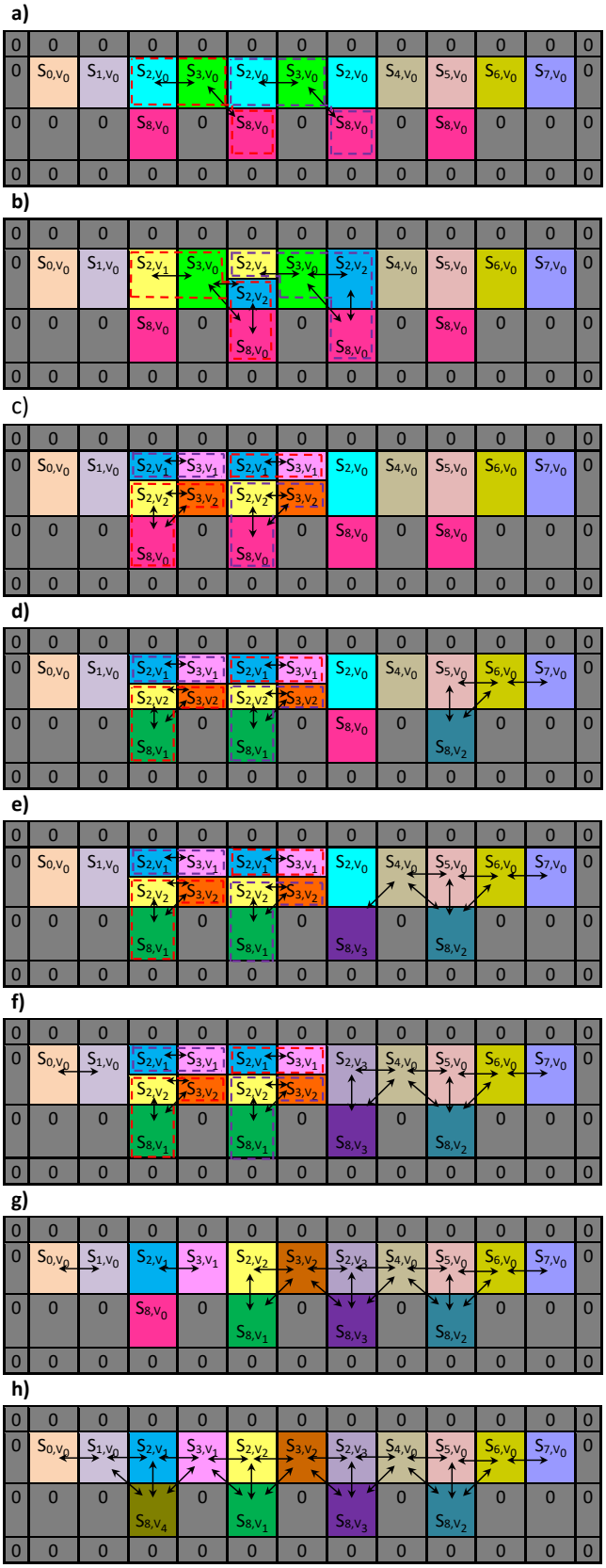


Figure 7.9: Learning sequence for maze Littman57, here S_{i,V_j} are coloured differently.

into two unique states, i.e. S_{3,V_1}, and S_{3,V_2}. The corresponding BBKs are update with S_{3,V_1}, i.e. $S_{3,V_0} \xrightarrow{\text{A6}} S_{2,V_1}$, and $S_{2,V_1} \xrightarrow{\text{A2}} S_{3,V_1}$. All other BBKs are removed or cleared (disconnected). Two new constituent level CPs are created, i.e. $S_{2,V_2} \xrightarrow{\text{A2}} S_{3,V_2}$, and $S_{3,V_2} \xrightarrow{\text{A6}} S_{2,V_2}$, see Fig. 7.9-c.

The agent moved from S_{7,V_0} to S_{6,V_0} to S_{5,V_0} and created the corresponding constituent and holistic CPs. Subsequently, the agent moved from S_{5,V_0} to S_{8,V_0} by executing an action $A4$. The agent successfully identified that the current state S_{8,V_0} is an aliased state because of the existing CPs $S_{2,V_2} \xrightarrow{\text{A4}} S_{8,V_0}$ and $S_{8,V_0} \xrightarrow{\text{A0}} S_{2,V_2}$ (i.e. holistic view). Consequently, the agent disambiguated the aliased state S_{8,V_0} into two unique states, i.e. S_{8,V_1}, and S_{8,V_2}. The corresponding BBKs are update with S_{8,V_1}, i.e. $S_{2,V_2} \xrightarrow{\text{A4}} S_{8,V_1}$, and $S_{8,V_1} \xrightarrow{\text{A0}} S_{2,V_2}$. Two new constituent level CPs are created, i.e. $S_{5,V_0} \xrightarrow{\text{A4}} S_{8,V_2}$, and $S_{8,V_2} \xrightarrow{\text{A0}} S_{5,V_0}$. Moreover, the common neighborhood connections are created or updated, see Fig. 7.9-d.

The agent moved from S_{5,V_0} to S_{4,V_0} by executing an action $A6$. The corresponding CP is created. Subsequently, the agent moved from S_{4,V_0} to S_{8,V_0} by executing an action $A5$. The agent successfully identified that the current state S_{8,V_0} is an aliased state because of the existing CPs $S_{3,V_2} \xrightarrow{\text{A5}} S_{8,V_1}$ and $S_{6,V_0} \xrightarrow{\text{A5}} S_{8,V_2}$ (i.e. holistic view). The maximum aliased versions for state S_8 are two and two constituent level CPs exist with the same action $A5$. The agent considered it another version of state S_8 and disambiguated it by assigning a new version, i.e. V_3. Two new constituent level CPs are created, i.e. $S_{4,V_0} \xrightarrow{\text{A5}} S_{8,V_3}$, and $S_{8,V_3} \xrightarrow{\text{A1}} S_{4,V_0}$, see Fig. 7.9-e.

The agent moved from S_{4,V_0} to S_{2,V_0} by executing an action $A6$. The agent successfully identified that the current state S_{2,V_0} is an aliased state because of the existing CPs $S_{3,V_1} \xrightarrow{\text{A6}} S_{2,V_1}$ and $S_{3,V_2} \xrightarrow{\text{A6}} S_{2,V_2}$ (i.e. holistic view). The maximum aliased versions for state S_2 are two and two constituent level CPs exist with the same action $A6$. The agent considered it another version of state S_2 and disambiguated it by assigning a new version, i.e. V_3. Two new constituent level CPs are created, i.e. $S_{4,V_0} \xrightarrow{\text{A6}} S_{2,V_3}$, and $S_{2,V_3} \xrightarrow{\text{A2}} S_{4,V_0}$. Subsequently, the agent explores so other correspond-

ing CPs are created, see Fig. 7.9-f.

The agent was again randomly placed in state S_8. It successfully reached the goal state by executing a series of actions, i.e. $A0, A2, A2, A2, A3$. The agent virtually traversed the path from the goal state to the start state, i.e. in reverse order by utilizing flipped actions. The agent confidently identified the CPs from the goal state to two next states because of the non-aliased state S_{4,V_0}. Whereas the agent was not confident about the remaining three states. The remaining reverse path is "$S_{3,V_*} - A6 - S_{2,V_*} - A4 - S_{8,V_*}$" or the original path is "$S_{8,V_*} - A0 - S_{2,V_*} - A2 - S_{3,V_*}$", here V_* indicates that the agent did not need to consider the versions. The agent searched the isolated clusters of states, by utilizing multi-states CPs and ASM, that matched these states but are totally disconnected from other neighboring states. The agent successfully found such a block of states and linked it with the next confirmed state. Consequently, two new constituent level CPs are created, i.e. $S_{3,V_2} \xrightarrow{\text{A2}} S_{2,V_3}$, and $S_{2,V_3} \xrightarrow{\text{A6}} S_{3,V_2}$, see Fig. 7.9-g.

The agent moved from S_{2,V_2} to S_{3,V_*} by executing an action $A6$. The agent identified that the current state is S_{3,V_1} because a CP with the same action existed for $S3$ (i.e. $S_{2,V_3} \xrightarrow{\text{A6}} S_{3,V_2}$) and maximum number of aliased states are two. Consequently, two new constituent level CPs are created, i.e. $S_{2,V_2} \xrightarrow{\text{A6}} S_{3,V_1}$, and $S_{3,V_1} \xrightarrow{\text{A2}} S_{2,V_2}$. Subsequently, the agent moved from S_{3,V_1} to S_{8,V_*} by executing action $A5$. The agent successfully identified that the current state S_{8,V_*} is an aliased state because of the existing CPs $S_{3,V_2} \xrightarrow{\text{A5}} S_{8,V_1}$, $S_{4,V_0} \xrightarrow{\text{A5}} S_{8,V_3}$ and $S_{6,V_0} \xrightarrow{\text{A5}} S_{8,V_2}$ (i.e. holistic view). The maximum aliased versions for state S_8 are three and three constituent level CPs exist with the same action $A5$. The agent considered it another version of state S_8 and disambiguated it by assigning a new version, i.e. V_4. Two new constituent level CPs are created, i.e. $S_{3,V_1} \xrightarrow{\text{A5}} S_{8,V_4}$, and $S_{8,V_4} \xrightarrow{\text{A1}} S_{3,V_1}$. Subsequently, the agent continues to traverse the maze such that any outstanding CPs are created. Ultimately, the agent has successfully transformed the non-Markov environment into a Markov environment. A

map of the final environment (without any ambiguous aliased states, i.e. Markov environment) is shown in Fig. 7.9-h.

7.6 Discussion

The ability to consider an input signal at a constituent level and holistic level simultaneously is a critical feature of the lateralized architecture. But this consideration not necessarily to be left/right, for navigation problems it tended to be FoRs based consideration of the same environmental signal from the local viewpoint (constituent level) and world viewpoint (holistic level). An in-depth analysis of the FoRs based lateralized system is presented in Chapter 8.

The FoRs-based system is designed to address the perceptual aliasing problem in non-Markov environments. The novel system simultaneously considers the environmental instance from a local viewpoint (single-step CPs, egocentric FoR) and world viewpoint (multi-step CPs, policies; allocentric and route-centric FoRs). Consequently, the learning speed of the novel system is slower than that of other state-of-the-art systems for large scale deterministic mazes. However, the ability to consider the same problem instance at multiple viewpoints empowers the novel system to efficiently learn the complex patterns of aliased states that characterise non-Markov environments. As the problem scales in size and complexity, there will be more and more constituent-level and holistic-level BBKs, which will slow down learning. Nevertheless, the novel system has the ability to identify and disambiguate the clusters of aliased states by utilizing the BBKs at different levels of abstraction. Consequently, the novel system efficiently solves complex non-Markov mazes that homogeneous systems struggle to solve.

The per-step computational cost of the novel system is 4.4 times higher than the conventional AI system. However, the average number of steps required by the lateralized agent to successfully reach the goal state is 7.2

times less than the conventional AI agent. Therefore, the overall computational cost of the lateralized system is less than the conventional AI system. Moreover, the lateralized agent successfully reached the goal state in all trials but this is not true for conventional AI agents.

For this work, it is assumed that the actions always correctly affect the environment. If not, LCSs do have an error threshold that can be used to handle noise. This problem is beyond the scope of this work, but is the subject of future work. Moreover, the assumption that the agent has the freedom to explore, (e.g. flipped actions), may not be reasonable in practical situations, e.g. driving on one-way roads. The FoRs-based approach may not work well for problems, such as numerical optimization, in which constituent knowledge cannot be used or reused to solve higher-level problem components.

Although it is expensive to learn constituent-level BBKs, once learned, they can be used or reused to form holistic level BBKs. The novel system applies these learned BBKs at different levels of abstraction to solve heterogeneous patterns in complex problems.

7.7 Chapter Summary

The novel system successfully applied FoRs to learn stable policies for multi-step tasks in non-Markov environments. The ability to represent the same environmental instance from different viewpoints, i.e. local viewpoint (single-step CPs, egocentric FoR) and world viewpoint (multi-step CPs, policies; allocentric and route-centric FoRs), empowers the novel system to successfully address perceptual aliasing problems by identifying and disambiguating aliasing patterns. Consequently, the novel system transforms a non-Markov environment into a deterministic environment. EC played a critical role by enabling the novel system to evolve fitter rules at a constituent level and optimal policies at a holistic level. Otherwise, it was not practical to enumerate the huge search space of a complex non-

Markov environment. The experiments demonstrate that the novel system has the ability to utilize or re-utilize relevant learned BBKs at different levels of abstraction to learn aliasing patterns that are made up of patterns that are made up of features. The novel system effectively solves complex aliasing patterns in the environments that have previously been challenging to artificial agents. For example, the novel system utilizes only 6.5, 3.71, and 3.22 steps to resolve Maze10, Littman57, and Woods102, respectively.

The novel system is robust against aliasing states because of its focus on the appropriate parts of the reward signal to achieve a necessary level of abstraction. Aliasing challenges existing evolutionary computing systems across a wide range of problem domains. How this approach functions with dynamic states, especially in domains with little information to start forming code paths, can now be investigated. Although, the lateralized framework has been adapted to develop lateralized systems for a wide range of problem domains and a number of aspects of the novel lateralized approach have been investigated, more deep and thorough investigation is needed. The initial results are promising and show the potential of the lateralized approach to tackle different problems from different domains.

8

Discussions

In this thesis, lateralization has been successfully applied in artificial intelligence systems to solve complex problems in three different domains. A wide range of experiments have been conducted to evaluate the robustness and effectiveness of the lateralized approach. The experimental results demonstrate that the lateralized systems outperform state-of-the-art non-lateralized systems in resolving complex problems. It is considered that the advantages arise from the ability of lateralized systems to, (i) represent an input signal at both the constituent level and holistic level simultaneously, such that the most appropriate viewpoint controls the system; and (ii) avoid extraneous computations by generating excitatory and inhibitory signals between modules. The lateralized approach may resemble some other artificial intelligence approaches. However, a critical analysis reveals that the novel lateralized approach is substantially different from other approaches.

In cognitive neuroscience, lateralization has been associated with

> *both poor and good performance, making the relationship between lateralization and performance ambiguous. It is anticipated that investigating lateralization in artificial intelligence can provide insight into the benefits and costs of lateralization for agents addressing complex tasks.*

The goals of this chapter are three-fold: first, to discuss the lateralized framework, its adaptation/customization for different artificial intelligence systems that can solve various tasks; second, to provide critical analysis of the state-of-the-art artificial intelligence approaches that resemble the novel lateralized approach; and third, to highlight lateralization in an agent's decision making to obtain evidence of benefits/costs from artificial intelligence in order to inform cognitive neuroscience.

8.1 Lateralized Systems

A lateralized framework has been created that encompasses all the key aspects of lateralized artificial intelligence (AI) systems. As with standard models of mind [171], the novel lateralized framework has been utilized as a coherent basis for creating three different lateralized systems (see Chapters 5, 6, and 7). Instead of creating each lateralized system from scratch, the components of the lateralized framework have been adapted, at different levels of abstraction/scales of problems, depending on the nature and complexity of the problem at hand. Consequently, each lateralized system has its own approach for solving a given problem, whereas, the lateralized framework serves as a shared ontology. A schematic illustration of the lateralized framework and its adaptation for different AI systems is shown in Fig. 8.1.

