NANYANG
TECHNOLOGICAL
UNIVERSITY

KNOWLEDGE CONSOLIDATION AND INFERENCE IN AN INTEGRATED NEURO-COGNITIVE ARCHITECTURE

RICHARD JAYADI OENTARYO

SCHOOL OF COMPUTER ENGINEERING

2011

KNOWLEDGE CONSOLIDATION AND INFERENCE IN AN INTEGRATED NEURO-COGNITIVE ARCHITECTURE

RICHARD JAYADI OENTARYO

School of Computer Engineering

A thesis submitted to the Nanyang Technological University
in fulfillment of the requirement for the degree of
Doctor of Philosophy

2011

Abstract

Developing a general machine intelligence that can provide truly natural interaction and human-like cognition has been a major challenge in artificial intelligence research. To this end, cognitive architectures are increasingly investigated as generic blueprints for intelligent agents that can operate across different task domains. A variety of cognitive architectures have been formulated over the years, but there remains a need to further develop salient aspects of general intelligence, such as *knowledge consolidation*, *system scalability*, and *metacognitive functions*. This thesis first provides a comprehensive survey of the contemporary cognitive architectures, with systematic categorization of various design approaches and critical evaluations on their design properties, merits and shortcomings. Promising milestones and directions are also outlined, with emphasis on how consolidation, scalability and metacognition can help address the issues in the current cognitive architectures and contribute to the creation of better architectures/systems.

To realize the three salient aspects, an *Integrated Neuro-Cognitive Architecture* (INCA) is proposed in this thesis, which models the putative functional aspects of the major brain systems and their interactions. Accordingly, extensive studies on the neuroscientific and psychological aspects of learning and memory in the human brain have been conducted. Building upon this foundation, this thesis presents the outline of the INCA framework, whereby *neuro-fuzzy system* (NFS) is adopted as a primary modeling approach to realize the INCA's constituent modules. To systematically coordinate the interaction among the modules, two novel procedures namely *consolidation* and *inference cycles* are developed, which make INCA unique in comparison to other contemporary cognitive architectures. Evaluation on the INCA's cognitive plausibility has been performed, and its design features and capabilities are compared with several prominent architectures.

As a first step to realize INCA, particularly its long-term memory modules, a novel NFS model termed the *Reduced Fuzzy Cerebellar Model Articulation Controller* (RFCMAC) has been developed. The model emulates the two-stage neural development of cortical memories in the brain to construct and reduce memory representation, respectively. This construct-then-reduce idea is adopted in the label generation and rule generation phases of the RFCMAC learning process, prior to an iterative parameter tuning phase. These procedures, especially the reduction mechanisms, endow the RFCMAC with the ability to derive highly compact and intuitive rules, to produce satisfactory output generalization, and to scale well in the face of large tasks. The efficacy of the RFCMAC as knowledge extraction tool and prototype model for the INCA long-term memories has been exemplified by experimental results on complex regression and classification tasks.

To provide a key infrastructure supporting the consolidation and inference cycles in INCA, a novel model of memory consolidation called the *Dual Consolidation Network* (DCN) is subsequently proposed. Chiefly, the model emulates the complementary interactions between the hippocampal and cortical systems in the brain to effectively consolidate and exploit knowledge. The proposed DCN system consists of a cortex-like *slow-learning module* (SLM), realized using the RFCMAC model, and a hippocampus-like *fast-learning* module (FLM), realized using a transient NFS model. The latter serves to encode distinct events using a sparse, non-overlapping representation. Knowledge consolidation is achieved via a multi-episode complementary learning between SLM and FLM, whereby *pseudopatterns* are employed as a primary means for inter-module communication. With this mechanism, DCN is able to perform online sequential learning without recourse to the original patterns, while minimizing interference arising from the dynamics of the environment. Extensive experiments on basic cognitive and large-scale tasks have shown the knowledge consolidation, scalability, and generalization traits of the DCN system.

Acknowledgments

I would like to express my foremost gratitude to my advisor, A/P Michel Pasquier, for his constant guidance, support, and patience, without which I would not be able to complete this research study. I would also like to offer thanks to A/P Quek Chai for providing his expertise, wisdom and encouragement, especially during difficult times. Their supervision has made my research very rewarding, despite the many challenges I faced along the way. Through them, I have been able to learn not only useful skills and/or methods, but also great philosophies crucial for my future career and character development.

My fellow members at the Centre for Computational Intelligence (C2i) have provided me with various assistances, ranging from resolving technical problems to sharing their invaluable research ideas. In particular, I would like to thank Huang Haoming, Tan Tuan Zea, Ting Chan Wai, Tan Wi Meng, Cheu Eng Yeow, Tung Whye Loon, Ang Kai Keng, Khoo Suiyang, Ma Li, Zhao Shengkui, Nguyen Minh Nhut, Dudy Lim, Daniel Handoko, Tung Sau Wai, and Nguyen Ngoc Nam, with whom I enjoyed sharing real-life experiences and work directions. I also wish to express my thanks to the laboratory staffs Lau Boon Chee and Tan Swee Huat for their supports in numerous ways.

My life in Singapore would not be half as fun and be much more difficult if it had not been for the best friends I met on the way. My special thanks go to Djoko Wibowo and Rinaldo Tanumara, for sharing their profound insights and suggestions. I am also greatly indebted to my parents and younger brother for their unflagging encouragements and assistances throughout my study. This list should really be longer, including many more dear friends and colleagues. Last but not least, I wish to thank the School of Computer Engineering, Nanyang Technological University, for providing financial supports and an opportunity to work with distinguished teams of students and professors.

Contents

Abstract i

Acknowledgments iii

List of Figures ix

List of Tables xii

1 Introduction **1**

1.1 Towards Artificial General Intelligence 1

1.2 Current Challenges 3

1.3 Main Contributions 5

1.4 Thesis Organization 7

2 Cognitive Architectures: A Survey **8**

2.1 Basic Taxonomy 8

2.1.1 Symbolic Paradigm 9

2.1.2 Emergent Paradigm 10

2.1.3 Hybrid Paradigm 11

2.2 Symbolic Architectures 12

2.2.1 Representative Examples 12

2.2.1.1 SOAR 12

2.2.1.2 EPIC 14

2.2.1.3 Icarus 15

2.2.1.4 Prodigy 16

2.2.1.5 PRS 17

2.2.2 System Comparisons 18

2.2.3 Other Architectures . . . 19
2.3 Emergent Architectures . . . 20
2.3.1 Representative Examples . . . 20
2.3.1.1 IBCA . . . 20
2.3.1.2 Cortronics . . . 21
2.3.1.3 TSCA . . . 22
2.3.1.4 NuPIC . . . 23
2.3.1.5 GWCA . . . 24
2.3.2 System Comparisons . . . 25
2.3.3 Other Architectures . . . 26
2.4 Hybrid Architectures . . . 27
2.4.1 Representative Examples . . . 27
2.4.1.1 ACT-R . . . 27
2.4.1.2 CLARION . . . 28
2.4.1.3 LIDA . . . 29
2.4.1.4 Dual . . . 30
2.4.1.5 Polyscheme . . . 31
2.4.2 System Comparisons . . . 33
2.4.3 Other Architectures . . . 34
2.5 Discussion and Direction . . . 34
2.6 Summary . . . 37

3 Integrated Neuro-Cognitive Architecture 38

3.1 Brain Inspirations . . . 38
3.1.1 Memory . . . 38
3.1.1.1 Long-Term Memory . . . 39
3.1.1.2 Short-Term (Working) Memory . . . 42
3.1.2 Learning . . . 43
3.1.2.1 Nonassociative Learning . . . 43
3.1.2.2 Associative Learning . . . 44

3.1.3 Consolidation and Inference 45
3.2 Related Computational Models 46
3.3 INCA Architectural Framework 50
3.3.1 Architectural Modules 51
3.3.2 Neuro-Fuzzy System Modeling 53
3.3.3 Communication Protocols 56
3.4 Operational Cycles 57
3.4.1 Consolidation Cycle 58
3.4.2 Inference Cycle 61
3.5 System Evaluation 63
3.5.1 Cognitive Plausibility 63
3.5.2 Comparisons with Other Architectures 65
3.6 Summary 66

4 Reduced Localized Network Model for Knowledge Extraction 68
4.1 Knowledge Extraction 68
4.2 System Architecture 71
4.2.1 Connectionist Structure 71
4.2.2 Inference Scheme 74
4.3 Learning Procedure 75
4.3.1 Label Generation 75
4.3.2 Rule Generation 79
4.3.3 Parameter Tuning 81
4.3.4 Complexity Analysis 83
4.4 Pedagogical Example 84
4.5 Experimental Studies 87
4.5.1 Simulation Setup 87
4.5.2 Nakanishi Regression Benchmark 87
4.5.3 Water Plant Monitoring 91
4.5.4 Acute Leukemia Diagnosis 97

4.6 Summary . . . 102

5 Dual Network Model for Knowledge Consolidation 103

5.1 Knowledge Consolidation . . . 103

5.2 Transient Network Model . . . 107

5.2.1 Connectionist Structure . . . 107

5.2.2 Learning Procedure . . . 109

5.3 Dual Consolidation Network . . . 112

5.3.1 Connectionist Structure . . . 112

5.3.2 Consolidation Procedure . . . 113

5.3.2.1 Awake Phase . . . 114

5.3.2.2 Sleep Phase . . . 115

5.3.3 Inference Procedure . . . 118

5.3.4 Complexity Analysis . . . 119

5.4 Pedagogical Example . . . 121

5.5 Experimental Studies . . . 127

5.5.1 Simulation Setup . . . 127

5.5.2 Translation Initiation Sites Prediction . . . 127

5.5.3 Face Image Detection . . . 134

5.6 Summary . . . 143

6 Concluding Remarks 144

6.1 Major Accomplishments . . . 144

6.2 Future Research . . . 147

A Yager Inference Scheme 151

A.1 Basic Concept . . . 151

A.2 Illustrative Example . . . 152

B Parameter Tuning Derivation 154

B.1 Consequent Part . . . 154

B.2 Antecedent Part 156

C Time Performance **159**

C.1 RFCMAC Time Performance 159

C.2 DCN Time Performance 160

D Publication List **161**

D.1 Journal Paper 161

D.2 Conference Paper 162

D.3 Book Chapter 163

D.4 Technical Report 163

Bibliography **164**

List of Figures

2.1 Design taxonomy of cognitive architectures 9
2.2 The SOAR cognitive architecture (adapted from [138]) 13
2.3 The EPIC cognitive architecture (adapted from [168]) 14
2.4 The Icarus cognitive architecture (adapted from [141]) 15
2.5 The Prodigy cognitive architecture (adapted from [279]) 16
2.6 The PRS cognitive architecture (adapted from [51]) 17
2.7 The IBCA cognitive architecture (adapted from [198]) 20
2.8 The Cortronics cognitive architecture (adapted from [105]) 21
2.9 The TSCA cognitive architecture (adapted from [1]) 22
2.10 The NuPIC cognitive architecture (adapted from [102]) 23
2.11 The GWCA cognitive architecture (adapted from [233]) 24
2.12 The ACT-R cognitive architecture (adapted from [13]) 27
2.13 The CLARION cognitive architecture (adapted from [254]) 29
2.14 The LIDA cognitive architecture (adapted from [73]) 30
2.15 The Dual cognitive architecture (adapted from [133]) 31
2.16 The Polyscheme cognitive architecture (adapted from [38]) 32

3.1 Human learning and memory systems (adapted from [121]) 41
3.2 Neural network models of memory consolidation 49
3.3 Outline of the Integrated Neuro-Cognitive Architecture (INCA) 51
3.4 Neuro-fuzzy system modeling approach 54
3.5 Dual consolidation network in INCA 57
3.6 The two operational cycles in INCA 60

4.1 Overview of the proposed RFCMAC system 72

4.2 A simple dataset with redundant input features 84
4.3 Example of label generation in RFCMAC 85
4.4 Example of rule generation in RFCMAC 86
4.5 Fuzzy labels crafted by RFCMAC-Yager for Nakanishi datasets 89
4.6 Results of RFCMAC-Yager on Nakanishi datasets 90
4.7 Fuzzy labels crafted by RFCMAC-Yager for 2-class water dataset (CV1) 95
4.8 Results of RFCMAC-Yager on 2-class water dataset 95
4.9 Fuzzy labels crafted by RFCMAC-Yager for leukemia dataset 98
4.10 Results of RFCMAC-Yager on leukemia dataset (independent test) . . . 99
4.11 ROC curves for the RFCMAC-Yager predictions on leukemia dataset . . 100

5.1 Overview of the transient network model 108
5.2 Signal pathways in the hippocampal system (adapted from [101]) 111
5.3 Overview of the proposed DCN framework 113
5.4 Consolidation procedure of the DCN system 116
5.5 Workflow of the DCN consolidation procedure for the AB-AC list task . . 121
5.6 Recall of *A*-*B* items after learning *A*-*C* items 122
5.7 Recall of *A*-*B* items after learning *A*-*B* and *D*-*B* items 123
5.8 Recall of *A*-*B* items after learning *A*-*B*, *D*-*B* and *A*-*C* items 124
5.9 Recall of *D*-*B* items after learning *A*-*B*, *D*-*B* and *A*-*C* items 124
5.10 Recall of *A*-*B* items after learning *A*-*B*, *A*-*C* and *D*-*B* items 125
5.11 Recall of *D*-*B* items after learning *A*-*B*, *A*-*C* and *D*-*B* items 125
5.12 Fuzzy membership functions crafted in FLM for the TIS data (CV3) . . 128
5.13 Fuzzy membership functions crafted in SLM for the TIS data (CV3) . . . 129
5.14 Performances of DCN for the TIS task in a single episode scenario 131
5.15 Performances of DCN for the TIS task in multi-episode scenario 133
5.16 Examples of face and non-face images in the CBCL database 136
5.17 Fuzzy membership functions crafted in SLM for the original CBCL data . 137
5.18 Interpretation of fuzzy rules in SLM with respect to the image data . . . 139
5.19 Performances of DCN for the original CBCL data 139

5.20 Fuzzy membership functions crafted in SLM for the extended CBCL data 140

5.21 Generalization performance of DCN for the extended CBCL data 142

6.1 Major accomplishments of the current research 145

A.1 Example of Yager inference in the RFCMAC-Yager system 153

List of Tables

2.1 Comparisons of the five representative symbolic architectures 18
2.2 Other prominent examples of symbolic cognitive architecture 19
2.3 Comparisons of the five representative emergent architectures 25
2.4 Other prominent examples of emergent cognitive architecture 26
2.5 Comparisons of the five representative hybrid architectures 33
2.6 Other prominent examples of hybrid cognitive architecture 33

3.1 Comparisons between INCA and other established cognitive architectures 67

4.1 Complexity analysis of the RFCMAC learning procedure 83
4.2 Parameter configurations of RFCMAC-Yager for the experiments 87
4.3 Fuzzy rules identified by RFCMAC-Yager for Nakanishi datasets 88
4.4 Benchmark results on Nakanishi datasets 92
4.5 Distribution of output classes in water plant dataset 93
4.6 Fuzzy rules of RFCMAC-Yager for 2-class water dataset (CV1) 94
4.7 Benchmark results on 2-class water dataset 96
4.8 Benchmark results on 3-class water dataset 96
4.9 Fuzzy rules of RFCMAC-Yager for leukemia dataset 97
4.10 Benchmark results on leukemia dataset (independent test) 100
4.11 Benchmark results on leukemia dataset (leave-one-out) 101

5.1 Complexity analysis of the DCN consolidation procedure 120
5.2 Parameter configurations of DCN for the experiments 127
5.3 Sample fuzzy rules in FLM for the TIS task (CV3) 128
5.4 Initial fuzzy rules in SLM for the TIS task (CV3) 129

5.5 Reorganized fuzzy rules in SLM for the TIS task (CV3) 130
5.6 Comparative results of various methods for the TIS data 132
5.7 The original and extended CBCL datasets used for the experiments . . . 135
5.8 Fuzzy rules in SLM for the original CBCL data 138
5.9 Comparative results of various methods for the original CBCL data . . . 140
5.10 Fuzzy rules in SLM for the extended CBCL data 141
5.11 Consolidated results of the DCN system for the extended CBCL data . . 142

C.1 Learning time of RFCMAC-Yager for Nakanishi datasets 159
C.2 Learning time of RFCMAC-Yager for water plant dataset 159
C.3 Learning time of RFCMAC-Yager for leukemia dataset 159
C.4 Consolidation time of DCN for TIS dataset 160
C.5 Learning time of DCN for MIT face dataset 160

Chapter 1

Introduction

1.1 Towards Artificial General Intelligence

Artificial Intelligence (AI), as described in [169], is about "making machines more fathomable and more under the control of human beings, not less." These words are worth remembering, seeing how current technological progress is rendering our environment more and more difficult to comprehend. This paradoxical situation results from the development of increasingly complex hardware or software tools and methods that fewer can master, apply and/or improve. Simplicity is much needed, and achieving it is a major challenge. There is a need to develop a more general kind of machine intelligence that will provide the necessary natural interaction and human-like cognition.

In recent years, cognitive neuroscience, psychology and AI have witnessed tremendous progresses, resulting in remarkable insights towards building an integrated framework for general, human-like machine intelligence. This will most likely be achieved by combining the best AI methods in innovative ways, while incorporating new ideas from biologically-inspired models to human factors and cognitive engineering. Ultimately, the objective is to create autonomous intelligent systems that can operate in varied domains and solve complex problems in a human-like way. Such systems would help achieve that crucial goal of AI to make machines easier to build and use. Moreover, they would allow to address the many critical issues in complexity, security, robustness, maintenance and usability of computer-based systems, on which the world's key infrastructures rest today.

The ongoing quest toward an integrated theory of machine intelligence has led to the investigation of *cognitive architectures* [186, 250, 200, 59], which constitute generic

blueprints for building intelligent agents, integrating and testing various computational models of knowledge representation, reasoning, learning, etc. Their main characteristic is to provide a holistic framework for general intelligence that can work across different task domains, in contrast to narrow AI techniques which focus on specialized solutions to well-defined problems (such as expert systems). The former type of intelligence is also known as strong AI or *Artificial General Intelligence* (AGI), i.e., a machine intelligence that can accomplish virtually any general intellectual tasks a human being can.

Numerous cognitive architectures have been proposed over the years, of varying types, abilities and complexities, but they are frequently used to model human performance in multi-modal situations rather than general intelligence. Of particular focus in this thesis are selected architectures that constitute potential candidates for building AGI, driven by the desire to realize a *unified theory of cognition* that encompasses all cognitive behaviors of humans [186]. One objective in devising such architecture is to push its limits, so that it can develop a generic capability rather than resorting to some problem-specific tools, as classical AI does. In many ways, this would involve formalizing and simulating the mechanisms of human cognition, building an *artificial mind* using human as model [59]. Notably, the vast body of experiments in human cognition provides both a measure to gauge model performance (because previous results should be satisfactorily reproduced and explained) and an inspiration for architectural enhancements.

One research issue in cognitive architectures is concerned with the methods by which they can attain general intelligence. A school of thought proposes that intelligence can only be achieved by modeling first the brain architecture, and in turn the processes that will produce the desired behaviors. Another approach is to develop architectures that can behave intelligently without regard to their psychological plausibility. Interestingly, while much plausibility may be needed to achieve human-like intelligence, it can produce unwanted side-effects, such as cognitive biases or operational limitations. Another related issue is about what AGI should encompass, and the precise specifications of cognitive architectures supporting it. As the plethora of existing approaches clearly show, there is an emerging need to formulate reliable tests and benchmarks and, more generally, a

detailed roadmap to AGI. Section 1.2 presents some of the major challenges in current AI research that pertain to the roadmap to AGI.

1.2 Current Challenges

AI research has focused on specialized problem-solving approaches which are useful for developing expert systems for particular task domains. However, it is difficult to apply and extend such techniques to create AGI. Language domain is one area where AI shows limited successes. Restricted form of the Turing test (the full test being too hard to try), called Loebner Prize competition [276], has so far been won by chatterbots that are based on template matching, or more recently, contextual pattern matching methods. Such programs have little chance to develop real understanding of language, though can be used as a stereotyped question and answer system. Recently, Carpenter and Freeman [36] proposed a personal Turing test, where one tries to guess if the conversation is done with a program or a real known person. Humans are able to impersonate other people, and thus the personal Turing test may be an interesting landmark towards AGI.

Feigenbaum [68] proposed as a grand challenge to build a super-expert system in narrow domains. This appears to go in the direction of specialized intelligence, but one may argue that a super-expert without general intelligence required for communication with humans is not feasible. Sophisticated reasoning by human experts and AI systems in such fields as mathematics, bioscience or law may be compared by a panel of experts who pose problems, raise questions, and ask for further explanations to probe the understanding of the subject. A good example of such challenge is provided by the Automated Theorem Proving (ATP) system competition [207, 257] in many subcategories. Developing general theorem provers, perhaps using meta-learning techniques that rely on many specialized modules, would be an interesting step towards AGI in mathematics. Super-experts also face great challenges in the automatic curation of genomic/pathways databases and creation of models of genetic/metabolic processes for various organisms, since handling large amount of information in such databases is far beyond human capacities.

Defining similar challenges and milestones towards AGI in other fields is certainly

worthwhile. The goal is to devise programs that will advice human experts in their work, evaluate their reasoning, and perhaps add some creative ideas. Nilsson [192] has argued for the development of general-purpose educable systems that can be taught skills to perform human jobs, and to measure which fraction of these jobs can be performed by an AI system. Building such a system replaces the need for many specialized systems, as Turing noted in [276], where he proposed a "child machine" in his classical paper. Some human jobs are knowledge-based and can be done by information processing systems, where progress may be measured through passing a series of examinations. But most jobs involve manual labor, requiring sensorimotor coordination that should be mastered by household robots or autonomous vehicles. Creation of partners or personal assistants, rather than complete replacements for human workers, may be treated as a partial success. Unfortunately, specific milestones for this type of applications are not yet precisely defined. Some ordering of different jobs based on how difficult they can be learned may be worthwhile. In fact, many jobs are already fully automated, reducing the number of people in manufacturing, financial services, etc. In most cases, alternative organization of work is credited for reduction in the number of jobs (plant and factory automation, bank ATMs, vending machines, etc), not due to deployment of AI systems.