It is not the intention of this thesis to create a novel lateralized framework, or even a lateralized system, in which all of the details of the vertebrate intelligence are modelled. Rather this thesis takes inspiration from

Figure 8.1: A schematic illustration of the lateralized framework and its adaptation for different AI systems. (color key: constituent, holistic, and mix knowledge proceedings are represented by purple-white, pink-white, and light purple-pink gradients, respectively)

basic principles of lateralization that are fundamental to vertebrate intelligence. The first and most important principle of a lateralized AI system is the ability to simultaneously consider the given input signal at a constituent level and holistic level. The lateralized framework contains the *left-half* system module that considers the given signal at a constituent level and the *right-half* system module that simultaneously considers that signal at a holistic level. These terms are used because in right-handed people the left hemisphere processes information at a constituent level while the right hemisphere processes the same information at a holistic level, although individual differences can be observed in both the direction and degree of lateralization. It is critical for a lateralized AI system to have the ability to simultaneously process the input signal at a constituent level and holistic level. The left-half and right-half are just terminologies to represent the system modules, where the terms could be swapped or altered.

The processing of the same input signal at different levels of abstraction can be done once or multiple times, depending on the nature and complexity of the problem domain. For example, the lateralized AI system developed for the Boolean problems considers a given problem at a constituent level (LHSM module) and holistic level (RHSM module) only once (see Chapter 5). Whereas, it is necessary for the lateralized AI system, developed for computer vision (CV) problems, to consider the given input signal at a constituent level and holistic level, initially at the context phase, and then again at the attention phase (see Chapter 6).

The ability to consider the same problem at different levels of abstraction is one of the main features of lateralization. The left/right (or right/left) representation is not the only way to consider an input signal at a constituent level and a holistic level. A heterogeneous representation of knowledge could also be used to consider an input signal at different levels of abstraction. It is important to note that lateralization is a special type of heterogeneity. The lateralized AI systems were successfully developed, for

Boolean and CV problems, by using the left/right representation of an input signal. However, during the development of the lateralized AI system for navigation problems, it was observed that the left/right terminology was not quite appropriate for this task. In navigation tasks, lateralization does not tend to be left/right separated but it tended to be frame-of-reference (FoR) based constituent level (local viewpoint) and holistic level (world map) representation. However, it does not prevent the underlying notion of lateralization from being a reasonable way of describing the AI systems. For this thesis, lateralization is used as an inspiration that does not replicate biological hemispheres, in the similar way that an artificial neural network is a neural network although it does not exactly replicate biological neural networks.

The left-half and right-half modules do not depend on each other. They neither assist each other during the processing nor directly share the outcome of the processing. However, these modules can indirectly affect each other's processing, e.g. based on the feedback from the right-half, the resolve problem module can generate an inhibit signal for the left-half to stop further processing. Moreover, the left-half and right-half modules can communicate with the same system components (e.g. Resolution component, knowledge pool) and can utilize the shared resources (e.g. input signal, BBKs).

The first step to create a lateralized AI system is to consider a given problem at different levels of abstraction but this is not sufficient to create a lateralized system. A lateralized AI system needs to have excite/inhibit signals that can be used to achieve goal-driven processing. These signals assist the lateralized system to automatically identify and utilize the most suitable system module that can efficiently solve the given problem. These signals are implemented as a Boolean yes/no that can excite/inhibit the whole system module. There is no partially inhibit/excite signal that can allow a partial utilization of a particular system module.

The excite/inhibit signals can be implemented to automatically (with-

out human-in-the-loop) permit/prohibit the use of different system modules. For example, the lateralized AI system for Boolean problems uses excite/inhibit signals to continue/stop the processing of the RHSM module; the lateralized AI system for CV problems uses excite/inhibit signals to continue/stop the processing of the attention phase. These excite/inhibit signals can also be used to activate/deactivate system components that can control the decision making of the system. For example, the lateralized AI system for navigation problems uses excite/inhibit signals to activate/deactivate policies that can guide the agent to optimally reach the goal state.

The *Resolution* component is another important part of a lateralized system. This component analyses the feedback received from the left-half and right-half modules. The analysis can be performed once or multiple times, depending on the nature and complexity of the problem domain. Subsequently, the Resolution component assists the lateralized system to decide whether the current knowledge is sufficient, or further learning is required, to solve the given problem. The Resolution component is an essential part of all the lateralized systems; in some cases, it is very obvious in schematic figures (see Chapter 5); in other cases, it is part of the solution (see Chapters 6 and 7). For example, the lateralized system for Boolean problems has an explicit component named *Resolution*. This component analyses the feedback from both the LHSM and RHSM modules and decides new learning is needed or not for the given problem. The lateralized system for CV problem does not have a component named *Resolution*, but the system analyses the feedback from the constituent and holistic modules and decides whether the current knowledge is sufficient or further analysis is required to confidently resolve the given problem.

8.2 Relevant Approaches

This section provides a critical analysis of existing state-of-the-art approaches that resemble the novel lateralized approach.

8.2.1 Ensemble Systems

The novel lateralized systems may be confused with ensemble systems. A rigorous analysis reveals that the lateralization approach is essentially different from the ensemble approach. The majority of the ensemble systems are computationally expensive. These systems obtain an accurate prediction, by using multiple constituent learning algorithms, which could be obtained by using the single best constituent algorithm [334, 335, 336]. This approach requires many computations that do not contribute to the solution. The novel lateralized approach avoids extraneous computations by utilizing inhibit/excite signals to stop the processing of the system components that are irrelevant to the given problem domain.

The number of constituent algorithms has a great impact on the overall prediction accuracy of an ensemble system. Different techniques have been developed to determine the appropriate ensemble size but still, it is a debatable topic [337, 338]. Whereas, a lateralized system has the ability to automatically identify those BBKs (from the heterogeneous knowledge pool) that are relevant and appropriate to solve a given problem. In the novel lateralized approach, the coordination to determine the best BBKs at any abstraction level to address a problem instance is different from an ensemble approach that seeks to aggregate BBKs. This is the fundamental difference from an ensemble or cooperative coevolution system where all sub-problems act simultaneously and often inefficiently as they are ignored, redundant, or irrelevant.

It is important to note that there is no guarantee that the performance accuracy of an ensemble system is always better than the performance accuracy of the best constituent algorithm [339]. On the other hand, if

a lateralized system finds a holistic level BBK that has the ability to independently resolve the given problem, it generates an inhibit signal to stop further processing of other system components and utilizes only that holistic BBK to solve the future problem instances. Consequently, the performance accuracy of a lateralized system is always better than or equal to the performance accuracy of an individual BBK.

An ensemble system is more effective if its constituent algorithms exhibit diversity among themselves. Consequently, heterogeneous features are required to train such an ensemble system. Moreover, a constituent algorithm needs to be trained by using a separate (relevant) subset of these heterogeneous features [340]. Whereas, in a lateralized system, diversity depends on the nature and complexity of the problem at hand. For example, the lateralized learning of a $4-$bit Parity problem can utilize the learned concepts of $2-$bit and $3-$bit Parity problems (homogeneous problem domain); whereas, the lateralized learning of an $18-$bit hierarchical multiplexer problem can utilize the learned concepts of $3-$bit Parity and $6-$bit multiplexer problem (heterogeneous problem domain).

The splitting and processing of the input signal into a constituent level and holistic level in a lateralized system may make it an ensemble-like system. However, the ability to automatically (without human-in-the-loop) consider the same input signal at different levels of abstraction makes it a lateralized system rather than an ensemble system. For example, a lateralized Boolean system considers an input signal of a $6-$bit multiplexer problem at a single level, whereas, it considers an input signal of a $18-$bit hierarchical multiplexer problem at two levels of abstraction.

8.2.2 Deep Learning

Deep learning (DL) is a state-of-the-art approach that has been used to solve many real-world problems. The majority of DL-based systems store knowledge in multiple layers such that all features are treated equally in each layer. These systems generate a homogeneous knowledge represen-

tation that cannot be reused elsewhere in the system [9]. Consequently, these systems generate a huge/deep network of homogeneous knowledge to learn complex (hierarchical) problems and do not take advantage of the potential to transfer knowledge between levels in the hierarchy. It is important to note that transfer learning takes whole sections from one system and applies it to another system. But that is not (re)using information multiple times in the same system. In contrast, a lateralized system generates a heterogeneous knowledge representation at different levels of abstraction. It has the ability to automatically identify relevant BBKs from the heterogeneous knowledge pool and (re)utilize them at the constituent level or holistic level. A holistic level BBK learned for a lower level problem, could be used/reused as a constituent level BBK for a higher-level problem within the same system or across different systems.

The majority of DL-based systems exhibit poor robustness against noisy, irrelevant, and redundant data [52, 55, 56]. This is due to their reliance on homogeneous knowledge representation. Even a single targeted pattern can disrupt performance accuracy. Whereas, the novel lateralized systems can exhibit strong robustness against noisy, irrelevant, and redundant data. This is due to their ability to consider the same problem at different levels of abstraction (i.e. constituent level and holistic level). Not only the holistic level but also the constituent level components of a problem must be successfully challenged to fool a lateralized system.

Recently, capsule networks have been introduced to address the deficiencies of deep networks and to better model hierarchical relationships [18]. But the routing algorithm of capsule networks cannot differentiate vectors from their negative counterparts. Thus capsule networks can only learn if both the negative input and the original input represent the same class. Consequently, capsule networks are not suitable for learning some concrete but simple problems [341]. The majority of the capsule networks based systems have been applied to solve image recognition problems. Moreover, these systems fail to exhibit robustness against adversarial at-

tacks [342].

8.2.3 Granular Computing

Granular computing is an emerging methodology that has been applied to recognize regularities, present at different levels of abstraction, in the data [343]. This approach makes effective use of granular structure to represent the same problem at different levels of abstraction and different viewpoints [344]. A granular structure has three key elements, i.e. granules, levels, and hierarchies [343, 345, 346, 347].

Granules are an important part of the granular structure. They facilitate knowledge representation and processing. They can be considered as elements, units, concepts, and notions that are required for interpreting, representing, and processing a problem. For example, words, sentences, paragraphs, and articles are granules in a text processing problem. Granules can also exist in the form of groups, clusters, or sets of ideas [348].

The degree of abstraction (known as granularity, or size of granules) is an important feature of granules. A level of abstraction consists of granules that have similar nature or granularity. However, the novel lateralized approach allows integration of heterogeneous BBKs to solve a specific level of abstraction (see Chapters 6 and 7).

In order to solve a complex hierarchical problem, granules form a hierarchy of levels that are ordered based on their granularity [348]. The granularity level plays a critical role in creating a strategy to solve such a problem. However, it does not have a universal value; it is user-dependent and problem-oriented [344]. In granular computing, it is essential to identify an effective level of granularity concerning the given problem. Attribute reduction and feature selection studies have been used to find an appropriate level of granularity [348, 349]. Moreover, it is also important to identify the appropriate number of granules that can be used to provide a solution. Several studies have been conducted to address these problems but still, they are debatable topics [346, 348, 350].

In contrast, the novel lateralized approach has the ability to consider a given problem at multiple levels of abstraction, without human-in-the-loop. For each level, the relevant constituent and holistic level BBKs are identified. Subsequently, these BBKs are utilized to solve a part of the problem or the whole problem (see Chapters 5 and 7).

A hierarchical granular structure may consist of multiple levels. A hierarchy represents a problem from a particular viewpoint. It consists of granules that focus only on one aspect of perception, hence, can provide a solution only for a specific problem [344]. In the majority of the cases, a hierarchical representation can only be used for the intended problem and cannot be used for other problems [348]. To overcome this limitation, many hierarchies are created to serve multiple purposes. This makes granular computing a multilevel and multiview, but computationally expensive approach.

The novel lateralized approach has the ability to consider a given problem from multiple perspectives. This functionality is facilitated by utilizing the heterogeneous BBKs from the knowledge pool. For example, the novel lateralized system for the Boolean problem considered the given $18-$bit hierarchical multiplexer problem from four different perspectives (see Chapter 5). Moreover, the lateralized approach has the ability to avoid extraneous computations by generating inhibit and excite signals.

The process of decomposing or constructing granules is called granulation. The decomposition and construction processes are correlated and they play an important role in providing solutions. Moreover, it has been observed that the relationships among granules and their relationship with granulation form complex solution patterns [344, 351]. In contrast, the lateralized approach does not require such a decomposition or construction. During the learning process, constituent level and holistic level BBKs are created, utilized, and re-utilized at multiple levels of abstraction, depending on the nature and complexity of the problem (see Chapters 5 and 7). Moreover, the relationships between different BBKs (constituent and holis-

tic level) are automatically established (see 5). Furthermore, the decision-making process of the novel lateralized systems is interpretable. This assists in understanding the relationships between different level BBKs (see Chapters 5 and 6).

Granular computing originated from fuzzy set theory and later on was highly influenced by rough set theory [347, 349, 352]. Granular computing creates relationships between objects based on similarity measures. This similarity measure has been computed by using techniques such as rough sets and fuzzy logic [353]. It has been reported that the granular computing community has major interaction with fuzzy sets and rough sets but much less interaction with other communities [344, 354]. Conversely, the novel lateralized approach is not limited to a specific problem domain. For example, the novel lateralized approach has been implemented for Boolean, computer vision, and navigation domains (see Chapters 5, 6, and 7).

8.3 Evidence of Benefits/Costs from Artificial Intelligence

Biological intelligence has been used as a major source of inspiration for creating AI systems [242, 355, 356]. Given that lateralization is ubiquitous (at least amongst vertebrates, but also in many invertebrates), it likely has advantages that can benefit artificial intelligence. In turn, AI systems can be used as tools to advance understanding of biological intelligence. These advances can be created in three main ways [357, 358]. First, AI systems can be employed as data mining tools to extract patterns from neuroscience data that are too complex for conventional statistical approaches to handle well. Second, AI can model the brain in an attempt to replicate it. Third, AI systems can be used to probe fundamental aspects of cognitive architecture. This third approach is the focus here.

The influence of hemispheric asymmetry on task performance in bio-

logical species has been the subject of much debate [160, 161, 359], and there are many inconsistencies in the empirical literature. For example, in the cognitive domain, greater lateralization has been associated with poorer performance on verbal dichotic listening tasks [360], but also better performance on mental rotation and verbal memory, fluency, and intelligence [361, 362, 363]; the personality trait schizotypy has been associated with reduced lateralization, increased lateralization, and null effects [170]. It has been hypothesized that lateralization has benefits that may counterbalance its costs [82, 158, 159], but this is difficult to test in vivo.

Individual differences in lateralization are also associated with performance in animals, but again, lateralization has both costs and benefits [82, 83]. For example, strongly lateralized fish exhibit lower spatial learning and perform poorly in tasks that require simultaneously matching information from both hemispheres [364, 365]. However, lateralized fish exhibit stronger responses to a predator as compared to the responses from non-lateralized fish. But lateralized fish are poor competitors as compared to non-lateralized fish when interacting over a shelter resource, e.g. they show few displays, less attack, and exhibit avoidance [82]. Again the relationship between lateralization and cognitive enhancement is ambiguous.

These inconsistent findings may reflect poor methodology (small sample sizes, noisy data). But they may also indicate that lateralization has both costs and benefits, and so the association between lateralization and performance will depend heavily on whether specific task parameters are biased towards benefits or costs. Lateralised AI models could therefore be very useful in testing the trade-offs between costs and benefits that affect performance.

Empirical evidence regarding the relationship between performance and lateralization could be obtained from AI systems. It is anticipated that the integration of lateralization into AI systems is beneficial/advantageous in certain situations. However, it is not clear what those situations are,

what are the benefits/advantages, and what are the costs involved. This work provides such insight in the next section.

8.3.1 Lateralization in AI

The novel lateralized framework has been successfully adapted to develop lateralized AI systems for Boolean, CV, and navigation problem domains. Boolean problems are complex engineering problems which are not relevant to neuroscience community. These problems are neither real-world nor have uncertainty, therefore are not further discussed here.