A roadmap to AGI should therefore be based on detailed analysis of the relationships between various functions that should be implemented, system requirements to achieve these functions, and classes of problems that should be solved at a given stage. The importance of cognitive architectures in this enterprise is to provide a comprehensive and systematic framework for modeling human intelligence, which allows developers to formulate and think clearly in terms of the mechanisms and/or processes available within the framework that are not specifically designed for one task domain. Nevertheless, to achieve AGI, cognitive architectures require further elaboration and experimental validation of key features, such as *consolidation* mechanisms for robust self-organization of knowledge, *scalability* for dealing with task domains of varying complexities, and *metacognitive functions* for realizing autonomous reflective system that can bootstrap its performances over time, thereby contributing back to our understanding of human cognition.

1.3 Main Contributions

Our own take in relation to the roadmap to AGI consists of building a novel *Integrated Neuro-Cognitive Architecture* (INCA), which aims at addressing three salient features of general intelligence, i.e., knowledge consolidation, scalability, and metacognition, crucial for developing an intelligent machine capable of operating autonomously and robustly in a way similar to that of humans. In comparison to contemporary cognitive architectures, the proposed INCA is unique in that it features systematic and cognitively plausible *consolidation* and *inference* protocols to achieve effective acquisition and exploitation of knowledge, respectively. In turn, such mechanisms provide a means for building higher-level (meta) cognitive functions required to achieve a fully autonomous system. It must be noted, however, that the goal of this thesis is not to realize a complete AGI system, but rather formulate and evaluate basic computational models (especially with regard to consolidation process) that can be used as key building blocks in an AGI system.

The following points summarize the main contributions of this thesis:

- **Critical survey on contemporary cognitive architectures**. The first key contribution of this thesis is to provide a systematic and critical survey of the state of the arts in cognitive architectures, which facilitates good understanding of the contemporary approaches and useful insights for developing AGI. The survey begins by outlining a fundamental taxonomy of cognitive architectures, followed by critical reviews on and comparisons among the most renowned architectures of each category. Several recommendations and promising research directions are also laid out, with emphasis on the knowledge consolidation, scalability, and metacognition capacities that form the core focus of the INCA framework.

- **Brain-inspired integrated neuro-cognitive architecture**. Another major contribution of this thesis consist of formulating INCA as a novel generic framework supporting knowledge consolidation, scalability, and metacognitive functions. In essence, the proposed architecture emulates the putative, functional learning and memory aspects as well as the interactions among the prominent brain systems.

The latter is realized in the form of two inter-module communication protocols, namely consolidation and inference cycles, which support robust scalable knowledge self-organization and exploitation processes, respectively, and govern the overall behaviors of INCA. Discussion on the cognitive plausibility of the proposed INCA framework is also given, and its design merits are highlighted through feature comparisons with several other prominent cognitive architectures.

- **Reduced localized network model for knowledge extraction**. As an initial endeavor in the development of the INCA framework, a reduced rule-based localized model of cortical system in the brain has been constructed that implements the long-term memory modules in the architecture. Specifically, the model emulates the two-stage development of human cortical memories to construct and reduce the system's knowledge base, respectively. These two mechanisms provide the model with several key benefits, including discovery of highly concise and intuitive rules, excellent generalization performances, and enhanced system scalability. These features make it possible to use the proposed model in complex task domains, and to eventually realize a large-scale memory system necessary for AGI. These capacities have been verified through simulation studies on nonlinear regression benchmark, water plant monitoring, and high-dimensional leukemia diagnosis tasks.

- **Dual network model for knowledge consolidation**. As mentioned, pivotal to the development of INCA is the ability to consolidate knowledge acquired from the environment, that allows the system to operate in a robust and efficient manner. A dual network model has been developed for this purpose by integrating a fast-learning, transient learning memory, for rapid acquisition of novel information without catastrophically interfering the existing knowledge. The model simulates the complementary interaction between the hippocampal and cortical circuits in the brain, which constitute the key facilitators for robust consolidation of human memory. Also featured in the model is multi-episode consolidation scheme to sequentially learn from large data patterns and adapt to the environmental dynamics. A series of experiments, from basic categorization task to large-scale translation

initiation sites prediction and face detection tasks, have demonstrated the consolidation, scalability, and generalization features of the proposed model.

1.4 Thesis Organization

The remainder of this thesis consists of five chapters. Chapter 2 presents the survey on contemporary cognitive architectures that forms a background for the work presented in this thesis. Emphasis is given on the design properties and key merits and limitations of the architectures reviewed. Several directions are then laid out to conclude the survey.

Chapter 3 elaborates the INCA framework, starting with discussion on some brain inspirations and computational models that are related to its construction. The architectural organization and communication protocols of the INCA modules are then described, as well as the two operational cycles governing the overall INCA behaviors. The chapter concludes with discussion on the cognitive plausibility of the proposed framework, and comparison of features with several other renowned cognitive architectures.

Chapter 4 describes the reduced localized model of the cortical system in the brain that serves as a basic infrastructure for realizing the long-term memory modules in INCA. The chapter begins with background information about related knowledge extraction methodologies, followed by description of the architecture of the localized model and its detailed learning procedures. Simulation results on nonlinear regression benchmark, water plant monitoring, and leukemia diagnosis tasks are then reported to verify the scalability and interpretability traits of the model as a knowledge extraction tool.

Chapter 5 presents the dual network model that simulates the knowledge consolidation mechanisms in the human brain. An overview of neurocognitive inspirations related to knowledge consolidation is first provided. Next, the chapter describes the architecture and learning procedures of the proposed model. Experiments on large-scale biomedical and face datasets are then presented to showcase the ability of the model to achieve consolidation in the face of large tasks, which often involve multiple learning episodes.

Chapter 6 finally concludes this thesis, with emphasis on the key achievements of the current research, and discussions on the future development and applications of INCA.

Chapter 2

Cognitive Architectures: A Survey

2.1 Basic Taxonomy

A critical survey on cognitive architectures is given in this chapter. Newell [186] proposed 12 criteria for evaluating cognitive systems: adaptive behavior, dynamic behavior, flexible behavior, development, evolution, learning, knowledge integration, vast knowledge base, natural language, real-time performance, self-awareness, and brain realization. These criteria have been analyzed and applied to several architectures [14], but such fine-grained categorization makes their comparisons difficult. On the other hand, a number of surveys [166, 140, 280] have been published that describe the organization and working mechanisms of several prominent architectures. However, these surveys generally lack a critical evaluation and comparability of the key benefits and issues of the architectures. This chapter presents instead a simpler taxonomy, gives examples of different architecture types, and provides critiques and recommendations for better systems.

The two most fundamental features in the development of any cognitive architecture, as widely established in cognitive neuroscience and psychology, are *memory* and *learning*. Memory denotes the repository for background knowledge about the world and oneself, while learning is the key process responsible for shaping this knowledge [121, 10]. Together learning and memory form the indispensable substrates for higher-order intelligence, such as decision making, planning, executive regulation, and creativity. A simple taxonomy based on these two features categorizes existing cognitive architectures into three groups: *symbolic*, *emergent*, and *hybrid*, as illustrated in Figure 2.1. Their memory and learning aspects are elaborated in sections 2.1.1-2.1.3 respectively.

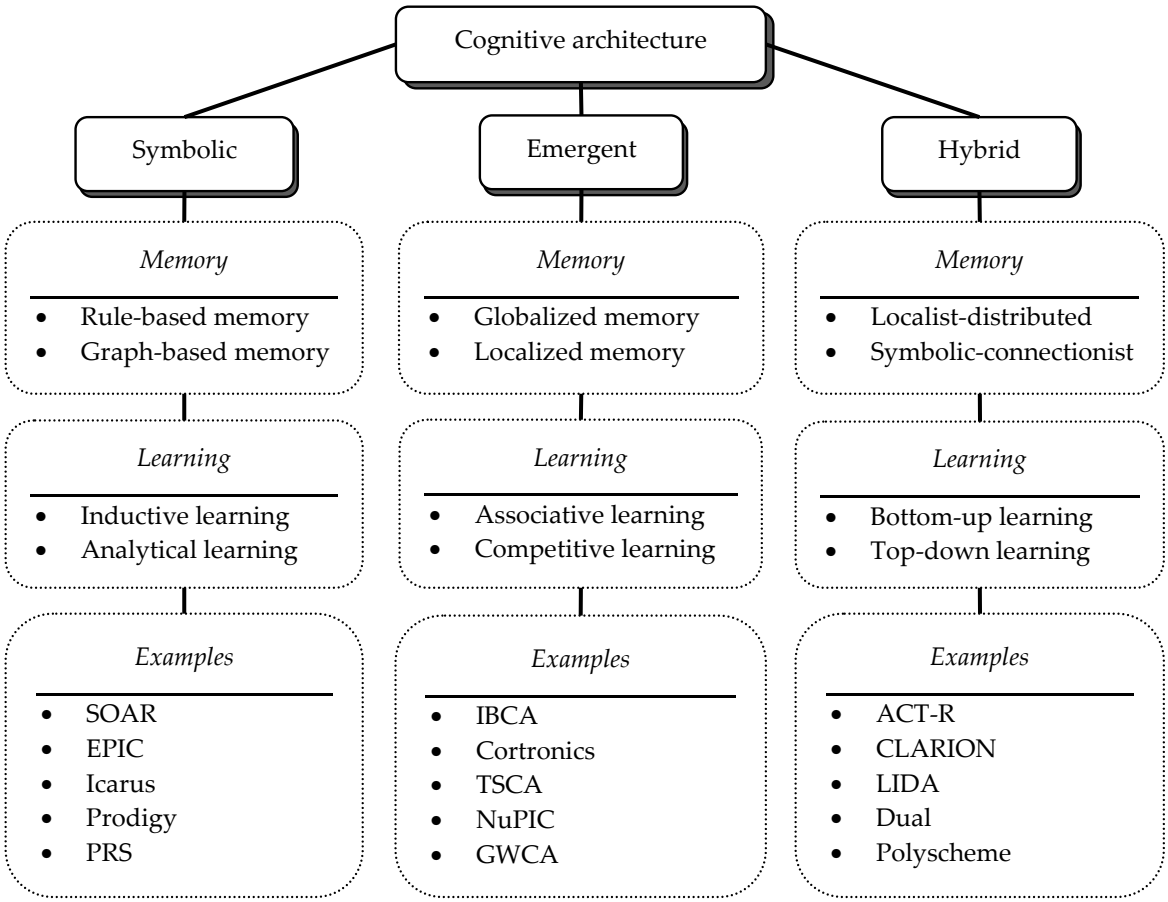


Figure 2.1: Design taxonomy of cognitive architectures

2.1.1 Symbolic Paradigm

The development of symbolic cognitive architectures is typical of classical AI and as such has been largely based on a top-down design approach, using symbols as the key means to capture explicit domain knowledge and to support information processing and interactions with the world [9, 186]. The majority of these architectures utilize a centralized control over the information flow from sensory inputs through memory to motor outputs. This approach is founded on the well-established *physical symbol system* hypothesis [187], stating that intelligent behavior can only be achieved through a physical symbol system with the ability to input, output, store and alter symbolic entities, and carry out appropriate actions to reach some goals.

Broadly, the memory organization of symbolic architectures consists of two types of representations: *rule-based* and *graph-based*. Rule-based representations of perception-action memory are used in knowledge-based systems for building programs that embody

the logical reasoning skills of human experts, such as production systems [186]. Graph-based memory is typically encoded as a directed graph comprising nodes (for symbolic entities or concepts or states and their attributes) and edges (for their relationships), altogether providing a way for efficient retrieval and inference of knowledge from memory. Main examples of this sort are frames/schemata [172], conceptual graphs [246], and reactive action packages (RAPs) [71]. While these approaches differ in their formalism and implementations, the underlying paradigms remain very similar or even identical.

Substantial efforts have been made over the years to introduce *analytical* and *inductive* learning to symbolic systems. The former aims at exploiting existing general or specific facts to infer other facts that they entail logically. Prominent examples include explanation-based learning (EBL) [50, 175] and analogical learning [46, 278]. On the other hand, inductive learning seeks to derive from specific facts or examples the more general ones that capture the underlying domain structure. Well-known exemplars of this type include knowledge-based inductive learning (KBIL) [143, 176], decision tree learning [217], Bayesian learning [27], and reinforcement learning [238, 284].

2.1.2 Emergent Paradigm

The design of emergent architectures is generally based on a bottom-up modeling approach of low-level activation signals flowing through a network of processing elements (PEs). The approach is largely inspired by the neuroscience findings about the emergent nature and working principles of the brain, leading to the connectionist ideas to designing intelligent machines that mimic those very brain processes. In this framework, PEs form network nodes that can self-organize and interact with one another in specific ways, modify their internal states, and capture properties of interest [163].

There are two complementary approaches to the memory organization of emergent architectures: *globalized* and *localized*. The Multi-Layer Perceptron (MLP) [163, 107] and other neural networks based on delocalized transfer functions process information in a distributed, global way. The output signals for a given input are affected by all network parameters, yielding good generalization of responses to novel stimuli. The Cerebellar Model Articulation Controller (CMAC) [4], Self-Organizing Map [132], and the basis set

expansion networks using localized functions (e.g. Gaussians) are examples of localized systems; the output signals for a given input rely only on a small subset of activated units. It should be noted that a modular organization of globalized network will easily create subgroups of PEs that react in a local manner.

The learning methods for emergent architectures are quite diverse. *Associative learning* maps specific input representation to some output representation, and remembers the reactions, heteroassociations, or enables pattern completion (autoassociations). In this, learning is guided directly by a set of correct target signals, e.g. back-propagation [163], or indirectly by critic signals, e.g. Q-learning [284], corresponding to the supervised and reinforcement learning paradigms respectively. In *competitive learning*, all PEs compete to become active and learn in an unsupervised fashion. The simplest form of this learning is the winner-takes-all (WTA) rule, where only one winning PE (or one per group) learns at a time and inhibits others losing the competition [94, 132]. Another example is Hebbian learning that captures the statistical properties of incoming signals, creating an internal model of the environment [103]. Evolutionary learning constitutes yet another example involving competition within a population of (random) potential solutions [89].

2.1.3 Hybrid Paradigm

The relative merits of the symbolic and emergent paradigms show that they are very complementary in nature. On the one hand, symbolic architectures can capture high-level knowledge and cognitive functions, such as planning and deliberative reasoning. However, they generally lack the means to learn symbolic entities from low-level information and to cope well with large amount of information or uncertainty. On the other hand, emergent architectures can better capture the context-specificity of human performance and handle low-level information concurrently. Yet the difficulty in deriving higher-order functions remains a major issue in this approach. By integrating both approaches, such that each can remedy the deficiencies of the other, one can envisage a comprehensive human-like architecture covering all levels of processing, from stimuli to high-level cognition.

Research in this area has led to several proposals of hybrid cognitive architectures, which can be roughly divided into two classes according to the memory type of the con-

stituent modules: *localist-distributed* and *symbolic-connectionist* [250]. The first approach comprises a combination of localist modules (where each concept is specified by one PE node) and distributed modules (with each concept denoted by a set of non-exclusive, overlapping nodes). The second class involves a mixture of symbolic modules (i.e., rule- or graph-based memory) and connectionist modules (either localist or distributed). Some of these hybridization approaches can be traced back to the idea of Smolensky [245], who coined the dichotomy of conceptual and sub-conceptual cognition.

Correspondingly, hybrid architectures can be categorized into two classes based on their direction of learning: *top-down* and *bottom-up learning* [254]. The former involves transition of knowledge from explicit/accessible, conceptual level to implicit/inaccessible, sub-conceptual level, whereas the latter goes from sub-conceptual to conceptual level. Top-down learning can be achieved by precoding a set of expert rules at the top level (e.g. localist or symbolic module) and then allowing the bottom-level (e.g. distributed ANN) to learn by observing actions guided by the top-level [155, 254]. Conversely, bottom-up learning may be accomplished via extracting or transforming implicit knowledge coded by a bottom-level module into a set of conceptual rules [252, 254].

The next three sections review several representative symbolic, emergent and hybrid cognitive architectures that constitute potential candidates for AGI and have a community of experts working on them. For each architecture, an introduction of its design focus, memory organization, and learning methodologies is presented. Several remarks are then given highlighting their merits and issues. A summary of their feature comparisons is also provided, followed by a list of several other similar/related architectures.

2.2 Symbolic Architectures

2.2.1 Representative Examples

2.2.1.1 SOAR

The *State, Operator And Result* (SOAR) is a classic expert rule-based architecture for modeling general intelligence [139, 138]. Based on theoretical framework of knowledge-based systems seen as an approximation to physical symbol systems, SOAR represents its

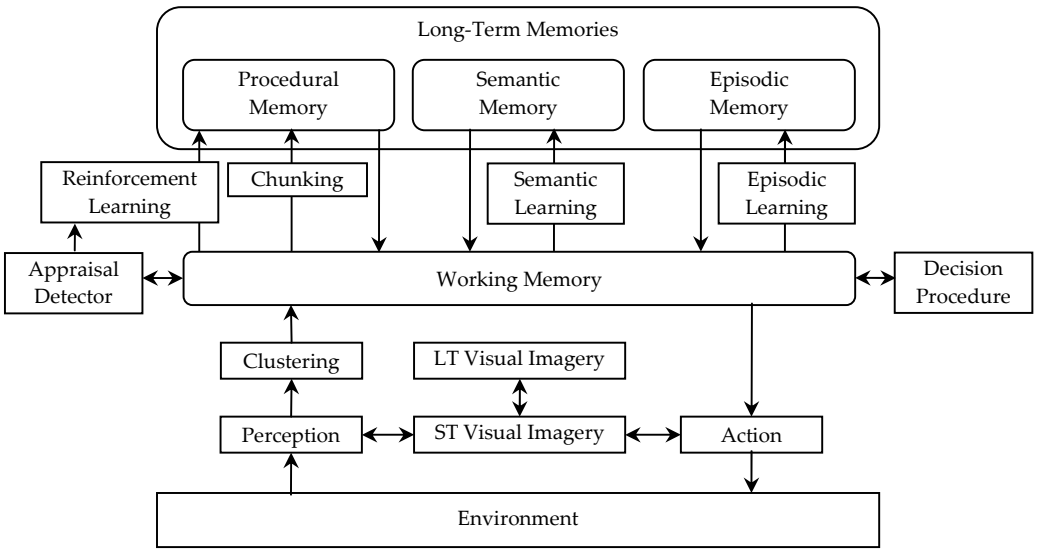


Figure 2.2: The SOAR cognitive architecture (adapted from [138])

knowledge using production rules, arranged in terms of operators acting in the problem space i.e. the set of states representing the task at hand. The primary learning in SOAR is termed *chunking*, a type of analytical EBL technique for formulating rules from problem solving traces [139]. Recently, many extensions of SOAR have been proposed, as per Figure 2.2; it includes a semantic memory (and learning) to store general facts about the world, an episodic memory (and learning) to store specific events, a reinforcement learning to adjust the preference values for operators, visual imagery, emotions, moods and feelings used to speed up reinforcement learning and direct reasoning [138].

SOAR has demonstrated a variety of high-level cognitive functions processing large and complex rule sets, involving planning, problem solving and natural language comprehension in real-time distributed environments [138]. The current design of its perceptual-motor system, however, is fairly unrealistic as it requires users to define their own input and output functions. SOAR has not yet integrated all the planned extensions at the moment, and a few more like memory decay/forgetting, attention and information selection, learning hierarchical representations, or handling uncertainty and imprecision will also be very useful. It also remains to be seen how well numerous problems that face such extensions can be handled using the old architecture as a base.

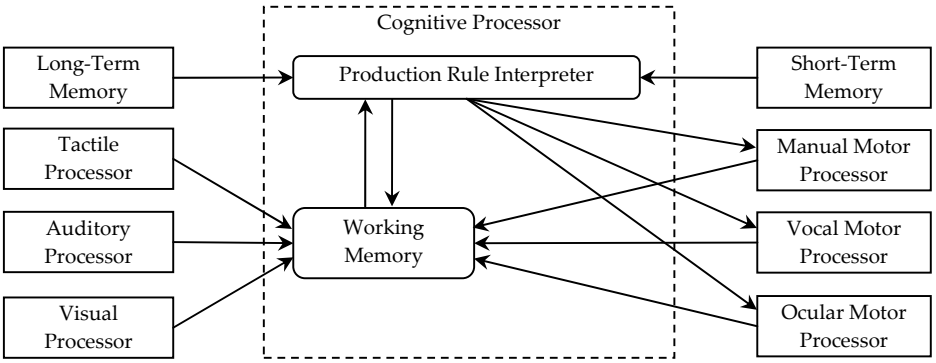


Figure 2.3: The EPIC cognitive architecture (adapted from [168])

2.2.1.2 EPIC

The *Executive Process Interactive Control* (EPIC) is a symbolic architecture for building models that subsume aspects of human performance [168]. It aims at capturing human perceptual, cognitive and motor activities via interconnected processors working in parallel, and building practical models of human-computer interaction. The EPIC memory comprises production rules for cognitive processor and a set of perceptual (visual, auditory, tactile) and motor (manual, ocular, vocal) processors operating on symbolically-coded features (instead of raw sensory data), as per Figure 2.3. Symbolic instructions from the cognitive processors are sent to the motor processors, giving movement specifications and timing details. No learning mechanism is employed in EPIC at the moment.