One of the important areas where the effectiveness of the lateralized approach could be evaluated is a real-world domain with uncertainty, noise, irrelevant, and redundant data. The majority of AI systems do not exhibit robustness against noisy and irrelevant data. For example, deep artificial neural networks are highly vulnerable to adversarial attacks in visual classification tasks [48, 52]. A lateralized approach that considers the input image at different levels of abstractions has shown performance benefits such that the contribution of lateralization can be interrogated (see Chapter 6).

The CV problems are used to investigate the robustness of the lateralized approach against adversarial attacks, where the additional benefits are evaluated to include exciting or inhibiting the appropriate BBKs to gain computation efficiency. The adversarial images have redundant, noisy, and irrelevant data. These images proved to be challenging for state-of-the-art (non-lateralized) deep networks. In contrast, the lateralized approach enables the novel system to correctly classify such images including those that are badly affected by the strong adversarial attacks. The computational cost of a lateralized AI system is double as compared with the conventional AI system. However, this cost is justified based on the robustness exhibited by the lateralized system against adversarial attacks. It is hard to fool the novel lateralized system because it requires an adversarial attack to successfully challenge the constituents as well as the

holistic components of an image. The costs of lateralization may dominate in simple CV problems where inputs are clean and unambiguous. However, when inputs include noise and irrelevant data (as is common in the real world), the benefits of a lateralised system outweigh the computations costs.

Visual classification tasks are single-step problems in spatial domains. However, the majority of the biological tasks where heterogeneity [1] is applied, are temporal in nature. Thus another area where lateralized/heterogeneous feature-based AI systems could be evaluated is in multi-step complex problems. The majority of the AI systems struggle to capture complex structures in an environment. For example, perceptual aliasing is one of the major hindrances in applying reinforcement learning techniques in artificial agents to handle multi-steps tasks [10, 16, 57, 58, 59, 14, 60]. The perceptual aliasing problem arises when the agent's immediate perception cannot differentiate environmental states that look similar but require different actions, which lateralization seeks to handle (see Chapter 7).

The navigation problems are multi-step path planning problems that provide a virtual environment that simulates real-world navigational problems. These problems are used to investigate the effectiveness of the heterogeneous approach in resolving complex non-Markov mazes. These mazes entail hierarchical aliasing patterns that pose additional challenges for the homogeneous systems due to their reliance on local viewpoint only. Whereas, the novel approach enables the lateralized AI agent to simultaneously consider the multiple viewpoints of the given environment, the local viewpoint, and the world viewpoint. Consequently, the novel system optimally resolves the complex hierarchical patterns of aliased states that similar homogeneous systems struggle to resolve. The per-step computational cost of the novel system is 4.4 times higher than the conventional AI system. However, the average number of steps required by the lateralized agent to successfully reach the goal state is 7.2 times less than

[1] Lateralization can be considered as a special type of heterogeneity.

the conventional AI agent. Therefore, the overall computational cost of the lateralized system is less than the conventional AI system. Moreover, the lateralized agent successfully reached the goal state in all trials but this is not true for conventional AI agents.

The lateralized AI system may not work better than a conventional AI system for the optimization (or other such) problems where a given problem instance cannot be considered at different levels of abstraction. Moreover, the lateralized approach may not be advantageous if the constituent level BBKs are either unavailable or cannot be (re)utilized to resolve higher (holistic) level problems. The learning pace of a lateralized system may be slow at the start due to extra computations required to learn constituent BBKs, however, once these BBKs are learned, the lateralized system scale quickly to solve complex problems. It is considered that the lateralization is beneficial/advantageous for AI systems in certain situations but there are associated costs as well. This work may open a new door for developing AI systems that can provide insight into contentious topics in neuroscience.

8.4 Chapter Summary

The novel lateralized framework may capture community consensus and become a cumulative reference point for creating lateralized AI systems. Different elements of the lateralized framework can be adapted, not only in different ways but also at different levels of abstraction, depending on the nature and complexity of the problem domain. In biological intelligence, lateralization is a special type of heterogeneity. In artificial intelligence, the underlying lateralized framework is equally beneficial for heterogeneous problems.

The novel lateralized approach is essentially different from other resembling AI approaches. A critical comparison is presented with three state-of-the-art approaches, i.e. ensemble systems, deep learning, and gran-

ular computing. These AI methodologies may incorporate some features of the novel lateralized approach but none of them include all the features. It is important to note that a lateralized AI system needs to implement all the features of the lateralized framework, otherwise, it will not be considered as a lateralized AI system (see Chapter 4).

The benefits and costs of lateralization in cognitive inspired AI systems are successfully highlighted. Creating AI systems with lateralization/heterogeneity represents an interesting way to explore the trade-offs and costs of these approaches. The ability to consider the same input signal (problem instance) at different levels of abstraction enables the lateralized AI systems to address constituent (local) features and high-level abstract patterns (features made-up of features) simultaneously. Consequently, the lateralized systems efficiently resolve complex hierarchical patterns by identifying and utilizing the relevant constituent and holistic BBKs. The overall computational costs of the lateralized AI systems are either less than that of conventional AI systems or compensated by the associated benefits such as robustness.

9

Conclusions and Future Work

The novel lateralized approach has the ability to apply lateralization and modular learning at different levels of abstraction to solve complex problems, in different domains, that similar homogeneous system struggle to solve. Considering the same environmental signal at different levels of abstraction (constituent level and holistic level) empowers the novel lateralized system to reframe a complex problem as a simple problem and efficiently resolve it.

Single or multiple step, supervised or reinforcement learning, Boolean or real-valued features, and Markov or partially observable Markov decision processes have all been shown to benefit from the lateralized learning approach. The ability to inhibit or excite the most appropriate learning structure offers efficiencies and effectiveness beyond ensemble, co-evolutionary, or granular computing approaches. This chapter concludes the discussion of

this thesis, presents the achieved objectives, highlights the main findings, and outlines directions for future work.

The comprehensive goal of this thesis was to accomplish lateralized learning, inspired by the principles of biological intelligence, in artificially intelligent (AI) systems. This goal was successfully achieved by creating a novel lateralized framework for AI systems and adapting this framework for developing lateralized AI systems to solve complex problems in Boolean, computer vision (CV), and navigation domains. The developed methods were evaluated on a range of problem benchmarks and compared with state-of-the-art techniques. The experimental results show that the novel lateralized methods, proposed in this thesis, have achieved either competitive performance or outperformed similar non-lateralized (homogeneous) methods.

The remainder of this chapter presents the achieved objectives, highlights the main findings from each contribution chapter, and finally suggests research directions for future work.

9.1 Achieved Objectives

This following research objectives have been accomplished to achieve the overall research goal.

1. A general lateralized framework, based on the essential principles of lateralization from cognitive neuroscience, was proposed for the first time. This framework highlights the key aspects of knowledge perception, knowledge representation and utilization, and connectivity patterns. It presents the essential functionality, critical methods, and associated parameters that are required to be incorporated into a lateralized AI system. The novel framework can be adapted to create lateralized AI systems for a wide range of problem domains. For this thesis, the lateralized framework has been successfully adapted

to create lateralized AI systems for Boolean, CV, and navigation domains.

2. A novel lateralized AI system was created, by adapting the lateralized framework, for complex Boolean problems to obtain a proof-of-concept of the lateralized approach. The novel system applies reinforcement learning techniques to solve interogatable, single-step, scalable, and complex Boolean problems. Two modules (termed LHSM and RHSM) were developed to consider the same input signal at different levels of abstraction. The LHSM considers the input signal at a constituent level and generates sub-groups of features to knowledge mapping, whereas, the RHSM considers the same input signal at a holistic level and generates all features to knowledge mapping. A heterogeneous knowledge pool stores the learned building blocks of knowledge (BBKs) that are (re)used at different levels of abstraction. Finally, the Resolution component was created to analyze the feedback received from LHSM and RHSM. This component considers whether the quality of the identified knowledge is sufficient to independently solve the problem, to contribute to solution in cooperation with the other knowledge, or is irrelevant to solving the problem. Moreover, it generates excite and inhibit signals to efficiently resolve the given problem and avoid extraneous computations. The resultant system has the ability to apply lateralization and modular learning to solve complex problems. The experimental results show that the novel lateralized system outperformed state-of-the-art systems in solving complex Boolean problems.

3. A novel lateralized AI system was created, by adapting the lateralized framework, for CV problems to evaluate the robustness of the lateralized approach. The novel system applies a supervised learning technique to solve single-step visual classification tasks that include uncertainty, noise, and irrelevant and redundant data. Two

phases (Context Phase and Attention Phase) were developed to address the same visual problem at different levels of abstraction. Both the phases consider the given problem instance at a constituent level and holistic level. These phases utilize heterogeneous knowledge representation such that holistic knowledge at one level can be used as a constituent knowledge at a higher level. Finally, the context phase communicates with the attention phase through excite/inhibit signals to efficiently solve the problem and avoid extraneous computations. Another novel lateralized system (an improved version of the developed system) was created for CV problems to show that the lateralized approach can be scaled and does not rely on the use of learning classifier systems (LCSs). The experimental results show that both novel lateralized systems successfully exhibited robustness against adversarial attacks and outperformed state-of-the-art deep models.

4. A novel frame-of-reference (FoR) based AI system was created, by adapting the lateralized framework, for navigation problems to evaluate the effectiveness of the lateralized approach. The novel system applies a reinforcement learning technique to address perceptual aliasing in multi-step decision making tasks. The novel system processes a single environmental signal at different levels of abstraction to provide multiple environmental views, i.e. a local viewpoint (constituent knowledge) and a world viewpoint (holistic knowledge, complete map) of the same state. Code paths (CPs) and policies were created to provide heterogeneous knowledge representation. The learned CPs are integrated at different levels of abstraction to generate an unambiguous representation of knowledge in policies. The learned policies are activated and deactivated such that the agent can reach the goal state by using the minimum number of steps. The experimental results show that the novel FoR based system successfully resolves complex patterns of aliased states and outperformed

state-of-the-art techniques.

In addition to achieving the above-established research objectives, this thesis highlights three important aspects of artificial lateralization: first, a thorough analysis of the lateralized framework is provided. It includes how different components of the lateralized framework can be adapted (in different ways, at different levels) for different problem domains; second, a critical analysis and comparison between the novel lateralized approach and other similar AI approaches is provided; third, the use of lateralized artificial intelligence to analyse the relationship between lateralization and performance in order to inform cognitive neuroscience is explored.

9.2 Main Conclusions

Overall, this thesis finds that lateralization is effective to solve complex problems in different domains. The novel lateralized methods outperformed state-of-the-art methods in solving complex problems in Boolean, CV, and navigation domains. The main conclusions drawn from the four contribution chapters (Chapter 4 through Chapter 7) are presented and discussed below.

9.2.1 Lateralized Framework

The general lateralized framework was inspired by the principles of biological intelligence, derived from the current understanding of cognitive mechanisms in vertebrate brains. This framework provides general guidance for creating a lateralized system for a wide range of problem domains. It highlights the essential features/functionality that are required to be incorporated into an AI system to behave as a lateralized AI system.

It is concluded that a lateralized AI system needs to incorporate all the essential functionality of the lateralized framework. Lateralized AI systems can not be created by developing some functions of the lateralized

framework. If any of the important features/functions of the lateralized framework is missed, the system will not be a lateralized AI system.

The lateralized framework has been adapted to create lateralized systems for Boolean, CV, and navigation domains. This framework can be revised and extended to accommodate new findings. It is hoped that the novel lateralized framework may capture community consensus and become a cumulative reference point for creating lateralized AI systems.

The novel lateralized framework identifies the following necessary features of a lateralized AI system:

Representation and Processing

The representation and processing of the same environmental signal at different levels of abstraction (i.e. constituent level and holistic level) is a critical function of a lateralized AI system. The lateralized framework proposes a *left* half system component to process the input signal at a constituent level and a *right* half to process the same signal at a holistic level. Consequently, the left half generates elementary knowledge representation, whereas, the right half generates a more abstract knowledge representation. This feature enables the lateralized systems to simultaneously address the details of the problem and the higher level big-picture. It is essential for a lateralized AI system to have these components to simultaneously address the details of the problem and the higher level big-picture. The left half and right half are just terminologies to represent the system components. The choice of terms "left" and "right" was inspired by a large of body of research in cognitive neuroscience linking the left hemisphere with local/analytical/constituent information processing, and the right with global/synthetic/holistic information processing, although the terms are used here metaphorically, and not with the intention to model a specific lateralized process in a specific vertebrate brain.

Coordination

The coordination between different system components is an important attribute of a lateralized AI system. Different system components solve the given problem at a constituent level and holistic level. These partial solutions need to be integrated to provide a complete solution. It is concluded that the coordination between system components is necessary for the transfer of critical information and knowledge integration.

Goal-driven Processing

Goal-driven processing is another important feature of a lateralized AI system. It enables a lateralized AI system to identify and utilize those system components that are more appropriate to solve the problem at hand. For this purpose, a lateralized system uses excite/inhibit signals to activate/deactivate system components, as the goal dictates. It is concluded that goal-driven processing is essential to avoid extraneous computations and efficiently solve the given problem.

Knowledge Identification and Utilization

The lateralized framework proposes a heterogeneous knowledge pool to store the learned BBKs. A lateralized system needs to create strategies to automatically identify and utilize the relevant knowledge from the knowledge pool. The learned knowledge can be (re)used at different levels of abstraction depending on the nature and complexity of the problem. That is, a given BBK can be used by either hemispheric module, scaled to the appropriate level of abstraction. Although biological brains do not have a central storage unit, but shared heterogeneous knowledge pool is an important component of the novel lateralized approach.

9.2.2 Proof-of-Concept of the Lateralized Approach

A novel lateralized system was created, by adapting the lateralized framework, to obtain a proof-of-concept of the lateralized approach. Two modules, LHSM and RHSM, were developed to process the same input signal at a constituent level and holistic level. This ability to consider the given problem at different levels of abstraction (i.e. constituent level and holistic level) enables the novel system to reframe a complex problem as a simple problem and efficiently resolve it. For example, the novel system addressed the n-bit Parity problem as a two-bit problem by utilizing the learned concept of the (n-1)-bit Parity problem and the one additional condition bit #n.

A heterogeneous knowledge pool was created to store the learned BBKs of knowledge. The novel system has to simultaneously consider the given problem at different levels of abstraction. There will be more and more candidate constituent BBKs as the problem scales and the knowledge pool grows. This adds extra computations and does slow down the behavior. However, the utilization of excite or inhibit signals assists the novel system to reduce extra computations. Moreover, the LHSM and RHSM modules have the ability to automatically (without human-in-the-loop intervention) identify the relevant BBKs that can solve the given problem. Consequently, the novel lateralized system scales much more quickly than systems that do not consider sub-problems.

A wide range of experiments were conducted to investigate the overhead created by irrelevant sub-problems to the novel system. The experimental results reveal that this overhead linearly increases as the knowledge pool grows. Moreover, this overhead is very small as compared to the total number of problem instances required for the training. For example, at worst 128000 problem instances are needed to find the relevant BBKs for 18-bit HMux when the system has learned 14 diverse knowledge steps that form 16 candidate combinations (see Chapter 5).

Lateralization and modular learning enable the novel system to encap-

sulate underlying knowledge patterns in the form of building blocks of knowledge. Problems with a natural hierarchy of patterns are solved to a scale beyond previous work, and reusing learned general patterns as constituents for future problems advances transfer learning. The novel lateralized system was anticipated to be suited to hierarchical Boolean problems. The experimental results show that it is not over-fitted or only-suited to such engineered problems as demonstrated by the Parity results where the problem is not observably (or constructed to be) hierarchical.

9.2.3 Robustness of the Lateralized Approach

A novel lateralized system was created, by adapting the lateralized framework, to evaluate the robustness of the lateralized approach. Lateralization enables the novel system to process the same visual environmental input at constituent and holistic levels of abstraction. Consequently, the novel lateralized system exhibits robustness against adversarial attacks. This is because an adversarial attack needs to successfully challenge the constituents as well as the holistic components of an image to fool the lateralized system. It is important to note that the aim of this work was to create a system that can exhibit natural robustness against noisy, irrelevant, and redundant data and not create another adversarial avoidance technique for a specific model or specific adversarial attack.

The novel lateralized system has two main phases, i.e. context phase and attention phase. The simple problem instances are handled at the context phase, whereas, more attention is automatically (without human-in-the-loop intervention) given to the noisy and corrupt problem instances based on the feedback from the context phase. This strategy empowers the novel lateralized system to make correct decisions for badly corrupted images where either the constituent predictions are confused or the holistic prediction favors the wrong class.

Lateralization enables the novel system to use or reuse BBKs at different levels of abstraction. That is, a holistic level BBK created for one

representation can be used as a constituent level BBK in another representation. For example, during the context phase, processing the eyes, nose, and mouth are constituent level sub-problems, whereas the face is a holistic level sub-problem. However, during the final feedback analysis, the face becomes a constituent level sub-problem, whereas the whole image prediction is a holistic level sub-problem (see Chapter 6). This ability to address the problem at different scales empowers the novel system to successfully exhibit robustness against adversarial attacks. The experimental results demonstrate that the lateralized system outperformed all the state-of-the-art deep models for the classification of normal and adversarial images by $0.43\% - 2.56\%$ and $2.15\% - 25.84\%$, respectively.

The decision-making process of the novel system is interpretable (see Chapter 6). It is possible to know what features and at what confidence level led to a decision. This system, therefore, makes a step toward explainable artificial intelligence.

The novel system is an ensemble-like system, in that it resolves different components of a problem to make a final decision. However, the ability to consider the same sub-problem at different levels of abstraction and the use of excite/inhibit signals to activate/deactivate system components make it a lateralized system rather than an ensemble system.

Another novel lateralized systems was created for CV problems to show that the lateralized approach can be scaled and not reliant on LCSs. The first lateralized system was developed to address binary-class image classification tasks, whereas, the second lateralized system was an improved version of the first implementation to address multi-class (200 classes) image classification tasks. The experimental results show that the novel multi-class system outperformed all the state-of-the-art deep models for the classification of normal and adversarial images by $19.05\% - 41.02\%$ and $1.36\% - 49.22\%$, respectively.

9.2.4 Effectiveness of the Lateralized Approach

The simultaneous processing of an input signal at a constituent level and holistic level is an essential part of a lateralized framework. The lateralized AI systems were successfully developed, for Boolean and CV problems, by using the left/right representation of an input signal. But it is not the only way to consider a problem at different levels of abstraction. For example, in navigation tasks, the two levels of abstraction are described by their FoR: either a constituent/local (local viewpoint or egocentric FoR) or a holistic/global (world map or route-centric FoR) representation. Although these levels of representation are not strongly lateralised to left and right hemispheres, they retain the basic principle of simultaneous and coordinated processing that is characteristic of lateralization. However, it does not stop the underlying notion of lateralization being an effective way of describing the AI systems.

A novel FoRs based system was created, by adapting the lateralized framework, to evaluate the effectiveness of the lateralized approach. FoR based strategy enables the novel system to consider the same environmental signal at different levels of abstraction, which provides multiple environmental views, i.e. local viewpoint (constituent knowledge, egocentric FoR) and world viewpoint (holistic knowledge, allocentric and routecentric FoRs). This strategy empowers the novel system to successfully overcome perceptual aliasing problems in multi-step decision-making tasks.

CPs and policies assist the novel system to generate heterogeneous BBKs. Single-step CPs are used to provide constituent representation, whereas, multi-step CPs, policies, and the adjacent states map are used to provide holistic representation. The learned BBKs are integrated at different levels of abstraction to generate an unambiguous representation of knowledge. The resultant knowledge assists the novel system to disambiguate complex patterns of aliased states. Consequently, the novel system transforms a non-Markov environment into a Markov environment, in which it can learn stable policies.

The learning pace of the novel system is slow at the start. It is expensive to learn constituent level BBKs, i.e. single-step CPs. Once learned, these BBKs are (re)used to learn holistic level BBKs, i.e. multi-step CPs and policies. As the problem scales in size and complexity, there are increasing number of constituent-level and holistic-level BBKs, which may slow down learning. However, the novel system has the ability to automatically (without human-in-the-loop intervention) activate/deactivate the most suitable policy. This enables the AI agent to reach the goal state utilizing the minimum number of steps and avoid extraneous computations. The experimental results show that the overall computation cost to reach the goal state for the novel lateralized system is less than that of conventional system (see Chapter 7).

The novel system is robust against aliasing states because of its focus on the appropriate parts of the reward signal to achieve a necessary level of abstraction. Aliasing challenges existing evolutionary computing systems across a wide range of problem domains. The experiments demonstrate that the novel system uses (or re-uses) relevant learned BBKs at different levels of abstraction to learn aliasing patterns that consist of patterns of features. A step-change in performance is achieved, e.g. the state-of-the-art in the heavily aliased maze10 was reduced from an average to 35 to 6.5 steps.

9.3 Future Work

This section presents research directions enabled by this work.

9.3.1 Lateralized AI Systems for Neuroscience

In cognitive neuroscience, the relationship between lateralization and performance is ambiguous. Lateralized AI systems could be developed to provide insight into contentious topics in neuroscience. For example, in

vision (human or other animals), vertebrate brains do have complementary lateralized modules that represent objects at constituent (left) and holistic (right) levels. A lateralized AI system for visual tasks could be used to explain the costs and benefits associated with lateralization. For this purposes, different experiments could be designed to analyze the (i) simultaneous processing of an input signal at constituent and holistic levels, (ii) processing of an input signal at a constituent level only, (iii) using excite/inhibit signals for performance efficiency, (iv) always processing an input signal at multiple levels of abstraction. Such an analysis could provide an insight into the intact and brain-damaged humans. Moreover, such systems could also be created for other contentious topics in neuroscience.

9.3.2 Lateralized Ensemble Systems

Ensemble learning is a state-of-the-art methodology of generating a solution by utilizing multiple learning algorithms. It combines multiple hypotheses with an intention to generate a better hypothesis and reduce the risk of selecting a poor hypothesis [334, 335, 336]. The majority of ensemble systems are computationally expensive. Ensemble systems encourage diversity for better performance, but there is no guarantee that the performance of an ensemble system is always equal to the performance of the best constituent algorithm [366, 367, 368]. Moreover, the performance of an ensemble system depends on ensemble size, however, it is hard to find an appropriate ensemble size (see Chapter 8).

Lateralized ensemble systems could be developed by adapting the lateralized framework to overcome the limitations of ensemble systems. The use of excite/inhibit signals would enable the lateralized ensemble systems to use only the relevant and appropriate constituent algorithms as goal dictate. Consequently, it would eliminate the risk of poor selection and reduce the computational costs. The ability to consider the same problem at different levels of abstraction would enable the lateralized ensemble

systems to automatically identify the constituent algorithms that can efficiently solve the given problem. This functionality would assist the system to automatically find the appropriate ensemble size. Moreover, the ability to (re)use learned BBKs at different levels of abstraction could empower lateralized ensemble systems to solve problems beyond the existing work.

9.3.3 Lateralized Deep Learning

Deep learning (DL) is a state-of-the-art approach for extracting useful patterns and higher-level features by using multiple layers in artificial neural networks [287]. It has been used to solve many real-world problems. The majority of DL-based systems generate a homogeneous knowledge representation such that all features within a layer are treated equally. This knowledge can not be reused at different levels of abstraction [9]. DL-based systems can not identify and transfer relevant knowledge between levels in the hierarchy. Consequently, these systems generate a huge/deep network of homogeneous knowledge to solve complex problems.

Early attempts to address these deficiencies have been made in Capsule networks that can be used to better model hierarchical relationships [18]. Capsule networks add entity-oriented vectorial structures, named capsules, to a convolutional neural network. A capsule is a set of neurons whose activity vector represents the instantiation parameters of an entity of a particular form, such as an individual, or a part, of an object. Active capsules at one level make predictions for the instantiation parameters of higher-level capsules, through transformation matrices. A higher-level capsule becomes active when several low-level predictions agree [18]. In capsule networks, the lower layers outputs are routed to upper layers by using routing algorithms. The majority of these routing procedures can not differentiate positive vectors from their negative counterparts. Thus a capsule network can only learn if both the negative input and the original input represent the same class. Consequently, capsule networks are not suitable for learning some concrete but simple problems [341].

Lateralized DL systems could be developed, by adapting the lateralized framework, to overcome the limitations of DL-based systems. Instead of treating all features equally, a lateralized system would have the ability to simultaneously consider the features at a constituent level and holistic level. These features could then be (re)used at different levels of abstractions between the layers. Consequently, a heterogeneous knowledge representation could be generated.

A capsule (from the capsule networks) could be considered as a BBK in a lateralized DL system. These BBKs could resolve a part of a problem or the whole problem. Such learned BBKs could be stored in the shared heterogeneous knowledge pool. The left half and right half modules could use these BBKs at different levels of abstraction to solve complex problems. Novel strategies could be created to activate/deactivate capsules (BBKs) by using excite/inhibit signals as the goal dictates.

9.3.4 Lateralized Granular Computing

Granular computing is an emerging technique used to identify regularities in the data, that are present at different levels of abstraction [343]. The approach makes efficient use of granular structure to represent the same problem at different levels and different viewpoints [344]. Granular computing has the following limitations. First, a level of abstraction is homogeneous, i.e. it consists only of granules that have a similar nature or granularity. Second, it is hard to find the best granularity level and the number of granules that are appropriate to solve a complex problem. Third, a hierarchical representation provides only a particular viewpoint and can not be used for other purposes. Fourth, relationships among granules and their relationship with granulation form complex solution patterns. Fifth, their use is limited to fuzzy sets and rough sets only (see Chapter 8).

Lateralized granular computing-based systems could be developed by adapting the lateralized framework to overcome the limitations of granular computing. A lateralized system would consider the same level of

abstraction from a constituent viewpoint and holistic viewpoint. Consequently, constituent as well as holistic granules could be utilized to generate heterogeneous knowledge representation for the level of abstraction. The lateralized system would have the ability to automatically consider the given problem at multiple levels of abstraction. For each level, the relevant constituent and holistic level BBKs would be identified. Subsequently, these BBKs would be used to solve a part of the problem or the whole problem. The heterogeneous knowledge pool would allow the novel lateralized system to consider the given problem from multiple perspectives. Different BBKs could be used/reused at different levels of abstraction to generate different perspectives.

Instead of generating a complex granulation structure, the lateralized system could automatically consider the given problem at multiple levels of abstraction, and resolve each level by utilizing learned constituent level and holistic level BBKs, depending on the nature and complexity of the problem. Moreover, the relationships between different BBKs (constituent and holistic level) could be automatically established. Finally, the novel lateralized systems would not be limited to fuzzy sets and rough sets.

This thesis devises a novel lateralized framework that has been successfully adapted to create lateralized AI systems for three different problem domains. The experimental results demonstrate that the novel lateralized AI systems outperformed state-of-the-art (non-lateralized) systems in resolving complex problems. It shows that lateralization is beneficial for AI systems. It is hoped that this thesis may open a door for creating lateralized AI systems for a wide range of problem domains.

Bibliography

[1] E. Ackerman, "Toyota's Gill Pratt on self-driving cars and the reality of full autonomy," *IEEE Spectrum*, vol. 23, 2017.

[2] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, *et al.*, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

[3] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, "Deep learning applications and challenges in big data analytics," *Journal of Big Data*, vol. 2, no. 1, p. 1, 2015.

[4] C. M. Checka, J. E. Chun, F. R. Schnabel, J. Lee, and H. Toth, "The relationship of mammographic density and age: implications for breast cancer screening," *American Journal of Roentgenology*, vol. 198, no. 3, pp. W292–W295, 2012.

[5] T. M. Mitchell, "Machine learning. 1997," *Burr Ridge, IL: McGraw Hill*, vol. 45, no. 37, pp. 870–877, 1997.

[6] A. Orriols-Puig, J. Casillas, and E. Bernadó-Mansilla, "Genetic-based machine learning systems are competitive for pattern recognition," *Evolutionary Intelligence*, vol. 1, no. 3, pp. 209–232, 2008.

[7] S. J. Russell and P. Norvig, "Artificial intelligence: a modern approach (international edition)," 2002.

[8] S. Russell and P. Norvig, "A modern approach," *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs*, vol. 25, p. 27, 1995.

[9] M. Shanahan, K. Nikiforou, A. Creswell, C. Kaplanis, D. Barrett, and M. Garnelo, "An explicitly relational neural network architecture," *CoRR*, vol. abs/1905.10307, 2019.

[10] S. Frazier and M. Riedl, "Improving deep reinforcement learning in minecraft with action advice," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 15, pp. 146–152, 2019.

[11] S. Krening and K. M. Feigh, "Newtonian action advice: Integrating human verbal instruction with reinforcement learning," in *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 720–727, International Foundation for Autonomous Agents and Multiagent Systems, 2019.

[12] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[13] P. Domingos, *The master algorithm: How the quest for the ultimate learning machine will remake our world*. Basic Books, 2018.

[14] M. V. Butz, *"Anticipatory learning classifier systems"*, vol. 4. Springer Science & Business Media, 2002.

[15] M. Iqbal, W. N. Browne, and M. Zhang, "Reusing building blocks of extracted knowledge to solve complex, large-scale boolean problems," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 465–480, 2013.

[16] K. Suzuki and S. Kato, "Hierarchical reinforcement learning introducing genetic algorithm for POMDPs environments.," in *ICAART (2)*, pp. 318–327, 2019.

[17] G. Konidaris, "On the necessity of abstraction," *Current Opinion in Behavioral Sciences*, vol. 29, pp. 1–7, 2019.

[18] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Advances in neural information processing systems*, pp. 3856–3866, 2017.

[19] J. Zerilli, "Neural reuse and the modularity of mind: Where to next for modularity?," *Biological Theory*, vol. 14, no. 1, pp. 1–20, 2019.

[20] M. Anderson, "Neural reuse: A fundamental organizational principle of the brain," *Behavioral and brain sciences*, vol. 33, no. 4, p. 245, 2010.

[21] W. H. Gaddes, *Learning disabilities and brain function: A neuropsychological approach*. Springer Science & Business Media, 2013.

[22] S. Finger, *Origins of Neuroscience: a history of explorations into brain function*. Oxford University Press, USA, 2001.

[23] A. R. Luria, *Higher cortical functions in man*. Springer Science & Business Media, 2012.

[24] M. C. Corballis, "The evolution of lateralized brain circuits," *Frontiers in Psychology*, vol. 8, p. 1021, 2017.

[25] M. L. Anderson, *After phrenology: Neural reuse and the interactive brain*. MIT Press, 2014.

[26] J. L. Krichmar, "The neuromodulatory system: a framework for survival and adaptive behavior in a challenging world," *Adaptive Behavior*, vol. 16, no. 6, pp. 385–399, 2008.

[27] A. S. Alexander and D. A. Nitz, "Retrosplenial cortex maps the conjunction of internal and external spaces," *Nature Neuroscience*, vol. 18, no. 8, pp. 1143–1151, 2015.

[28] D. A. Nitz, "Spaces within spaces: Rat parietal cortex neurons register position across three reference frames," *Nature Neuroscience*, vol. 15, no. 10, pp. 1365–1367, 2012.