As all processors can run in parallel, EPIC offers a good support to perform multi-task coordination and perceptual-motor activities through executive control. Another merit is that it is fairly simple to program, due to its design simplicity. EPIC focuses on multiple simple tasks, but in one experiment it has been connected to SOAR for problem solving, planning and learning, and the EPIC-SOAR combination has been applied to air traffic control simulation [224]. One issue in EPIC, however, is the massively parallel rule selections and executions within the system, which is cognitively implausible as knowledge may be acquired without redirection or disruption of attention from the main task. Another limitation is the lack of accounts on intermediate and longer-term memories crucial in human cognition. Also, the perception-symbol mapping within the architecture seems "hardwired", yielding rather unrealistic (pre-)attentive comprehension.

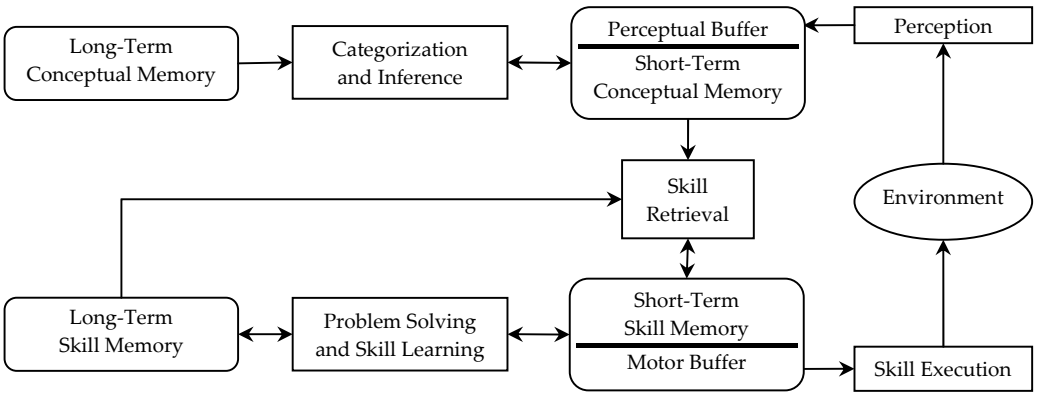


Figure 2.4: The Icarus cognitive architecture (adapted from [141])

2.2.1.3 Icarus

Icarus is an integrated architecture for physical agents, with knowledge specified as reactive skills denoting goal-relevant reactions to a class of problems [140, 141]. It includes a perceptual module, a skill execution module, a problem solving/skill learning module, a categorization/inference module, and several types of memory, as shown in Figure 2.4. Concepts are matched to percepts in a bottom-up way, and goals to skills in a top-down way. Conceptual memory contains knowledge about general classes of objects and their relations, while skill memory knowledge about how to do things. Each comprises a long-term and a short-term memory. The former has a hierarchical organization, with conceptual memory directing bottom-up, percept-driven inference and skill memory controlling top-down, goal-driven action selection. Icarus acquires knowledge via hierarchical reinforcement learning that propagates reward values backward through time.

These hierarchical traits enable Icarus to focus its attention on a specific object/event in its sensor range and reduce the reaction time or search space. Its planning and memory modules also allow to efficiently acquire new concepts by constructing a feature tree that the system can comprehend. Relational reinforcement learning that gives priority to high-utility beliefs and rapidly finds the most useful inferences is able to handle large memory hierarchies [141]. Interesting applications to in-city car driving, blocks world or free-cell solitaire have been demonstrated. However, an outstanding issue is the lack of concurrent processing to handle multiple sensory inputs while coordinating resources/actions across different modalities. Issues related to uncertainty have also been largely ignored.

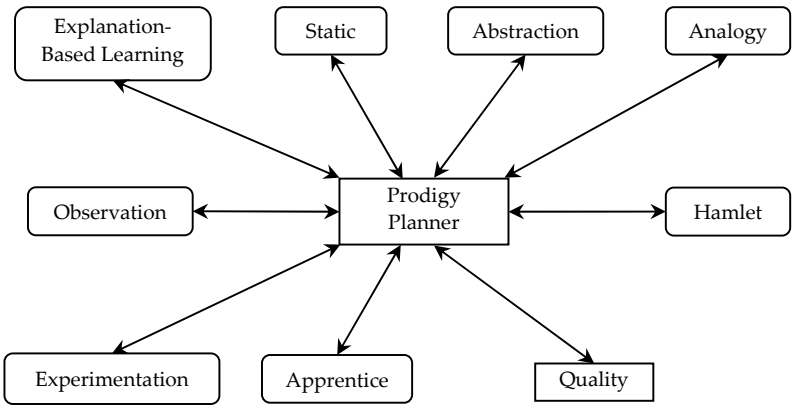


Figure 2.5: The Prodigy cognitive architecture (adapted from [279])

2.2.1.4 Prodigy

Prodigy is a modular architecture that integrates planning with multiple learning methods [35, 279]. Distinction between domain/declarative and control/procedural knowledge is stressed, both represented as conceptual graphs. Control knowledge is gained via an EBL module that derives rules from problem solving traces, a static module that analyzes domain description before planning, an abstraction module that breaks a domain description into multiple abstraction levels, and an analogy module that solves a problem using similar previous solutions, as per Figure 2.5. Domain knowledge is learned via an experimentation module that refines a domain description via inductive learning, an observation module that accumulates domain knowledge by observing expert agents or own practice, and an apprentice module that offers a user interface to evaluate and guide planning/learning. Several modules are also included for plan quality improvement.

A primary feature of Prodigy is the modularity imposed by the domain-control knowledge distinction, making it easy to insert or alter new knowledge of each type and realize different planning strategies. Prodigy has been successfully applied to robot navigation, blocks world, algebra manipulation, scheduling, and logistics planning [35, 279]. One main limitation, though, is its opaque control knowledge, where control points (subgoal ordering) are hidden from the learning modules. The EBL module also relies heavily on complete domain knowledge, which is often unavailable in practice. The dependency of its learning modules on the internal structures of the planner makes its extension to more complex tasks rather difficult as well.

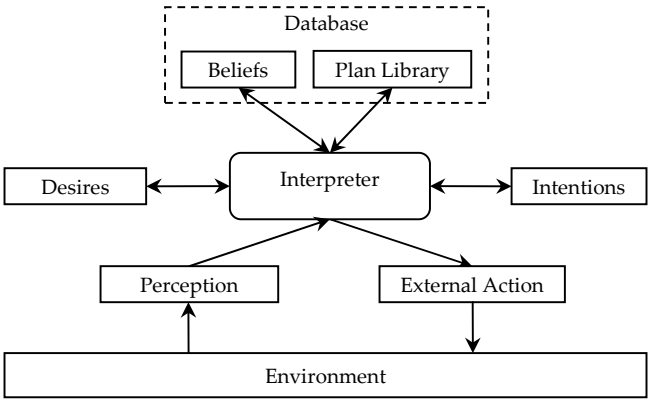


Figure 2.6: The PRS cognitive architecture (adapted from [51])

2.2.1.5 PRS

Procedural Reasoning System (PRS) is a symbolic architecture for encapsulating procedural knowledge and building generic reactive reasoning systems [111, 51]. Procedural knowledge is encoded as expert rules specifying actions to take in specific situations. PRS has its conceptual root on the *belief-desire-intention* (BDI) framework [31]; it has a database comprising current beliefs about the world and a library of plans/procedures to devise action sequences to reach specific goals or react in some situations, a set of desires/goals to be accomplished, and an intention module for plans chosen for future executions, as per Figure 2.6. An interpreter is also included to manage these modules to select/execute appropriate plans towards some goals. PRS performs analytical EBL learning as it interacts with the world via its database (to acquire new beliefs in response to environmental changes) and via its intention module (to carry out actions).

PRS features support for attention shifting and is able to achieve goal-directed tasks while being responsive to incoming events. It also enables multiple agents to work in parallel via asynchronous message passing to solve problems in a cooperative manner. Moreover, it supports meta-level (reflective) reasoning mechanisms to realize complex operations on top of the basic PRS cycle. PRS has been employed in space shuttle control, mobile robot, and air traffic management [111, 51]. Nevertheless, it still lacks a systematic means to translate its abstract design into practical models, and the specifications of the models are often ad-hoc. Its scalability are also not yet demonstrated, as its applications have so far been limited to resource-bounded practical reasoning.

Table 2.1: Comparisons of the five representative symbolic architectures

Aspect	Category	SOAR	EPIC	Icarus	Prodigy	PRS
Memory	Rule-based memory	DE,PR	DE,PR	DE,PR	-	PR
	Graph-based memory	-	-	-	DE,PR	-
Learning	Inductive learning	RE,EP,SE	-	RE	EX,AB,AP	-
	Analytical learning	EB	-	-	EB,AN,ST	EB
Capability	Perception	Yes	Yes	Yes	-	Yes
	Planning/problem solving	Yes	-	Yes	Yes	Yes
	Reactivity	Yes	Yes	Yes	-	Yes
	Executive regulation	Yes	Yes	-	Yes	Yes
	Emotion	-	-	-	-	-
	Parallel processing	-	Yes	-	-	Yes
	Hierarchical processing	Yes	-	Yes	-	-
	Language comprehension	Yes	-	-	-	-
Application	Cognitive tasks and games	[139, 186, 193]	[127, 168]	[140, 142]	[173, 35]	[51]
	Conversation and tutoring	[147, 220, 259]				
	Process control and robotics	[264, 113, 291]	[224]	[140, 141, 42]	[98, 279]	[84, 111, 83]

DE = declarative, PR = procedural, RE = reinforcement, AB = abstraction, EB = explanation-based
EP = episodic, SE = semantic, EX = experimentation, ST = static, AN = analogy, AP = apprentice

2.2.2 System Comparisons

Table 2.1 presents a comparative list of features to summarize the memory organization and learning methodologies of the five exemplar architectures described in section 2.2.1. For each memory type (i.e., rule- or graph-based), a further distinction is made between *declarative* and *procedural* memory, which are concerned with knowledge about facts and events and about working skills or procedures, respectively [9]. Comparisons are also made in terms of supported architectural capabilities, including: *perception* (ability to recognize objects in the world), *planning/problem solving* (ability to devise sequence of actions to reach some goals), *reactivity* (ability to handle unforeseen events in a bounded time), *executive regulation* (ability to monitor and alter own cognitive processes), *emotion* (ability to internally produce affective signals to direct cognitive processes), *parallel processing* (ability to process many pieces of information simultaneously), *hierarchical processing* (ability to decompose (abstract) tasks into (from) smaller subtasks), *language comprehension* (ability to interpret or communicate using natural language). Finally, the application areas of the architectures are presented, comprising three groups: *cognitive tasks and games*, *conversation and tutoring*, and *process control and robotics*.