[29] M. T. Banich and R. Compton, *Cognitive Neuroscience*. Nelson Education, 2010.

[30] J. Hutsler and R. A. Galuske, "Hemispheric asymmetries in cerebral cortical networks," *Trends in Neurosciences*, vol. 26, no. 8, pp. 429–435, 2003.

[31] L. J. Rogers, P. Zucca, and G. Vallortigara, "Advantages of having a lateralized brain," *Proceedings of the Royal Society of London B: Biological Sciences*, vol. 271, no. Suppl 6, pp. S420–S422, 2004.

[32] L. C. Robertson and M. R. Lamb, "Neuropsychological contributions to theories of part/whole organization," *Cognitive Psychology*, vol. 23, no. 2, pp. 299–330, 1991.

[33] L. C. Robertson and D. C. Delis, "Part-whole processing in unilateral brain-damaged patients: Dysfunction of hierarchical organization," *Neuropsychologia*, vol. 24, no. 3, pp. 363–370, 1986.

[34] M. Martin, "Hemispheric specialization for local and global processing," *Neuropsychologia*, vol. 17, no. 1, pp. 33–40, 1979.

[35] G. Hickok and D. Poeppel, "Neural basis of speech perception," in *Neurobiology of Language*, pp. 299–310, Elsevier, 2016.

[36] G. M. Grimshaw, J. A. Séguin, and H. K. Godfrey, "Once more with feeling: The effects of emotional prosody on hemispheric specialisation for linguistic processing," *Journal of Neurolinguistics*, vol. 22, no. 4, pp. 313–326, 2009.

[37] D. Wildgruber, H. Ackermann, B. Kreifelts, and T. Ethofer, "Cerebral processing of linguistic and emotional prosody: fMRI studies," *Progress in Brain Research*, vol. 156, pp. 249–268, 2006.

[38] M. V. Chafee, B. B. Averbeck, and D. A. Crowe, "Representing spatial relationships in posterior parietal cortex: single neurons code object-referenced position," *Cerebral Cortex*, vol. 17, no. 12, pp. 2914–2932, 2007.

[39] R. Morris, P. Garrud, J. a. Rawlins, and J. O'Keefe, "Place navigation impaired in rats with hippocampal lesions," *Nature*, vol. 297, no. 5868, pp. 681–683, 1982.

[40] S. Frässle, F. M. Paulus, S. Krach, S. R. Schweinberger, K. E. Stephan, and A. Jansen, "Mechanisms of hemispheric lateralization: Asymmetric interhemispheric recruitment in the face perception network," *Neuroimage*, vol. 124, pp. 977–988, 2016.

[41] Y.-C. Chan, T.-L. Chou, H.-C. Chen, Y.-C. Yeh, J. P. Lavallee, K.-C. Liang, and K.-E. Chang, "Towards a neural circuit model of verbal humor processing: An fMRI study of the neural substrates of incongruity detection and resolution," *Neuroimage*, vol. 66, pp. 169–176, 2013.

[42] S. K. Yelle and G. M. Grimshaw, "Hemispheric specialization for linguistic processing of sung speech," *Perceptual and Motor Skills*, vol. 108, no. 1, pp. 219–228, 2009.

[43] M. Dharmaretnam and L. Rogers, "Hemispheric specialization and dual processing in strongly versus weakly lateralized chicks," *Behavioural Brain Research*, vol. 162, no. 1, pp. 62–70, 2005.

[44] D. E. Stark, D. S. Margulies, Z. E. Shehzad, P. Reiss, A. C. Kelly, L. Q. Uddin, D. G. Gee, A. K. Roy, M. T. Banich, F. X. Castellanos,

et al., "Regional variation in interhemispheric coordination of intrinsic hemodynamic fluctuations," *Journal of Neuroscience*, vol. 28, no. 51, pp. 13754–13764, 2008.

[45] W. Browne, K. Holford, C. Moore, and J. Bullock, "A practical application of a learning classifier system in a steel hot strip mill," in *Artificial Neural Nets and Genetic Algorithms*, pp. 611–614, Springer, 1998.

[46] W. N. Browne, "The development of an industrial learning classifier system for data-mining in a steel hop strip mill," in *Applications of Learning Classifier Systems*, pp. 223–259, Springer, 2004.

[47] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 427–436, 2015.

[48] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, vol. 6, pp. 14410–14430, 2018.

[49] A. Gopnik, "Will a.i. ever be smarter than a four-year-old?." `https://www.smithsonianmag.com/innovation/will-ai-ever-be-smarter-than-four-year-old-180971259/`, 2020. [Online; accessed July 27, 2020].

[50] R. Szeliski, *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[51] I. M. Alvarez, W. N. Browne, and M. Zhang, "Human-inspired scaling in learning classifier systems: Case study on the n-bit multiplexer problem set," in *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 429–436, ACM, 2016.

[52] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[53] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2574–2582, 2016.

[54] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 5, pp. 828–841, 2019.

[55] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[56] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," *arXiv preprint arXiv:1607.02533*, 2016.

[57] S. D. Whitehead and D. H. Ballard, "Learning to perceive and act by trial and error," *Machine Learning*, vol. 7, no. 1, pp. 45–83, 1991.

[58] L. Chrisman, "Reinforcement learning with perceptual aliasing: The perceptual distinctions approach," in *AAAI*, vol. 1992, pp. 183–188, 1992.

[59] P. L. Lanzi, "Adaptive agents with reinforcement learning and internal memory," in *Sixth International Conference on the Simulation of Adaptive Behavior (SAB2000)*, pp. 333–342, MIT Press, 2000.

[60] Z. V. Zatuchna and A. J. Bagnall, "AgentP Classifier System: self-adjusting vs. gradual approach," in *2005 IEEE Congress on Evolutionary Computation*, vol. 2, pp. 1196–1203, IEEE, 2005.

[61] "Cognitive systems." https://www.gov.uk/government/collections/cognitive-systems, 2017. [Online; accessed December 04, 2017].

[62] R. Sternberg and K. Sternberg, *Cognitive Psychology*. Cengage Learning, 2016.

[63] O. Blomberg, "Conceptions of cognition for cognitive engineering," *The international journal of aviation Psychology*, vol. 21, no. 1, pp. 85–104, 2011.

[64] M. S. Gazzaniga, *Handbook of cognitive Neuroscience*. Springer, 2014.

[65] M. Shanahan, "The brain's connective core and its role in animal cognition," *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, vol. 367, no. 1603, pp. 2704–2714, 2012.

[66] "Hemispheres." https://en.wikipedia.org/wiki/Cerebral_hemisphere, 2017. [Online; accessed Nov 12, 2020].

[67] M. L. Concha, I. H. Bianco, and S. W. Wilson, "Encoding asymmetry within neural circuits," *Nature Reviews Neuroscience*, vol. 13, no. 12, pp. 832–843, 2012.

[68] W. D. Chapple, "Central and peripheral origins of asymmetry in the abdominal motor system of the hermit crab," *Annals of the New York Academy of Sciences*, vol. 299, no. 1, pp. 43–58, 1977.

[69] A. Pascual, K.-L. Huang, J. Neveu, and T. Préat, "Neuroanatomy: brain asymmetry and long-term memory," *Nature*, vol. 427, no. 6975, pp. 605–606, 2004.

[70] Y. Shinohara, H. Hirase, M. Watanabe, M. Itakura, M. Takahashi, and R. Shigemoto, "Left-right asymmetry of the hippocampal synapses

with differential subunit allocation of glutamate receptors," *Proceedings of the National Academy of Sciences*, vol. 105, no. 49, pp. 19498–19503, 2008.

[71] R. Kawakami, Y. Shinohara, Y. Kato, H. Sugiyama, R. Shigemoto, and I. Ito, "Asymmetrical allocation of nmda receptor ε2 subunits in hippocampal circuitry," *Science*, vol. 300, no. 5621, pp. 990–994, 2003.

[72] R. Young and C. Govind, "Neural asymmetry in male fiddler crabs," *Brain Research*, vol. 280, no. 2, pp. 251–262, 1983.

[73] M. Faust and Y. N. Kenett, "Rigidity, chaos and integration: hemispheric interaction and individual differences in metaphor comprehension," *Frontiers in Human Neuroscience*, vol. 8, 2014.

[74] H. J. Mirous and M. Beeman, "Bilateral processing and affect in creative language comprehension," *The Handbook of the Neuropsychology of Language, Volume 1&2*, pp. 317–341, 2012.

[75] M. Faust, "Thinking outside the left box: the role of the right hemisphere in novel metaphor comprehension," *Advances in the neural substrates of language: Toward a synthesis of basic science and clinical research*, pp. 425–448, 2012.

[76] M. Faust and A. Kahana, "Priming summation in the cerebral hemispheres: Evidence from semantically convergent and semantically divergent primes," *Neuropsychologia*, vol. 40, no. 7, pp. 892–901, 2002.

[77] O. M. Razoumnikova, "Functional organization of different brain areas during convergent and divergent thinking: an eeg investigation," *Cognitive Brain Research*, vol. 10, no. 1, pp. 11–18, 2000.

[78] M. Faust and C. Chiarello, "Sentence context and lexical ambiguity resolution by the two hemispheres," *Neuropsychologia*, vol. 36, no. 9, pp. 827–835, 1998.

[79] C. Burgess and G. B. Simpson, "Cerebral hemispheric mechanisms in the retrieval of ambiguous word meanings," *Brain and language*, vol. 33, no. 1, pp. 86–103, 1988.

[80] H. H. Brownell, D. Michel, J. Powelson, and H. Gardner, "Surprise but not coherence: Sensitivity to verbal humor in right-hemisphere patients," *Brain and language*, vol. 18, no. 1, pp. 20–27, 1983.

[81] L. J. Rogers, "Evolution of hemispheric specialization: advantages and disadvantages," *Brain and language*, vol. 73, no. 2, pp. 236–253, 2000.

[82] D. P. Chivers, M. I. McCormick, D. T. Warren, B. J. Allan, R. A. Ramasamy, B. K. Arvizu, M. Glue, and M. C. Ferrari, "Competitive superiority versus predation savvy: the two sides of behavioural lateralization," *Animal Behaviour*, vol. 130, pp. 9–15, 2017.

[83] D. P. Chivers, M. I. McCormick, B. J. Allan, M. D. Mitchell, E. J. Gonçalves, R. Bryshun, and M. C. Ferrari, "At odds with the group: changes in lateralization and escape performance reveal conformity and conflict in fish schools," *Proceedings of the Royal Society B: Biological Sciences*, vol. 283, no. 1841, p. 20161127, 2016.

[84] A. V. Flevaris and L. C. Robertson, "Spatial frequency selection and integration of global and local information in visual processing: A selective review and tribute to shlomo bentin," *Neuropsychologia*, vol. 83, pp. 192–200, 2016.

[85] L. C. Robertson and R. Ivry, "Hemispheric asymmetries: Attention to visual and auditory primitives," *Current Directions in Psychological Science*, vol. 9, no. 2, pp. 59–63, 2000.

[86] D. A. Nitz, "Parietal cortex, navigation, and the construction of arbitrary reference frames for spatial information," *Neurobiology of learning and memory*, vol. 91, no. 2, pp. 179–185, 2009.

[87] T. Iachini, G. Ruggiero, M. Conson, and L. Trojano, "Lateralization of egocentric and allocentric spatial processing after parietal brain lesions," *Brain and cognition*, vol. 69, no. 3, pp. 514–520, 2009.

[88] G. Committeri, G. Galati, A.-L. Paradis, L. Pizzamiglio, A. Berthoz, and D. LeBihan, "Reference frames for spatial cognition: different brain areas are involved in viewer-, object-, and landmark-centered judgments about object location," *Journal of Cognitive Neuroscience*, vol. 16, no. 9, pp. 1517–1535, 2004.

[89] T. Zaehle, K. Jordan, T. Wüstenberg, J. Baudewig, P. Dechent, and F. W. Mast, "The neural basis of the egocentric and allocentric spatial frame of reference," *Brain research*, vol. 1137, pp. 92–103, 2007.

[90] K. Iglói, C. F. Doeller, A. Berthoz, L. Rondi-Reig, and N. Burgess, "Lateralized human hippocampal activity predicts navigation based on sequence or place memory," *Proceedings of the National Academy of Sciences*, vol. 107, no. 32, pp. 14466–14471, 2010.

[91] V. Borghesani and M. Piazza, "The neuro-cognitive representations of symbols: the case of concrete words," *Neuropsychologia*, vol. 105, pp. 4–17, 2017.

[92] M. A. L. Ralph, E. Jefferies, K. Patterson, and T. T. Rogers, "The neural and computational bases of semantic cognition," *Nature Reviews Neuroscience*, vol. 18, no. 1, p. 42, 2017.

[93] T. T. Rogers and J. L. McClelland, *Semantic cognition: A parallel distributed processing approach*. MIT press, 2004.

[94] T. T. Rogers, M. A. Lambon Ralph, P. Garrard, S. Bozeat, J. L. McClelland, J. R. Hodges, and K. Patterson, "Structure and deterioration of semantic memory: a neuropsychological and computational investigation.," *Psychological Review*, vol. 111, no. 1, p. 205, 2004.

[95] A. M. Collins and E. F. Loftus, "A spreading-activation theory of semantic processing.," *Psychological Review*, vol. 82, no. 6, p. 407, 1975.

[96] A. M. Collins and M. R. Quillian, "Retrieval time from semantic memory," *Journal of verbal learning and verbal behavior*, vol. 8, no. 2, pp. 240–247, 1969.

[97] L. W. Barsalou, "Perceptions of perceptual symbols," *Behavioral and brain sciences*, vol. 22, no. 4, pp. 637–660, 1999.

[98] K. Patterson, P. J. Nestor, and T. T. Rogers, "Where do you know what you know? the representation of semantic knowledge in the human brain," *Nature Reviews Neuroscience*, vol. 8, no. 12, pp. 976–987, 2007.

[99] A. Martin, "GRAPES—Grounding representations in action, perception, and emotion systems: How object properties and categories are represented in the human brain," *Psychonomic Bulletin & Review*, vol. 23, no. 4, pp. 979–990, 2016.

[100] J. R. Binder, L. L. Conant, C. J. Humphries, L. Fernandino, S. B. Simons, M. Aguilar, and R. H. Desai, "Toward a brain-based componential semantic representation," *Cognitive Neuropsychology*, vol. 33, no. 3-4, pp. 130–174, 2016.

[101] M. A. Lambon Ralph, K. Sage, R. W. Jones, and E. J. Mayberry, "Coherent concepts are computed in the anterior temporal lobes," *Proceedings of the National Academy of Sciences*, vol. 107, no. 6, pp. 2717–2722, 2010.

[102] E. Jefferies and M. A. Lambon Ralph, "Semantic impairment in stroke aphasia versus semantic dementia: a case-series comparison," *Brain*, vol. 129, no. 8, pp. 2132–2147, 2006.

[103] D. Badre, R. A. Poldrack, E. J. Paré-Blagoev, R. Z. Insler, and A. D. Wagner, "Dissociable controlled retrieval and generalized selection mechanisms in ventrolateral prefrontal cortex," *Neuron*, vol. 47, no. 6, pp. 907–918, 2005.

[104] A. D. Wagner, E. J. Paré-Blagoev, J. Clark, and R. A. Poldrack, "Recovering meaning: left prefrontal cortex guides controlled semantic retrieval," *Neuron*, vol. 31, no. 2, pp. 329–338, 2001.

[105] S. L. Thompson-Schill, M. D'Esposito, G. K. Aguirre, and M. J. Farah, "Role of left inferior prefrontal cortex in retrieval of semantic knowledge: a reevaluation," *Proceedings of the National Academy of Sciences*, vol. 94, no. 26, pp. 14792–14797, 1997.

[106] M. J. Farah and J. L. McClelland, "A computational model of semantic memory impairment: Modality specificity and emergent category specificity.," *Journal of experimental Psychology: General*, vol. 120, no. 4, p. 339, 1991.

[107] E. K. Warrington and T. Shallice, "Category specific semantic impairments," *Brain*, vol. 107, no. 3, pp. 829–853, 1984.

[108] L. W. Barsalou, W. K. Simmons, A. K. Barbey, and C. D. Wilson, "Grounding conceptual knowledge in modality-specific systems," *Trends in cognitive sciences*, vol. 7, no. 2, pp. 84–91, 2003.

[109] M. A. Lambon Ralph, "Neurocognitive insights on conceptual knowledge and its breakdown," *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, vol. 369, no. 1634, p. 20120392, 2014.

[110] L. Meteyard, S. R. Cuadrado, B. Bahrami, and G. Vigliocco, "Coming of age: A review of embodiment and the Neuroscience of semantics," *Cortex*, vol. 48, no. 7, pp. 788–804, 2012.

[111] J. A. Fodor, *The modularity of mind: An essay on faculty Psychology*. MIT press, 1983.

[112] L. W. Barsalou, "Grounded cognition," *Annu. Rev. Psychol.*, vol. 59, pp. 617–645, 2008.

[113] L. Wittgenstein, "Philosophische untersuchungen. the german text, with a revised english translation," 2001.

[114] E. E. Smith and D. L. Medin, *Categories and concepts*. Harvard University Press Cambridge, MA, 1981.

[115] E. Rosch, "Cognitive representations of semantic categories.," *Journal of experimental Psychology: General*, vol. 104, no. 3, p. 192, 1975.

[116] P. Hoffman, J. L. McClelland, L. Ralph, and A. Matthew, "Concepts, control, and context: A connectionist account of normal and disordered semantic cognition.," *Psychological Review*, vol. 125, no. 3, p. 293, 2018.

[117] D. A. Nitz, "Tracking route progression in the posterior parietal cortex," *Neuron*, vol. 49, no. 5, pp. 747–756, 2006.

[118] C. R. Olson and S. N. Gettner, "Object-centered direction selectivity in the macaque supplementary eye field," *Science*, vol. 269, no. 5226, pp. 985–988, 1995.

[119] J. O'keefe and L. Nadel, *The hippocampus as a cognitive map*. Oxford: Clarendon Press, 1978.

[120] J. Cho and P. E. Sharp, "Head direction, place, and movement correlates for cells in the rat retrosplenial cortex.," *Behavioral Neuroscience*, vol. 115, no. 1, p. 3, 2001.

[121] L. M. Frank, E. N. Brown, and M. Wilson, "Trajectory encoding in the hippocampus and entorhinal cortex," *Neuron*, vol. 27, no. 1, pp. 169–178, 2000.

[122] E. R. Wood, P. A. Dudchenko, R. J. Robitsek, and H. Eichenbaum, "Hippocampal neurons encode information about different types of memory episodes occurring in the same location," *Neuron*, vol. 27, no. 3, pp. 623–633, 2000.

[123] J. O'Keefe and J. Dostrovsky, "The hippocampus as a spatial map. preliminary evidence from unit activity in the freely-moving rat," *Brain Research*, vol. 34, no. 1, pp. 171–175, 1971.

[124] J. Ferbinteanu and M. L. Shapiro, "Prospective and retrospective memory coding in the hippocampus," *Neuron*, vol. 40, no. 6, pp. 1227–1239, 2003.

[125] E. R. Wood and P. A. Dudchenko, "Aging, spatial behavior and the cognitive map," *Nature Neuroscience*, vol. 6, no. 6, pp. 546–548, 2003.

[126] J. J. Knierim, H. S. Kudrimoti, and B. L. McNaughton, "Place cells, head direction cells, and the learning of landmark stability," *Journal of Neuroscience*, vol. 15, no. 3, pp. 1648–1659, 1995.

[127] M. A. Wilson and B. L. McNaughton, "Dynamics of the hippocampal ensemble code for space," *Science*, vol. 261, no. 5124, pp. 1055–1059, 1993.

[128] H. Eichenbaum, C. Stewart, and R. Morris, "Hippocampal representation in place learning," *Journal of Neuroscience*, vol. 10, no. 11, pp. 3531–3542, 1990.

[129] B. McNaughton, C. A. Barnes, and J. O'keefe, "The contributions of position, direction, and velocity to single unit activity in the hippocampus of freely-moving rats," *Experimental Brain Research*, vol. 52, no. 1, pp. 41–49, 1983.

[130] R. Czajkowski, B. Jayaprakash, B. Wiltgen, T. Rogerson, M. C. Guzman-Karlsson, A. L. Barth, J. T. Trachtenberg, and A. J. Silva,

"Encoding and storage of spatial information in the retrosplenial cortex," *Proceedings of the National Academy of Sciences*, vol. 111, no. 23, pp. 8661–8666, 2014.

[131] S. A. Marchette, L. K. Vass, J. Ryan, and R. A. Epstein, "Anchoring the neural compass: coding of local spatial reference frames in human medial parietal lobe," *Nature Neuroscience*, vol. 17, no. 11, pp. 1598–1606, 2014.

[132] S. D. Auger and E. A. Maguire, "Assessing the mechanism of response in the retrosplenial cortex of good and poor navigators," *Cortex*, vol. 49, no. 10, pp. 2904–2913, 2013.

[133] S. D. Auger, S. L. Mullally, and E. A. Maguire, "Retrosplenial cortex codes for permanent landmarks," *PloS one*, vol. 7, no. 8, p. e43620, 2012.

[134] L. L. Chen, L.-H. Lin, E. J. Green, C. A. Barnes, and B. L. McNaughton, "Head-direction cells in the rat posterior cortex," *Experimental Brain Research*, vol. 101, no. 1, pp. 8–23, 1994.

[135] P. Byrne, S. Becker, and N. Burgess, "Remembering the past and imagining the future: a neural model of spatial memory and imagery.," *Psychological Review*, vol. 114, no. 2, p. 340, 2007.

[136] N. Burgess, S. Becker, J. A. King, and J. O'Keefe, "Memory for events and their spatial context: models and experiments," *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, vol. 356, no. 1413, pp. 1493–1503, 2001.

[137] P. Malhotra, E. J. Coulthard, and M. Husain, "Role of right posterior parietal cortex in maintaining attention to spatial locations over time," *Brain*, vol. 132, no. 3, pp. 645–660, 2009.

[138] E. Bullmore and O. Sporns, "The economy of brain network organization," *Nature Reviews Neuroscience*, vol. 13, no. 5, pp. 336–349, 2012.

[139] C. A. Buneo, M. R. Jarvis, A. P. Batista, and R. A. Andersen, "Direct visuomotor transformations for reaching," *Nature*, vol. 416, no. 6881, pp. 632–636, 2002.

[140] C. L. Colby and M. E. Goldberg, "Space and attention in parietal cortex," *Annual review of Neuroscience*, vol. 22, no. 1, pp. 319–349, 1999.

[141] J.-R. Duhamel, C. L. Colby, and M. E. Goldberg, "The updating of the representation of visual space in parietal cortex by intended eye movements," *Science*, vol. 255, no. 5040, p. 90, 1992.

[142] L. H. Snyder, K. L. Grieve, P. Brotchie, and R. A. Andersen, "Separate body-and world-referenced representations of visual space in parietal cortex," *Nature*, vol. 394, no. 6696, pp. 887–891, 1998.

[143] R. A. Andersen, G. K. Essick, and R. M. Siegel, "Encoding of spatial location by posterior parietal neurons," *Science*, vol. 230, no. 4724, pp. 456–458, 1985.

[144] U. J. Ilg, S. Schumann, and P. Thier, "Posterior parietal cortex neurons encode target motion in world-centered coordinates," *Neuron*, vol. 43, no. 1, pp. 145–151, 2004.

[145] F. Klam and W. Graf, "Vestibular signals of posterior parietal cortex neurons during active and passive head movements in macaque monkeys," *Annals of the New York Academy of Sciences*, vol. 1004, no. 1, pp. 271–282, 2003.

[146] K. Kawano, M. Sasaki, and M. Yamashita, "Vestibular input to visual tracking neurons in the posterior parietal association cortex of the monkey," *Neuroscience letters*, vol. 17, no. 1, pp. 55–60, 1980.

[147] B. L. McNaughton, S. Mizumori, C. Barnes, B. Leonard, M. Marquis, and E. Green, "Cortical representation of motion during unrestrained spatial navigation in the rat," *Cerebral Cortex*, vol. 4, no. 1, pp. 27–39, 1994.

[148] T. Hafting, M. Fyhn, S. Molden, M.-B. Moser, and E. I. Moser, "Microstructure of a spatial map in the entorhinal cortex," *Nature*, vol. 436, no. 7052, pp. 801–806, 2005.

[149] J. Jacobs, M. J. Kahana, A. D. Ekstrom, M. V. Mollison, and I. Fried, "A sense of direction in human entorhinal cortex," *Proceedings of the National Academy of Sciences*, vol. 107, no. 14, pp. 6487–6492, 2010.

[150] J. Jacobs, C. T. Weidemann, J. F. Miller, A. Solway, J. F. Burke, X.-X. Wei, N. Suthana, M. R. Sperling, A. D. Sharan, I. Fried, *et al.*, "Direct recordings of grid-like neuronal activity in human spatial navigation," *Nature Neuroscience*, vol. 16, no. 9, pp. 1188–1190, 2013.

[151] N. J. Killian, M. J. Jutras, and E. A. Buffalo, "A map of visual space in the primate entorhinal cortex," *Nature*, vol. 491, no. 7426, pp. 761–764, 2012.

[152] M. M. Yartsev, M. P. Witter, and N. Ulanovsky, "Grid cells without theta oscillations in the entorhinal cortex of bats," *Nature*, vol. 479, no. 7371, pp. 103–107, 2011.

[153] C. F. Doeller, C. Barry, and N. Burgess, "Evidence for grid cells in a human memory network," *Nature*, vol. 463, no. 7281, pp. 657–661, 2010.

[154] M. Fyhn, T. Hafting, M. P. Witter, E. I. Moser, and M.-B. Moser, "Grid cells in mice," *Hippocampus*, vol. 18, no. 12, pp. 1230–1238, 2008.

[155] "Gridcells." `http://www.scholarpedia.org/article/Grid_cells`, 2017. [Online; accessed Jan 04, 2018].

[156] "Gridcells." `https://en.wikipedia.org/wiki/Grid_cell`, 2017. [Online; accessed Jan 04, 2018].

[157] D. C. Rowland, Y. Roudi, M.-B. Moser, and E. I. Moser, "Ten years of grid cells," *Annual review of Neuroscience*, vol. 39, pp. 19–40, 2016.

[158] A.-L. Bibost and C. Brown, "Laterality influences cognitive performance in rainbowfish melanotaenia duboulayi," *Animal cognition*, vol. 17, no. 5, pp. 1045–1051, 2014.

[159] M. Magat and C. Brown, "Laterality enhances cognition in australian parrots," *Proceedings of the Royal Society of London B: Biological Sciences*, vol. 276, no. 1676, pp. 4155–4162, 2009.

[160] D. B. Boles, J. M. Barth, and E. C. Merrill, "Asymmetry and performance: Toward a neurodevelopmental theory," *Brain and cognition*, vol. 66, no. 2, pp. 124–139, 2008.

[161] M. Hirnstein, K. Hugdahl, and M. Hausmann, "How brain asymmetry relates to performance–a large-scale dichotic listening study," *Frontiers in psychology*, vol. 4, p. 997, 2014.

[162] J. Levy, "Lateral specialization of the human brain, behavioral manifestations and possible evolutionary basis," *The biology of behavior*, 1972.

[163] M. Holder, *What Does Handedness Have to Do with Brain Lateralization (and Who Cares?).* MK Holder., 2002.

[164] M. Hirnstein, K. Hugdahl, and M. Hausmann, "Cognitive sex differences and hemispheric asymmetry: A critical review of 40 years of research," *Laterality: Asymmetries of Body, Brain and Cognition*, vol. 24, no. 2, pp. 204–252, 2019.

[165] D. W. Johnston, M. E. Nicholls, M. Shah, and M. A. Shields, "Nature's experiment? handedness and early childhood development," *Demography*, vol. 46, no. 2, pp. 281–301, 2009.

[166] M. E. Nicholls, H. L. Chapman, T. Loetscher, and G. M. Grimshaw, "The relationship between hand preference, hand performance, and general cognitive ability," *Journal of the International Neuropsychological Society*, vol. 16, no. 4, pp. 585–592, 2010.

[167] M. C. Corballis, J. Hattie, and R. Fletcher, "Handedness and intellectual achievement: An even-handed look," *Neuropsychologia*, vol. 46, no. 1, pp. 374–378, 2008.

[168] S. Leask and T. Crow, "A single optimum degree of hemispheric specialisation in two tasks, in two uk national birth cohorts," *Brain and cognition*, vol. 62, no. 3, pp. 221–227, 2006.

[169] J. M. Barth, D. B. Boles, A. A. Giattina, and C. E. Penn, "Preschool child and adult lateralisation and performance in emotion and language tasks," *Laterality: Asymmetries of Body, Brain and Cognition*, vol. 17, no. 4, pp. 412–427, 2012.

[170] G. M. Grimshaw and L. Kranz, "Hemispheric asymmetries in schizotypy," in *Schizotypy*, pp. 62–80, Routledge, 2015.

[171] J. E. Laird, C. Lebiere, and P. S. Rosenbloom, "A standard model of the mind: Toward a common computational framework across artificial intelligence, cognitive science, neuroscience, and robotics," *Ai Magazine*, vol. 38, no. 4, pp. 13–26, 2017.

[172] A. Lieto, M. Bhatt, A. Oltramari, and D. Vernon, "The role of cognitive architectures in general artificial intelligence," 2018.

[173] UCS-ICT, "Cognitive architecture." `https://cogarch.ict.usc.edu/`, 2020. [Online; accessed Dec 13, 2020].

[174] I. Kotseruba and J. K. Tsotsos, "40 years of cognitive architectures: core cognitive abilities and practical applications," *Artificial Intelligence Review*, vol. 53, no. 1, pp. 17–94, 2020.

[175] J. E. Laird, *The Soar cognitive architecture*. MIT press, 2012.

[176] R. Sun, "The importance of cognitive architectures: An analysis based on clarion," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 19, no. 2, pp. 159–193, 2007.

[177] A. J. Butt, N. A. Butt, A. Mazhar, Z. Khattak, and J. A. Sheikh, "The soar of cognitive architectures," in *2013 International Conference on Current Trends in Information Technology (CTIT)*, pp. 135–142, IEEE, 2013.

[178] P. Langley, J. E. Laird, and S. Rogers, "Cognitive architectures: Research issues and challenges," *Cognitive Systems Research*, vol. 10, no. 2, pp. 141–160, 2009.

[179] W. Duch, R. J. Oentaryo, and M. Pasquier, "Cognitive architectures: Where do we go from here?," in *Agi*, vol. 171, pp. 122–136, 2008.

[180] P. Thagard, "Cognitive architectures," *The Cambridge handbook of cognitive science*, pp. 50–70, 2012.

[181] J. K. Tsotsos and W. Kruijne, "Cognitive programs: software for attention's executive," *Frontiers in psychology*, vol. 5, p. 1260, 2014.

[182] J. K. Tsotsos, *A computational perspective on visual attention*. MIT Press, 2011.

[183] P. Öztürk, "Levels and types of action selection: the action selection soup," *Adaptive behavior*, vol. 17, no. 6, pp. 537–554, 2009.