Table 2.2: Other prominent examples of symbolic cognitive architecture

Architecture	Aspect	Description
3-Tier [29]	Design focus	Integration of planning, sequencing and control in robots
	Memory	Graph-based (reactive action package)
	Learning	Not specified
Remote Agent [177]	Design focus	Autonomous, flexible agent for space robot explorations
	Memory	Graph-based (reactive action package)
	Learning	Not specified
Oscar [212]	Design focus	Rational agent for epistemic and practical cognitions
	Memory	Graph-based (inference graph)
	Learning	Deductive (explanation-based learning)
NARS [281]	Design focus	Unified theory and model for representation and reasoning
	Memory	Rule-based
	Learning	Deductive (explanation-based learning)
CIRCA [178]	Design focus	Intelligent agent architecture for real-time control system
	Memory	Rule-based
	Learning	Not specified
Theo [174]	Design focus	Self-improving agent for learning and problem solving
	Memory	Graph-based (frame)
	Learning	Deductive (caching + explanation-based), inductive (knowledge-based)
SNePS [235]	Design focus	Representation/reasoning agent for language comprehension
	Memory	Graph-based (frame + semantic network)
	Learning	Deductive (explanation-based)

2.2.3 Other Architectures

Over the years, a variety of other symbolic architectures have been developed that are inspired by, or related to, the architectures presented in section 2.2.1. Table 2.2 summarizes some of these architectures in terms of their design focus, memory organization, and learning procedure. In conclusion, while these architectures exhibit certain features that make them unique, they bear some important similarities in their design principles and approaches, especially with regard to memory representation and learning procedure. In addition, although these architectures can accomplish high-level cognitive functions, they put little reference (if at all) to their biological validity, particularly with regard to the salient brain traits such as self-organization, distributed processing, etc. These aspects are addressed in the emergent approach, as will be elaborated in section 2.3.

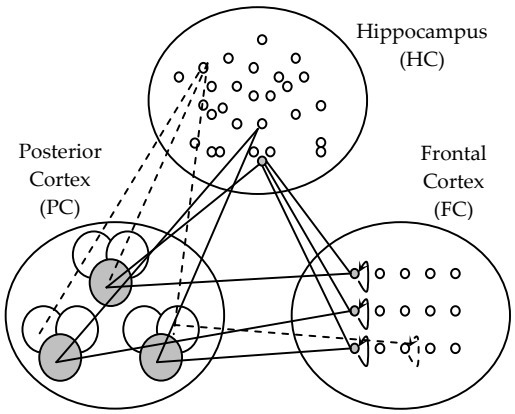


Figure 2.7: The IBCA cognitive architecture (adapted from [198])

2.3 Emergent Architectures

2.3.1 Representative Examples

2.3.1.1 IBCA

The *Integrated Biologically-based Cognitive Architecture* (IBCA) is a large-scale emergent architecture that emulates the automatic and distributed information processing in the brain [198, 200]. Three brain regions are emphasized, as illustrated in Figure 2.7: posterior cortex (PC), frontal cortex (FC), and hippocampus (HC). The PC module assumes an overlapping, distributed organization focusing on sensory-motor and multi-modal hierarchical processing. The FC module employs a non-overlapping, recurrent organization whereby working memory units are isolated from one another and contribute combinatorially. The HC module has a sparse, conjunctive organization where all units contribute interactively (not combinatorially), enabling rapid binding of activation patterns across PC and FC while reducing interference. The LEABRA learning algorithm includes error-driven learning of skills and Hebbian learning with inhibitory competition dynamics. In this, the PC and FC modules employ a slow learning that integrates many experiences to capture the salient regularities of the environment, while the HC module a fast learning that retains and separates individual experiences. The cooperation between HC and FC/PC reflects chiefly the complementary learning paradigms in the brain.

IBCA posits the benefits of generalization, parallelism and flexibility of the content-specific, distributed representation in the brain, missing in symbolic models. Its complementary learning helps resolve the catastrophic forgetting issues in connectionist sys-

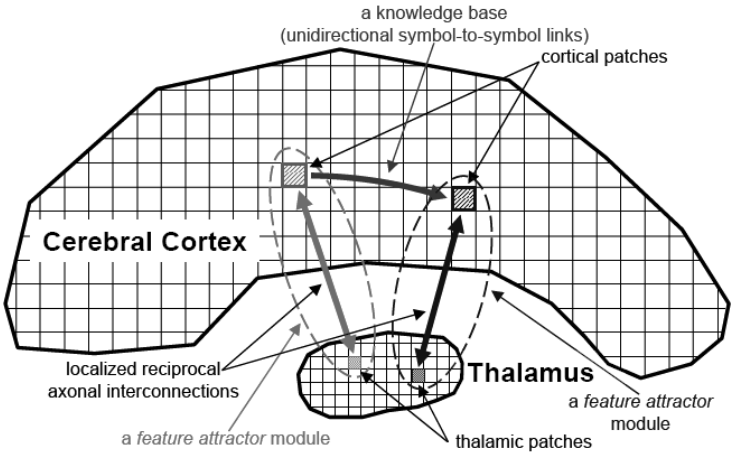


Figure 2.8: The Cortronics cognitive architecture (adapted from [105])

tems. Higher-level functions, such as variable binding, chaining of operations, can emerge from invocation, maintenance, and updating of active representations in the FC module. These capacities have been validated in basic psychological tasks, e.g. Stroop test, dynamic sorting task, visual feature binding [200]. However, the IBCA's large, fine-grained structure raises several scalability issues. Its learning is currently limited to the weight parameters only, not the structure/topology. While IBCA can explain human behaviors probed by psychological studies, its capacity in tasks requiring high-level reasoning (that most symbolic architectures can already achieve) is not yet demonstrated.

2.3.1.2 Cortronics

Cortronics is an emergent architecture that models the biological functions of the thalamocortex system in the brain [104, 105]. Its memory comprises modular feature attractors called lexicons, each composed of a localized cortical patch, a localized thalamic patch, and their reciprocal links, as per Figure 2.8. A lexicon realizes a stable set of attractor states termed symbols, each represented by a group of neurons. In turn, knowledge in Cortronics takes the form of parallel, indirect unidirectional links between the neurons denoting one symbol in a lexicon and those denoting a symbol in another. A competitive activation of symbols of lexicons, called *confabulation*, is used to learn and retrieve information. The states of neurons involved evolve dynamically via their parallel, mutual interactions, and the minority that ends up in the excited state denote the conclusions (symbols that won the competition) or a null symbol ("don't know" answer). The model

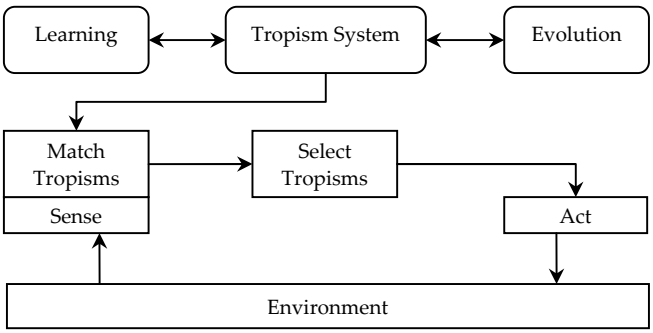


Figure 2.9: The TSCA cognitive architecture (adapted from [1])

can thus predict/anticipate the next state, e.g. move or word that should follow.

The confabulation operation provides a fast decision-making mechanism to find the best conclusion for a universally-applicable type of probabilistic questions. It also has the merit of the massively parallel application of knowledge and interoperability of knowledge links across attributes (i.e., multi-confabulation). These capacities have been showcased in several studies in language domain [105]. However, it is not clear how such architecture can be extended to create an AGI, as confabulation is not sufficient for reasoning with complex knowledge. Nevertheless, it is an interesting process involved in anticipation, imagination and creativity, occurring at a shorter time scale than reasoning processes.

2.3.1.3 TSCA

The *Tropism System Cognitive Architecture* (TSCA) focuses on autonomous control of team of robots, so that they can operate robustly while adapting to the environment and improving their performance [1, 2]. As shown in Figure 2.9, TSCA has a tropism module, a learning module, and an evolution module. The first constitutes the core TSCA memory, while the last two facilitate the system's adaptation. The tropism module is realized using a globalized ANN that maps the entities in the world and their states to action and represents the agent's likes (positive tropisms) and dislikes (negative tropisms). The learning module performs ontogenetic learning on each agent to insert, modify, or omit its tropism elements via stochastic reinforcement learning that adapts from perception, success and failure. The evolution module carries out phylogenetic learning via genetic algorithm [89] to evolve a colony of agents, supporting in turn social cognition.

The non-deterministic nature of these two learning modes enables the agents to ex-

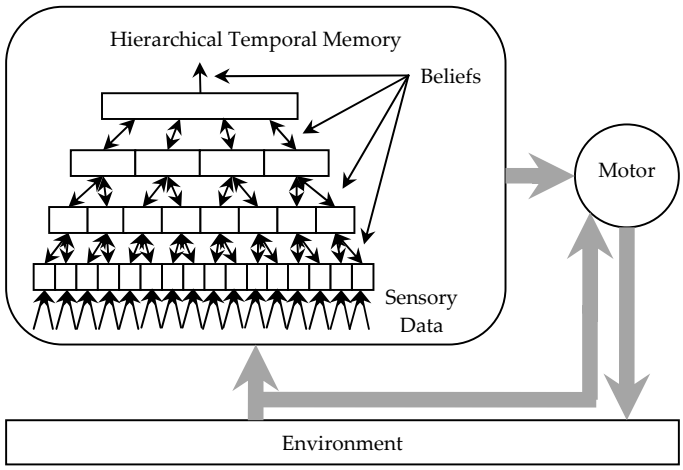


Figure 2.10: The NuPIC cognitive architecture (adapted from [102])

plore various actions and learn which are the most useful. Another trait of TSCA is its simple, demand-driven design, making it scalable to large teams comprising hundreds of agents. This also entails the prospect for hardware realizations to verify concepts developed in simulations. TSCA has been applied in resource gathering tasks, involving both simulation and hardware robots [1, 2]. Nevertheless, TSCA is still lacking support for high-level cognitive functions, such as deliberative reasoning and executive control. Its stochastic learning also makes its behaviors variable and non-reproducible, in contrast to the deterministic nature of the human competitive, altruistic or cooperative behaviors.

2.3.1.4 NuPIC

The *Numenta Platform for Intelligent Computing* (NuPIC) is an emergent architecture modeled on the putative algorithm that the neocortex in the brain uses [102]. It is built upon the *Hierarchical Temporal Memory* (HTM) technology, a variant of Bayesian belief network with extended functions for handling time, self-training, and discovery of causes (denoting objects in the world). It comprises a hierarchy of nodes, each observing the spatiotemporal pattern of its input and assigning causes to this pattern. In this framework, specific connectivity between different layers leads to growing and invariant object representation. Learning and inference in NuPIC are done using belief propagation (BP), an associative technique commonly employed in Bayesian networks that allows all nodes in the system to rapidly settle on a mutually consistent set of cause beliefs.