[184] N. Cowan, "What are the differences between long-term, short-term, and working memory?," *Progress in brain research*, vol. 169, pp. 323–338, 2008.

[185] L. R. Squire, "Declarative and nondeclarative memory: Multiple brain systems supporting learning and memory," *Journal of cognitive neuroscience*, vol. 4, no. 3, pp. 232–243, 1992.

[186] E. Brynjolfsson, D. Rock, and C. Syverson, "Artificial intelligence and the modern productivity paradox: A clash of expectations and statistics," in *Economics of Artificial Intelligence*, University of Chicago Press, 2017.

[187] P. Lin, K. Abney, and R. Jenkins, *Robot Ethics 2.0: From Autonomous Cars to Artificial Intelligence*. Oxford University Press, 2017.

[188] S. Makridakis, "The forthcoming artificial intelligence (ai) revolution: Its impact on society and firms," *Futures*, vol. 90, pp. 46–60, 2017.

[189] E. Alpaydin, *Introduction to machine learning*. MIT press, 2020.

[190] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.

[191] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[192] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the royal statistical society. series c (applied statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[193] R. Sibson, "Slink: an optimally efficient algorithm for the single-link cluster method," *The computer journal*, vol. 16, no. 1, pp. 30–34, 1973.

[194] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, vol. 1. MIT press Cambridge, 1998.

[195] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Machine learning*, vol. 3, no. 1, pp. 9–44, 1988.

[196] F. S. Chance, J. B. Aimone, S. S. Musuvathy, M. R. Smith, C. M. Vineyard, and F. Wang, "Crossing the cleft: communication challenges between neuroscience and artificial intelligence," *Frontiers in Computational Neuroscience*, vol. 14, p. 39, 2020.

[197] wikipedia, "Adversarial attack to capsule networks." `https://en.wikipedia.org/wiki/Artificial_neuron`, 2021. [Online; accessed Aug 29, 2021].

[198] D. L. Yamins, H. Hong, C. F. Cadieu, E. A. Solomon, D. Seibert, and J. J. DiCarlo, "Performance-optimized hierarchical models predict neural responses in higher visual cortex," *Proceedings of the national academy of sciences*, vol. 111, no. 23, pp. 8619–8624, 2014.

[199] F. H. Sinz, X. Pitkow, J. Reimer, M. Bethge, and A. S. Tolias, "Engineering a less artificial intelligence," *Neuron*, vol. 103, no. 6, pp. 967–979, 2019.

[200] C. Blundell, B. Uria, A. Pritzel, Y. Li, A. Ruderman, J. Z. Leibo, J. Rae, D. Wierstra, and D. Hassabis, "Model-free episodic control," *arXiv preprint arXiv:1606.04460*, 2016.

[201] B. Thomee, M. J. Huiskes, E. Bakker, and M. S. Lew, "Visual information retrieval using synthesized imagery," in *Proceedings of the 6th ACM international conference on Image and video retrieval*, pp. 127–130, 2007.

[202] V. Mnih, N. Heess, A. Graves, *et al.*, "Recurrent models of visual attention," in *Advances in neural information processing systems*, pp. 2204–2212, 2014.

[203] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska,

et al., "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[204] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[205] D. George, W. Lehrach, K. Kansky, M. Lázaro-Gredilla, C. Laan, B. Marthi, X. Lou, Z. Meng, Y. Liu, H. Wang, *et al.*, "A generative vision model that trains with high data efficiency and breaks text-based captchas," *Science*, vol. 358, no. 6368, 2017.

[206] D. George, A. Lavin, J. S. Guntupalli, D. Mely, N. Hay, and M. Lázaro-Gredilla, "Cortical microcircuits from a generative vision model," *arXiv preprint arXiv:1808.01058*, 2018.

[207] A. Nayebi, D. Bear, J. Kubilius, K. Kar, S. Ganguli, D. Sussillo, J. J. DiCarlo, and D. L. Yamins, "Task-driven convolutional recurrent models of the visual system," *arXiv preprint arXiv:1807.00053*, 2018.

[208] K. Kar, J. Kubilius, K. Schmidt, E. B. Issa, and J. J. DiCarlo, "Evidence that recurrent circuits are critical to the ventral stream's execution of core object recognition behavior," *Nature neuroscience*, vol. 22, no. 6, pp. 974–983, 2019.

[209] J. Kubilius, M. Schrimpf, K. Kar, H. Hong, N. J. Majaj, R. Rajalingham, E. B. Issa, P. Bashivan, J. Prescott-Roy, K. Schmidt, *et al.*, "Brain-like object recognition with high-performing shallow recurrent anns," *arXiv preprint arXiv:1909.06161*, 2019.

[210] K. Fukushima and S. Miyake, "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition," in *Competition and cooperation in neural nets*, pp. 267–285, Springer, 1982.

[211] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[212] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[213] A. Tavanaei, M. Ghodrati, S. R. Kheradpisheh, T. Masquelier, and A. Maida, "Deep learning in spiking neural networks," *Neural Networks*, vol. 111, pp. 47–63, 2019.

[214] T. J. Draelos, N. E. Miner, C. C. Lamb, J. A. Cox, C. M. Vineyard, K. D. Carlson, W. M. Severa, C. D. James, and J. B. Aimone, "Neurogenesis deep learning: Extending deep networks to accommodate new classes," in *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 526–533, IEEE, 2017.

[215] N. Y. Masse, G. D. Grant, and D. J. Freedman, "Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization," *Proceedings of the National Academy of Sciences*, vol. 115, no. 44, pp. E10467–E10475, 2018.

[216] R. V. Rikhye, A. Gilra, and M. M. Halassa, "Thalamic regulation of switching between cortical representations enables cognitive flexibility," *Nature neuroscience*, vol. 21, no. 12, pp. 1753–1763, 2018.

[217] S. D. Wiederman, J. M. Fabian, J. R. Dunbier, and D. C. O'Carroll, "A predictive focus of gain modulation encodes target trajectories in insect vision," *Elife*, vol. 6, p. e26478, 2017.

[218] S. D. Wiederman and D. C. O'Carroll, "Selective attention in an insect visual neuron," *Current Biology*, vol. 23, no. 2, pp. 156–161, 2013.

[219] C. J. McAdams and J. H. Maunsell, "Effects of attention on orientation-tuning functions of single neurons in macaque cortical area v4," *Journal of Neuroscience*, vol. 19, no. 1, pp. 431–441, 1999.

[220] S. Treue and J. C. M. Trujillo, "Feature-based attention influences motion processing gain in macaque visual cortex," *Nature*, vol. 399, no. 6736, pp. 575–579, 1999.

[221] S. D. Wiederman, P. A. Shoemaker, and D. C. O'Carroll, "A model for the detection of moving targets in visual clutter inspired by insect physiology," *PloS one*, vol. 3, no. 7, p. e2784, 2008.

[222] Z. M. Bagheri, S. D. Wiederman, B. S. Cazzolato, S. Grainger, and D. C. O'Carroll, "Properties of neuronal facilitation that improve target tracking in natural pursuit simulations," *Journal of The Royal Society Interface*, vol. 12, no. 108, p. 20150083, 2015.

[223] Z. M. Bagheri, S. D. Wiederman, B. S. Cazzolato, S. Grainger, and D. C. O'Carroll, "Performance of an insect-inspired target tracker in natural conditions," *Bioinspiration & biomimetics*, vol. 12, no. 2, p. 025006, 2017.

[224] Z. M. Bagheri, B. S. Cazzolato, S. Grainger, D. C. O'Carroll, and S. D. Wiederman, "An autonomous robot inspired by insect neurophysiology pursues moving features in natural environments," *Journal of neural engineering*, vol. 14, no. 4, p. 046030, 2017.

[225] J. S. Taube, R. U. Muller, and J. B. Ranck, "Head-direction cells recorded from the postsubiculum in freely moving rats. i. description and quantitative analysis," *Journal of Neuroscience*, vol. 10, no. 2, pp. 420–435, 1990.

[226] A. Arleo and W. Gerstner, "Spatial cognition and neuro-mimetic navigation: a model of hippocampal place cell activity," *Biological cybernetics*, vol. 83, no. 3, pp. 287–299, 2000.

[227] M. Milford, G. Wyeth, and D. Prasser, "Simultaneous localization and mapping from natural landmarks using ratslam," in *Australasian Conference on Robotics and Automation, Canberra, Australia*, 2004.

[228] M. Fyhn, S. Molden, M. P. Witter, E. I. Moser, and M.-B. Moser, "Spatial representation in the entorhinal cortex," *Science*, vol. 305, no. 5688, pp. 1258–1264, 2004.

[229] A. Banino, C. Barry, B. Uria, C. Blundell, T. Lillicrap, P. Mirowski, A. Pritzel, M. J. Chadwick, T. Degris, J. Modayil, *et al.*, "Vector-based navigation using grid-like representations in artificial agents," *Nature*, vol. 557, no. 7705, pp. 429–433, 2018.

[230] C. J. Cueva and X.-X. Wei, "Emergence of grid-like representations by training recurrent neural networks to perform spatial localization," *arXiv preprint arXiv:1803.07770*, 2018.

[231] F. Yu, J. Shang, Y. Hu, and M. Milford, "Neuroslam: a brain-inspired slam system for 3d environments," *Biological cybernetics*, vol. 113, no. 5, pp. 515–545, 2019.

[232] J. L. Bellmund, P. Gärdenfors, E. I. Moser, and C. F. Doeller, "Navigating cognition: Spatial codes for human thinking," *Science*, vol. 362, no. 6415, 2018.

[233] J. Hawkins, M. Lewis, M. Klukas, S. Purdy, and S. Ahmad, "A framework for intelligence and cortical function based on grid cells in the neocortex," *Frontiers in neural circuits*, vol. 12, p. 121, 2019.

[234] A. E. Eiben, J. E. Smith, *et al.*, *Introduction to evolutionary computing*, vol. 53. Springer, 2003.

[235] K. A. De Jong, *Evolutionary computation: a unified approach*. MIT press, 2006.

[236] M. Iqbal, W. N. Browne, and M. Zhang, "Extracting and using building blocks of knowledge in learning classifier systems," in *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation*, pp. 863–870, ACM, 2012.

[237] J. H. Holland *et al.*, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[238] J. Holland and D. Goldberg, "Genetic algorithms in search, optimization and machine learning," *Massachusetts: Addison-Wesley*, 1989.

[239] R. J. Urbanowicz and W. N. Browne, *Introduction to Learning Classifier Systems*. Springer, 2017.

[240] R. J. Urbanowicz and J. H. Moore, "Learning classifier systems: a complete introduction, review, and roadmap," *Journal of Artificial Evolution and Applications*, vol. 2009, p. 1, 2009.

[241] M. V. Butz and W. Stolzmann, "An algorithmic description of acs2," in *International Workshop on Learning Classifier Systems*, pp. 211–229, Springer, 2001.

[242] J. Holland and J. Reitman, "Cognitive systems based on adaptive algorithms reprinted in: Evolutionary computation. the fossil record," *IEEE Press, New York*, 1998.

[243] T. Kovacs, "Evolving optimal populations with XCS classi er systems," *Master's thesis, School of Computer Science, University of Birmingham, Birmingham, UK*, 1996.

[244] M. V. Butz, *Rule-based evolutionary online learning systems*. Springer, 2006.

[245] R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, *Machine learning: An artificial intelligence approach*. Springer Science & Business Media, 2013.

[246] E. Bernadó-Mansilla and J. M. Garrell-Guiu, "Accuracy-based learning classifier systems: models, analysis and applications to classification tasks," *Evolutionary computation*, vol. 11, no. 3, pp. 209–238, 2003.

[247] L. B. Booker, D. E. Goldberg, and J. H. Holland, "Classifier systems and genetic algorithms," *Artificial intelligence*, vol. 40, no. 1-3, pp. 235–282, 1989.

[248] S. F. Smith, "A learning system based on genetic adaptive algorithms," 1980.

[249] S. W. Wilson, "ZCS: A zeroth level classifier system," *Evolutionary computation*, vol. 2, no. 1, pp. 1–18, 1994.

[250] M. V. Butz, T. Kovacs, P. L. Lanzi, and S. W. Wilson, "Toward a theory of generalization and learning in XCS," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 1, pp. 28–46, 2004.

[251] S. W. Wilson, "Classifier fitness based on accuracy," *Evolutionary Computation*, vol. 3, no. 2, pp. 149–175, 1995.

[252] A. Siddique, M. Iqbal, and W. N. Browne, "A comprehensive strategy for mammogram image classification using learning classifier systems," in *Evolutionary Computation (CEC), 2016 IEEE Congress on*, pp. 2201–2208, IEEE, 2016.

[253] H. Al-Sahaf, "Genetic programming for automatically synthesising robust image descriptors with a small number of instances," 2017.

[254] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[255] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the seventh IEEE international conference on computer vision*, vol. 2, pp. 1150–1157, Ieee, 1999.

[256] Y. Ke, R. Sukthankar, L. Huston, Y. Ke, and R. Sukthankar, "Efficient near-duplicate detection and sub-image retrieval," in *Acm Multimedia*, vol. 4, p. 5, Citeseer, 2004.

[257] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol. 1, pp. 886–893, IEEE, 2005.

[258] J. Chen, Z. Chen, Z. Chi, and H. Fu, "Facial expression recognition based on facial components detection and hog features," in *International workshops on electrical and computer engineering subfields*, pp. 884–888, 2014.

[259] T. Kovacs, "XCS classifier system reliably evolves accurate, complete, and minimal representations for boolean functions," in *Soft Computing in Engineering Design and Manufacturing*, pp. 59–68, Springer, 1998.

[260] T. Dean, *Network+ guide to networks*. Cengage Learning, 2012.

[261] J. J. Brophy *et al.*, "Basic electronics for scientists," 1971.

[262] L. Huelsbergen, "Finding general solutions to the parity problem by evolving machine-language representations," *Genetic Programming*, pp. 158–166, 1998.

[263] M. Iqbal, "Improving the scalability of XCS-based learning classifier systems," 2014.

[264] kaggleCats, "kagglecats." `https://www.kaggle.com/crawford/cat-dataset`, 2020. [Online; accessed Feb 02, 2020].

[265] open source, "C++ library for machine learning." `http://dlib.net/files/data/`, 2020. [Online; accessed Feb 02, 2020].

[266] J. Liu, A. Kanazawa, D. Jacobs, and P. Belhumeur, "Dog breed classification using part localization," in *European conference on computer vision*, pp. 172–185, Springer, 2012.

[267] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The caltech-ucsd birds-200-2011 dataset," 2011.

[268] A. S. Alexander and D. A. Nitz, "Spatially periodic activation patterns of retrosplenial cortex encode route sub-spaces and distance traveled," *Current Biology*, vol. 27, no. 11, pp. 1551–1560, 2017.

[269] D. A. Nitz, "Path shape impacts the extent of ca1 pattern recurrence both within and across environments," *Journal of neurophysiology*, vol. 105, no. 4, pp. 1815–1824, 2011.

[270] T. Oess, J. L. Krichmar, and F. Röhrbein, "A computational model for spatial navigation based on reference frames in the hippocampus, retrosplenial cortex, and posterior parietal cortex," *Frontiers in neurorobotics*, vol. 11, p. 4, 2017.

[271] P. L. Lanzi and S. W. Wilson, "Toward optimal classifier system performance in non-Markov environments," *Evolutionary Computation*, vol. 8, no. 4, pp. 393–418, 2000.

[272] Z. V. Zatuchna and A. Bagnall, "Learning mazes with aliasing states: An lcs algorithm with associative perception," *Adaptive Behavior*, vol. 17, no. 1, pp. 28–57, 2009.