The foremost feature of NuPIC is its ability to automatically build internal represen-

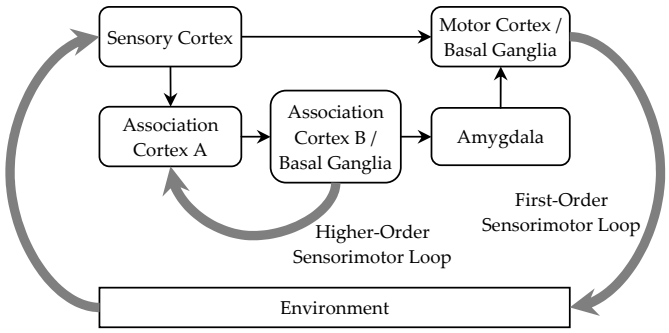


Figure 2.11: The GWCA cognitive architecture (adapted from [233])

tations of the causes in the world, and infer from novel sensory inputs what known causes are likely to occur. It is also unique in stressing the temporal aspect of perception, implementing memory for sequences of patterns that facilitate anticipation and planning. Each level in HTM is trained separately to memorize spatiotemporal objects, and is able to recognize similar objects in either a bottom-up or top-down way. Potential applications include vision systems, data mining, robotics, and fault detection/analysis. However, NuPIC is still lacking an automatic structural learning mechanism to formulate the best hierarchy and architecture/topology for a given task. Many aspects of cognition are not yet supported, e.g. episodic memory, executive control, emotion, etc. Yet another issue is the need for a large amount of data to build an accurate model.

2.3.1.5 GWCA

The *Global Workspace Cognitive Architecture* (GWCA) is an emergent architecture that operates based on internal simulation of interactions with the world [234, 233]. It has a first-order and a higher-order sensorimotor loop that are closed externally via the world and internally via associative memories, respectively, as shown in Figure 2.11. The former comprises a sensory cortex (SC) and a basal ganglia (BG) module that jointly act as a reactive action selection system, while the latter two association cortex (AC) modules that run offline simulation of the agent behaviors. The second-order loop modulates action selection in BG via an amygdala (AM) module. All these modules are realized using the *Generalized Random Access Memory* (G-RAM) [6], that performs single-shot training and whose update functions can be updated quickly. GWCA incorporates reinforcement and competitive learning, carried out in the AM and BG modules respectively. Learning

Table 2.3: Comparisons of the five representative emergent architectures

Aspect	Category	IBCA	Cortronics	TSCA	NuPIC	GWCA
Memory	Globalized memory	DE,PR	-	PR	DE,PR	PR
	Localized memory	DE,PR	DE,PR	-	-	-
Learning	Associative learning	BP	-	RE	BBP	RE
	Competitive learning	HB	WTA	EV	-	WTA
Capability	Perception	Yes	Yes	Yes	Yes	Yes
	Planning/problem solving	Yes	Yes	-	Yes	Yes
	Reactivity	Yes	Yes	Yes	Yes	Yes
	Executive regulation	Yes	-	-	-	Yes
	Emotion	Yes	-	-	-	Yes
	Parallel processing	Yes	Yes	Yes	Yes	Yes
	Hierarchical processing	Yes	Yes	-	Yes	-
	Language comprehension	-	Yes	-	-	-
Application	Cognitive tasks and games	[198, 200]			[102]	
	Conversation and tutoring		[104, 105]			
	Process control and robotics			[1, 2]		[234, 233]

DE = declarative, PR = procedural, WTA = winner takes all, HB = Hebbian, EV = evolutionary
BP = back-propagation (or its variants), RE = reinforcement, BBP = Bayes belief propagation

also occurs in the second-order loop, where AM and BG play similar roles as before.

The existence of the second-order loop enables GWCA to anticipate and plan for future behaviors by exercising its imagination (internal sensory-motor simulation). The GWCA topographic memory is also particularly suitable for spatial cognition, often unaccounted in symbolic methods. In its overall conception, GWCA appeals strongly to the aspects of human imagination, emotion and consciousness. These capacities have been validated experimentally in the Webot simulation environment [233]. As a planning and anticipatory system, however, GWCA lacks compositional structure and systematicity, particularly about chaining a series of associations. In contrast to typical planning systems that search in the space of all plausible plans, GWCA ignores large portions of the search space, and so only supports a very crude type of backtracking. Its large number of neurons or connections also raises the issues of scalability and practicality.

2.3.2 System Comparisons

Table 2.3 presents a list of feature comparisons summarizing the five architectures studied in section 2.3.1. As with section 2.2.2, here comparisons are made based on four aspects: memory organization, learning methodology, system capabilities, and key applications.

Table 2.4: Other prominent examples of emergent cognitive architecture

Architecture	Aspect	Description
NOMAD [136]	Design focus	Brain-based agent for robot design and practical tasks
	Memory	Globalized (neural network), localized (neural network)
	Learning	Competitive (Hebbian), associative (reinforcement)
Ersatz [8]	Design focus	Brain-like computing system for practical cognitive tasks
	Memory	Localized (brain-state-in-a-box network)
	Learning	Competitive (Hebbian)
CBM [48]	Design focus	Large-scale artificial brain model for behavior control
	Memory	Globalized (cellular automata)
	Learning	Competitive (evolutionary)
SRCA [135]	Design focus	Self-referential agent for control of learning and recall
	Memory	Globalized (neural network), localized (neural network)
	Learning	Associative (reinforcement)
Blue Brain [158]	Design focus	Large-scale simulator of the cortical circuits in the brain
	Memory	Globalized (neural network), localized (neural network)
	Learning	Not specified
SASE [285]	Design focus	Developmental robot agent for real-time interactions
	Memory	Globalized (neural network + subsumption)
	Learning	Associative (communicative + supervised + reinforcement)
TOSCA [272]	Design focus	Comprehensive brain-based agent for modeling human mind
	Memory	Globalized (neural network), localized (neural network)
	Learning	Associative (reinforcement)

2.3.3 Other Architectures

An overview of other similar or related emergent architectures proposed in the literature is given in Table 2.4, with emphasis on their design focus, memory organizations, and learning procedures. Equipped with distributed learning and memory mechanisms, these architectures are able to produce interesting emergent properties useful to capture and reason from low-level information. However, it is not clear at the moment how they can be extended to model higher-level intelligence matching that of symbolic architectures. The obscure and fine-grained knowledge representation used by most emergent architectures also generally renders their computations intractable and scaling up rather difficult. Accordingly, to obtain the best of the symbolic and emergent worlds, research on hybrid architectures is now being actively pursued, as shall be elaborated in section 2.4.

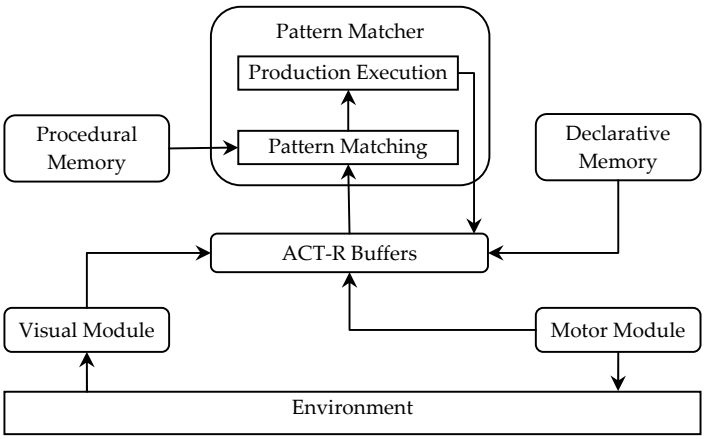


Figure 2.12: The ACT-R cognitive architecture (adapted from [13])

2.4 Hybrid Architectures

2.4.1 Representative Examples

2.4.1.1 ACT-R

The *Adaptive Components of Thought-Rational* (ACT-R) is a hybrid framework for building models capable of a full range of tasks and for explaining the mechanisms of human cognition [9, 14, 13]. It includes visual and motor modules, declarative memory (DM) and procedural memory (PM), a set of buffers, and a pattern matcher, as in Figure 2.12. The visual and motor modules serve as an interface between ACT-R and the world, while the buffers provide access to the ACT-R modules (except for PM). The DM and PM adopt a symbolic-connectionist representation, where the symbolic and sub-symbolic levels comprise productions (in PM) or chunks (in DM), and a set of numeric parameters that measure the usefulness of a chunk or production, respectively. Finally, the pattern matcher serves to find a production matching the current state of the buffers and then execute it. ACT-R utilizes a top-down learning approach, whereby chunks or productions are first created to describe the result of a complex operation so that it can be reused next time. The sub-symbolic parameters are then adjusted using Bayesian formulae to make more available chunks or productions that are frequently used.

One key feature of ACT-R is to provide a collection of quantitative measures that conform to those probed by human subjects and psychological theories. Another trait is its sub-symbolic learning, allowing declarative or procedural knowledge parameters to be

tuned specifically for a given task. ACT-R has been employed in various domains, e.g. human-computer interaction, tutoring system, psychological modeling, and computer-generated force [14, 13]. Nevertheless, perhaps the most subtle problem in ACT-R is the complexity of building a model and deploying it to a given application, which may hinder designers from using the system. ACT-R also exhibits two levels of serial bottlenecks in its processing cycle i.e., only one chunk can be retrieved from any ACT-R buffer at one time, and only one production can be fired in one cycle. As such, ACT-R is not able to cope with multiple goals simultaneously, yielding degraded speed and generalization.

2.4.1.2 CLARION

The *Connectionist Learning Adaptive Rule Induction ON-line* (CLARION) is a hybrid architecture that captures the distinction and interactions between explicit (symbolic) and implicit (sub-symbolic) processes [250, 254]. CLARION has an action-centered subsystem (ACS) for regulating selection of actions, a non-action-centered subsystem (NCS) for maintaining general factual knowledge, a motivational subsystem (MS) for providing motivation/impetus signals, and a metacognitive subsystem (MCS) for monitoring/directing the other three modules, as per Figure 2.13. Each adopts a localist-distributed representation, where the localized and distributed portions encode explicit and implicit knowledge respectively. Implicit knowledge is first acquired via reinforcement (e.g. Q-learning) or supervised (e.g. back-propagation) learning, and then used to derive explicit knowledge via a bottom-up procedure called rule extraction refinement. Conversely, top-down learning is done by precoding rules at the top level and letting the bottom-level accumulate knowledge from observing actions guided by these rules, so that operations that initially rely on the top level gradually become dependent on the bottom level.