[273] M. Métivier and C. Lattaud, "Anticipatory classifier system using behavioral sequences in non-Markov environments," in *International Workshop on Learning Classifier Systems*, pp. 143–162, Springer, 2002.

[274] P. Stone and M. Veloso, "Layered learning," in *Proceedings of European Conference on Machine Learning*, pp. 369–381, Springer, 2000.

[275] T. H. Hoang, R. I. McKay, D. Essam, and N. X. Hoai, "On synergistic interactions between evolution, development and layered learning," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 3, pp. 287–312, 2011.

[276] D. Jackson and A. P. Gibbons, "Layered learning in boolean gp problems," in *Proceedings of European Conference on Genetic Programming*, pp. 148–159, Springer, 2007.

[277] L. Rodriguez-Coayahuitl, A. Morales-Reyes, and H. J. Escalante, "Structurally layered representation learning: towards deep learning through genetic programming," in *European Conference on Genetic Programming*, pp. 271–288, Springer, 2018.

[278] D. O'Neill, H. Al-Sahaf, B. Xue, and M. Zhang, "Common subtrees in related problems: A novel transfer learning approach for genetic programming," in *2017 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1287–1294, IEEE, 2017.

[279] J. F. Miller, "Cartesian genetic programming," in *Proceedings of Cartesian Genetic Programming*, pp. 17–34, Springer, 2011.

[280] J. F. Miller and P. Thomson, "Cartesian genetic programming," in *Proceedings of European Conference on Genetic Programming*, pp. 121–132, Springer, 2000.

[281] M. Iqbal, W. N. Browne, and M. Zhang, "Extending learning classifier system with cyclic graphs for scalability on complex, large-scale

boolean problems," in *Proceedings of the 15th annual conference on Genetic and evolutionary computation*, pp. 1045–1052, 2013.

[282] R. P. Wiegand, *An analysis of cooperative coevolutionary algorithms*. PhD thesis, Citeseer, 2003.

[283] M. A. Potter and K. A. D. Jong, "Cooperative coevolution: An architecture for evolving coadapted subcomponents," *Evolutionary Computation*, vol. 8, no. 1, pp. 1–29, 2000.

[284] M. A. Potter and K. A. De Jong, "A cooperative coevolutionary approach to function optimization," in *Proceedings of International Conference on Parallel Problem Solving from Nature*, pp. 249–257, Springer, 1994.

[285] R. Cheng, M. N. Omidvar, A. H. Gandomi, B. Sendhoff, S. Menzel, and X. Yao, "Solving incremental optimization problems via cooperative coevolution," *IEEE Transactions on Evolutionary Computation*, 2018.

[286] T. B. Nguyen, W. N. Browne, and M. Zhang, "Improvement of code fragment fitness to guide feature construction in XCS," in *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 428–436, 2019.

[287] L. Deng, D. Yu, *et al.*, "Deep learning: methods and applications," *Foundations and Trends® in Signal Processing*, vol. 7, no. 3–4, pp. 197–387, 2014.

[288] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[289] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[290] S. Sankaranarayanan, A. Jain, R. Chellappa, and S. N. Lim, "Regularizing deep networks using efficient layerwise adversarial training," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[291] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1765–1773, 2017.

[292] N. Akhtar, J. Liu, and A. Mian, "Defense against universal adversarial perturbations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3389–3398, 2018.

[293] G. K. Dziugaite, Z. Ghahramani, and D. M. Roy, "A study of the effect of jpg compression on adversarial images," *arXiv preprint arXiv:1608.00853*, 2016.

[294] A. N. Bhagoji, D. Cullina, C. Sitawarin, and P. Mittal, "Enhancing robustness of machine learning systems via data transformations," in *2018 52nd Annual Conference on Information Sciences and Systems (CISS)*, pp. 1–5, IEEE, 2018.

[295] S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," *arXiv preprint arXiv:1412.5068*, 2014.

[296] A. S. Ross and F. Doshi-Velez, "Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients," in *Thirty-second AAAI conference on artificial intelligence*, 2018.

[297] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *2016 IEEE Symposium on Security and Privacy (SP)*, pp. 582–597, IEEE, 2016.

[298] M. Tan, "Cost-sensitive reinforcement learning for adaptive classification and control.," in *AAAI*, pp. 774–780, 1991.

[299] L. Chrisman, R. Caruana, and W. Carriker, "Intelligent agent design issues: Internal agent state and incomplete perception," in *Proceedings of the AAAI Fall Symposium on Sensory Aspects of Robotic Intelligence. AAAI Press/MIT Press*, Citeseer, 1991.

[300] N. Roy, G. Gordon, and S. Thrun, "Finding approximate POMDP solutions through belief compression," *Journal of artificial intelligence research*, vol. 23, pp. 1–40, 2005.

[301] W. S. Lovejoy, "A survey of algorithmic methods for partially observed Markov decision processes," *Annals of Operations Research*, vol. 28, no. 1, pp. 47–65, 1991.

[302] M. W. Mitchell, "Using Markov-k memory for problems with hidden-state.," in *MLMTA*, pp. 242–248, 2003.

[303] P. L. Lanzi, "An analysis of the memory mechanism of XCSM," *Genetic Programming*, vol. 98, pp. 643–651, 1998.

[304] G. G. Robertson and R. L. Riolo, "A tale of two classifier systems," *Machine Learning*, vol. 3, no. 2-3, pp. 139–159, 1988.

[305] R. E. Smith, "Memory exploitation in learning classifier systems," *Evolutionary Computation*, vol. 2, no. 3, pp. 199–220, 1994.

[306] T. Hayashida, I. Nishizaki, and K. Moriwake, "XCS with an internal action table for non-Markov environments," *Editorial Preface*, vol. 5, no. 6, 2014.

[307] T. Hayashida, I. Nishizaki, S. Sekizaki, and H. Takeuchi, "Improved anticipatory classifier system with internal memory for POMDPs with aliased states," *Procedia computer science*, vol. 112, pp. 215–224, 2017.

[308] W. Stolzmann, "Latent learning in khepera robots with anticipatory classifier systems," in *2nd International Workshop on Learning Classifier Systems, Orlando, Florida, USA*, 1999.

[309] R. Orhand, A. Jeannin-Girardon, P. Parrend, and P. Collet, "BACS: A thorough study of using behavioral sequences in ACS2," in *Int. Conf. on Parall. Prob. Solv. from Nature*, pp. 524–538, Springer, 2020.

[310] J. A. Fails and D. R. Olsen Jr, "Interactive machine learning," in *Proceedings of the 8th international conference on Intelligent user interfaces*, pp. 39–45, 2003.

[311] S. Griffith, K. Subramanian, J. Scholz, C. L. Isbell, and A. L. Thomaz, "Policy shaping: Integrating human feedback with reinforcement learning," in *Advances in neural information processing systems*, pp. 2625–2633, 2013.

[312] M. Wiering and J. Schmidhuber, "HQ-learning," *Adaptive Behavior*, vol. 6, no. 2, pp. 219–246, 1997.

[313] K. Suzuki and S. Kato, "Hybrid learning using profit sharing and genetic algorithm for partially observable Markov decision processes," in *International Conference on Network-Based Information Systems*, pp. 463–475, Springer, 2017.

[314] M. V. Butz, M. Pelikan, X. Llora, and D. E. Goldberg, "Automated global structure extraction for effective local building block processing in xcs," *Evolutionary Computation*, vol. 14, no. 3, pp. 345–380, 2006.

[315] J. F. Miller and S. L. Smith, "Redundancy and computational efficiency in cartesian genetic programming," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 2, pp. 167–174, 2006.

[316] M. Horiuchi and M. Nakata, "Self-adaptation of XCS learning parameters based on learning theory," in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, pp. 342–349, 2020.

[317] R. Chandra, A. Gupta, Y.-S. Ong, and C.-K. Goh, "Evolutionary multi-task learning for modular knowledge representation in neural networks," *Neural Processing Letters*, vol. 47, no. 3, pp. 993–1009, 2018.

[318] M. V. Butz and S. W. Wilson, "An algorithmic description of XCS," *Soft Computing*, vol. 6, no. 3-4, pp. 144–153, 2002.

[319] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[320] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[321] S. Chandra, "Implementation of papers on adversarial examples." `https://github.com/sarathknv/adversarial-examples-pytorch/tree/master`, 2020. [Online; accessed Feb 02, 2020].

[322] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[323] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size," *arXiv preprint arXiv:1602.07360*, 2016.

[324] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE transactions on pattern analysis and machine intelligence*, vol. 20, no. 8, pp. 832–844, 1998.

[325] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[326] C. Zhang and Y. Ma, *Ensemble machine learning: methods and applications*. Springer, 2012.

[327] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in Python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[328] P. Crook and G. Hayes, "Learning in a state of confusion: Perceptual aliasing in grid world navigation," *Towards Intelligent Mobile Robots*, vol. 4, 2003.

[329] W. Stolzmann, "An introduction to anticipatory classifier systems," in *International Workshop on Learning Classifier Systems*, pp. 175–194, Springer, 1999.

[330] M. L. Littman, A. R. Cassandra, and L. P. Kaelbling, "Learning policies for partially observable environments: Scaling up," in *Machine Learning Proceedings 1995*, pp. 362–370, Elsevier, 1995.

[331] open source, "C++ library for XCS." `http://xcslib.sourceforge.net/`, 2020. [Online; accessed March 26, 2020].

[332] M. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable mdps," *arXiv preprint arXiv:1507.06527*, 2015.

[333] G. Lample and D. S. Chaplot, "Playing fps games with deep reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, 2017.

[334] D. Opitz and R. Maclin, "Popular ensemble methods: An empirical study," *Journal of artificial intelligence research*, vol. 11, pp. 169–198, 1999.

[335] R. Polikar, "Ensemble based systems in decision making," *IEEE Circuits and systems magazine*, vol. 6, no. 3, pp. 21–45, 2006.

[336] L. Rokach, "Ensemble-based classifiers," *Artificial intelligence review*, vol. 33, no. 1-2, pp. 1–39, 2010.

[337] H. R. Bonab and F. Can, "A theoretical framework on the ideal number of classifiers for online ensembles in data streams," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pp. 2053–2056, 2016.

[338] H. Bonab and F. Can, "Less is more: a comprehensive framework for the number of components of ensemble classifiers," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 9, pp. 2735–2745, 2019.

[339] G. Fumera and F. Roli, "A theoretical and experimental analysis of linear combiners for multiple classifier systems," *IEEE transactions on pattern analysis and machine intelligence*, vol. 27, no. 6, pp. 942–956, 2005.

[340] D. Parikh and R. Polikar, "An ensemble-based incremental learning approach to data fusion," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 2, pp. 437–450, 2007.

[341] D. Peer, S. Stabinger, and A. Rodriguez-Sanchez, "Limitations of routing-by-agreement based capsule networks," *arXiv preprint arXiv:1905.08744*, 2019.

[342] Jsikyoon, "Adversarial attack to capsule networks." `https://github.com/jsikyoon/adv_attack_capsnet`, 2020. [Online; accessed Dec 16, 2020].

[343] Y. Yao, "Artificial intelligence perspectives on granular computing," in *Granular Computing and Intelligent Systems*, pp. 17–34, Springer, 2011.

[344] J. T. Yao, A. V. Vasilakos, and W. Pedrycz, "Granular computing: perspectives and challenges," *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 1977–1989, 2013.

[345] Y. Yao, "A triarchic theory of granular computing," *Granular Computing*, vol. 1, no. 2, pp. 145–157, 2016.

[346] A. Bargiela and W. Pedrycz, *Human-centric information processing through granular modelling*, vol. 182. Springer Science & Business Media, 2009.

[347] Y. Yao, "A partition model of granular computing," in *Transactions on rough sets I*, pp. 232–253, Springer, 2004.

[348] Y. Yao, "Three-way decision and granular computing," *International Journal of Approximate Reasoning*, vol. 103, pp. 107–123, 2018.

[349] Z. Pawlak, "Rough sets: Theoretical aspects of reasoning about data kluwer academic publishers," *Google Scholar*, 1991.

[350] S.-H. Chen and Y.-R. Du, "Granularity in economic decision making: An interdisciplinary review," in *Granular Computing and Decision-Making*, pp. 47–71, Springer, 2015.

[351] J. Yao, "Information granulation and granular relationships," in *2005 IEEE international conference on granular computing*, vol. 1, pp. 326–329, IEEE, 2005.

[352] L. A. Zadeh, "Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic," *Fuzzy sets and systems*, vol. 90, no. 2, pp. 111–127, 1997.

[353] J. Stepaniuk, *Rough–Granular Computing in Knowledge Discovery and Data Mining*, vol. 152. Springer, 2009.

[354] J. Yao, "A ten-year review of granular computing," in *2007 IEEE International Conference on Granular Computing (GRC 2007)*, pp. 734–734, IEEE, 2007.

[355] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[356] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the national academy of sciences*, vol. 79, no. 8, pp. 2554–2558, 1982.

[357] S. Ocklenburg, G. Berretz, J. Packheiser, and P. Friedrich, "Laterality 2020: entering the next decade," *Laterality*, pp. 1–33, 2020.

[358] S. Ocklenburg and O. Gunturkun, *The lateralized brain: The neuroscience and evolution of hemispheric asymmetries*. Academic Press, 2017.

[359] M. T. Banich and A. Belger, "Interhemispheric interaction: how do the hemispheres divide and conquer a task?," *Cortex*, vol. 26, no. 1, pp. 77–94, 1990.

[360] H. M. van Ettinger-Veenstra, M. Ragnehed, M. Hällgren, T. Karlsson, A.-M. Landtblom, P. Lundberg, and M. Engström, "Right-hemispheric brain activation correlates to language performance," *Neuroimage*, vol. 49, no. 4, pp. 3481–3488, 2010.

[361] R. Everts, K. Lidzba, M. Wilke, C. Kiefer, M. Mordasini, G. Schroth, W. Perrig, and M. Steinlin, "Strengthening of laterality of verbal and visuospatial functions during childhood and adolescence," *Human brain mapping*, vol. 30, no. 2, pp. 473–483, 2009.

[362] E. Mellet, L. Zago, G. Jobard, F. Crivello, L. Petit, M. Joliot, B. Mazoyer, and N. Tzourio-Mazoyer, "Weak language lateralization affects both verbal and spatial skills: An fmri study in 297 subjects," *Neuropsychologia*, vol. 65, pp. 56–62, 2014.

[363] A. Razafimandimby, N. Tzourio-Mazoyer, B. Mazoyer, O. Maïza, and S. Dollfus, "Language lateralization in left-handed patients with schizophrenia," *Neuropsychologia*, vol. 49, no. 3, pp. 313–319, 2011.

[364] C. Brown and V. A. Braithwaite, "Effects of predation pressure on the cognitive ability of the poeciliid brachyraphis episcopi," *Behavioral Ecology*, vol. 16, no. 2, pp. 482–487, 2005.

[365] M. Dadda, E. Zandonà, C. Agrillo, and A. Bisazza, "The costs of hemispheric specialization in a fish," *Proceedings of the Royal Society of London B: Biological Sciences*, vol. 276, no. 1677, pp. 4399–4407, 2009.

[366] L. I. Kuncheva and C. J. Whitaker, "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy," *Machine learning*, vol. 51, no. 2, pp. 181–207, 2003.

[367] J. J. G. Adeva, U. Beresi, and R. Calvo, "Accuracy and diversity in ensembles of text categorisers," *CLEI Electronic Journal*, vol. 9, no. 1, pp. 1–12, 2005.

[368] P. Sollich and A. Krogh, "Learning with ensembles: How overfitting can be useful," *Advances in neural information processing systems*, vol. 8, pp. 190–196, 1995.