CLARION is capable of learning autonomously and continuously from ongoing experiences, regardless of whether a priori knowledge is supplied. Furthermore, it exhibits the intriguing features of situated actions/reactions and cognitive deliberation, and can handle complex situations not acquiescent to simple logical rules. Its principal accounts for emotional/motivational and meta-cognitive functions also enable one CLARION agent to interact with another and with the environment effectively. However, the current CLAR-

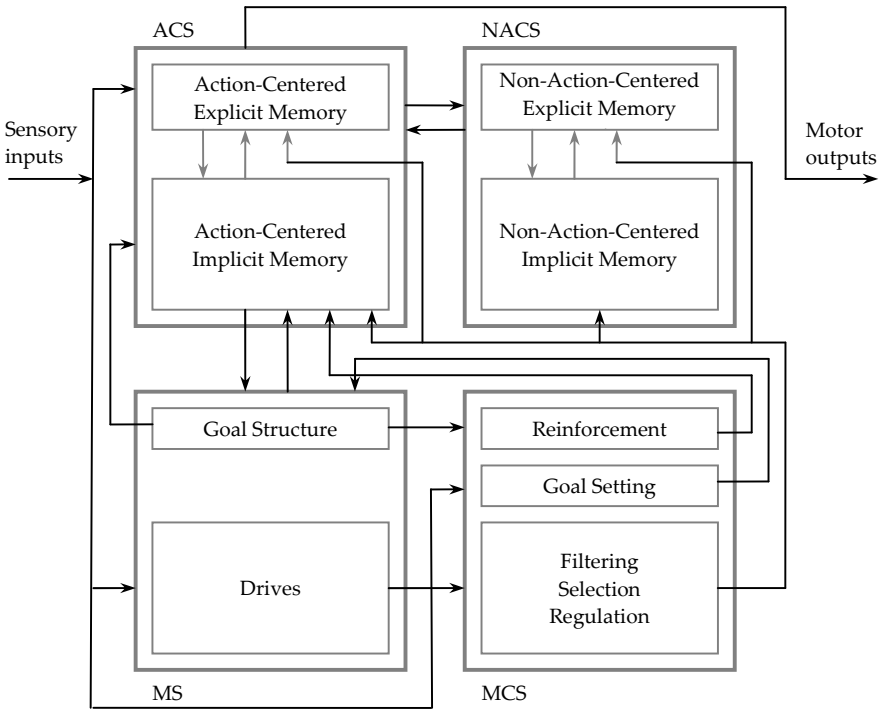


Figure 2.13: The CLARION cognitive architecture (adapted from [254])

ION is lacking a structural learning ability to automatically create or prune nodes and network topology, as well as an episodic learning capacity to track specific events. While CLARION can explain many psychological phenomena, its design is not established upon the neuronal aspects of the brain, thus lacking biological realism. Its capacities have been shown in numerous, but rather small, psychological and navigation tasks [250, 254], and it remains to be seen how the architecture can scale well to larger problem domains.

2.4.1.3 LIDA

The *Learning Intelligent Distribution Agent* (LIDA) is a conceptual and computational framework for "conscious" agent that implements some ideas in the global workspace (GW) theory [74, 73], similar to GWCA. It adopts a symbolic-connectionist organization, with symbols grounded in the physical world in the sense of Brooks [33]. LIDA has distinct modules for attention, action selection, sensory and motor processing, declarative, procedural and episodic memories, as well as working memory (workspaces), as illustrated in Figure 2.14. Much of this modularity, however, is just an illusion emerging from low-level components called codelets i.e., small pieces of program that specializes in a basic task and realizes the unconscious processors in the GW theory. LIDA includes

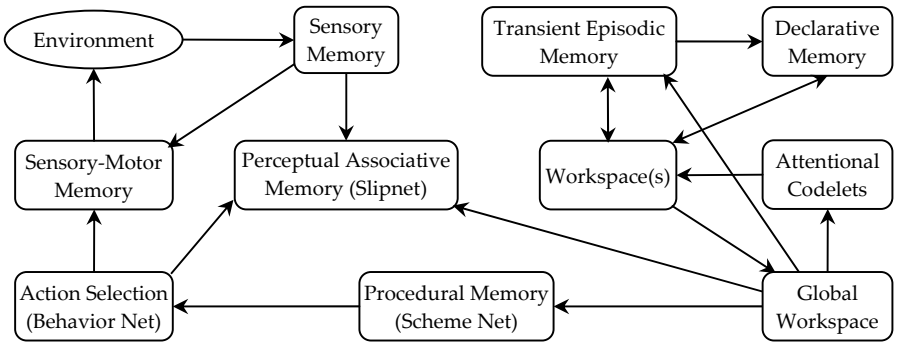


Figure 2.14: The LIDA cognitive architecture (adapted from [73])

three types of top-down learning: a perceptual learning to capture new objects, categories, relations, etc, an episodic learning to memorize specific events, and a procedural learning to acquire optimal actions or action sequences to accomplish new tasks.

A chief virtue of LIDA is its system modularity, conforming to the need for epitomizing different mechanisms of the brain systems and for primitives capable of robust autonomy. LIDA also supports a broad range of hypotheses about human/animal cognition, e.g. the percept-recognize-act cycle, perceptual and episodic memories, memory consolidation, functional consciousness, and voluntary or automatic memory retrievals. Additionally, it accounts for metacognitive capabilities, e.g. executive regulation and emotion, useful to evaluate and guide its cognitive operations. Being a "cognitive theory of everything", however, the breadth of the LIDA framework poses some questions about its usability in practical tasks. Many of the hypotheses offered by the framework, especially with regard to its biological plausibility, are not testable at the moment due to the lack of evaluation methods (e.g. brain imaging) with suitable scope and resolution. It is also not clear how the broad framework can cope with large-scale task domains.

2.4.1.4 Dual

Dual is a hybrid multi-agent architecture supporting dynamic emergent computation, with a unified description of mental representation, memory structures, and processing mechanisms carried out by interacting micro-agents [133, 134]. These agents interact forming larger complexes, coalitions and formations as per Figure 2.15, some of which may be reified. Such models may be evaluated at different levels of granularity: the micro level of micro-agents, the meso level of emergent and dynamic coalitions, and the macro

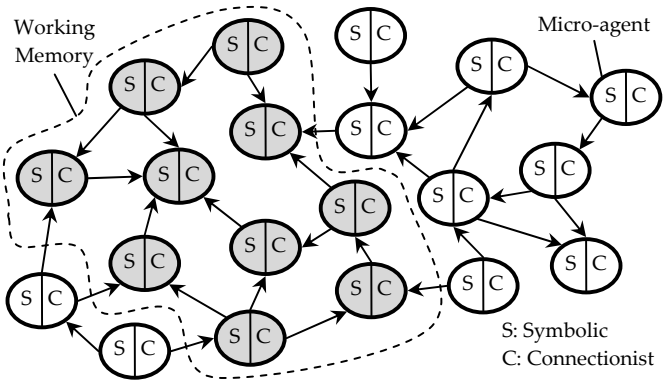


Figure 2.15: The Dual cognitive architecture (adapted from [133])

level of the whole system and models, where psychological interpretations may be used to describe model properties. Micro-frames are used for symbolic representation of facts, while relevance/activation level of these facts in a specific context is captured by network connections with spreading activation that changes node accessibility. Links between micro-agents are based on their frame slots and weights controls the influence of one agent on another's activity, which are learned using a constraint satisfaction algorithm.

The dynamic emergent computation of Dual allows it to efficiently operate in real-time environment, while providing flexibility to handle various situations. Moreover, Dual offers a context-sensitivity feature, where it employs different methods to solve the same task and then takes one that is more appropriate or efficient. Dual has been used in the *Associative Memory-Based Reasoning* (AMBR) project that advocates a set of ideas about human reasoning in general and analogy-making in particular, including models of episodic memory, human judgment, and perception [134, 185]. As its operations often involve large number of agents, however, it is not clear how well DUAL can scale up to real problems requiring complex reasoning. Also, its current design does not yet account for higher-level cognitive functions, such as executive regulation, attention focus, emotion.

2.4.1.5 Polyscheme

The *Polyscheme* architecture integrates multiple representation, reasoning and inference methods in problem solving [37, 38]. Polyscheme does not adhere to a specific memory representation, but its design can generally be regarded as that of the symbolic-connectionist approach. It comprises a set of specialists, each modeling a specific aspect

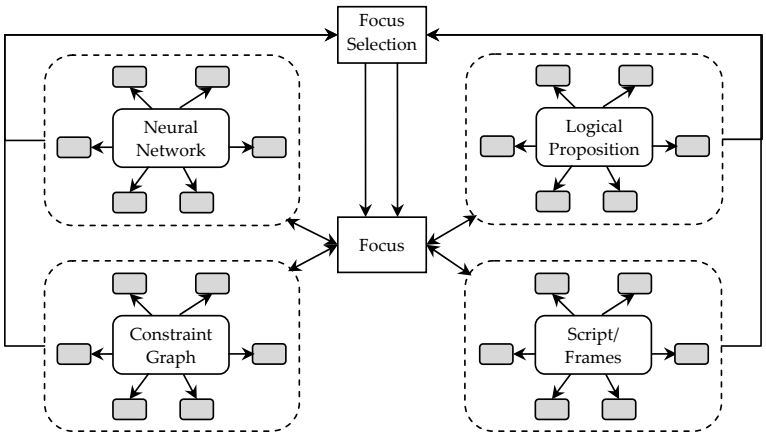


Figure 2.16: The Polyscheme cognitive architecture (adapted from [38])

of the world via different inference and representation (e.g. scripts, frames, logical propositions, neural networks, constraint graphs) methods interacting with or learning from other specialists, as in Figure 2.16. A reflective specialist is also included to guide the attention of the system, providing various focus schemes that implement inferences. High-order reasoning is guided by higher-level policies for shifting the specialists' attention focus. Many problem solving algorithms use forward inference, subgoaling, grounding, representing alternate worlds and identity matching as basic operations. Such operations are handled by different specialists that focus on the same aspect of the world, and may integrate also lower-level perceptual and motor processes.

As such, Polyscheme supports both abstract reasoning and common sense physical reasoning in robots. It offers also a meta-level framework combining different representation and inference methods to problem solving, which is certainly an important step towards AGI and common-sense reasoning. Polyscheme has been used in several domains, such as query answering, syntactic parsing, and cognitive robotics [37, 39, 38]. However, one issue with the current Polyscheme design is the fairly abstract, coarse-grained specifications for the learning and communication of the specialists. Another concern is the difficulties in devising a Polyscheme model and setting it up for a specific task, raising thus some questions about its practical usability. The current Polyscheme design is also missing accounts for several established aspects of the human cognition, e.g. emotion, episodic memory, hierarchical processing, etc., and its scaling properties to handle complex, large tasks have not been yet demonstrated so far.

Table 2.5: Comparisons of the five representative hybrid architectures

Aspect	Category	ACT-R	CLARION	LIDA	Dual	Polyscheme
Memory	Symbolic-connectionist	-	DE,PR	-	-	-
	Localist-distributed	DE,PR	-	DE,PR	DE,PR	DE,PR
Learning	Top-down learning	BPE	TRT	PE,EP,PR	-	*
	Bottom-up learning	-	RER	-	CSM	*
Capability	Perception	Yes	Yes	Yes	Yes	Yes
	Planning/problem solving	Yes	Yes	Yes	Yes	Yes
	Reactivity	Yes	Yes	Yes	Yes	Yes
	Executive regulation	Yes	Yes	Yes	-	Yes
	Emotion	Yes	Yes	Yes	-	-
	Parallel processing	Yes	Yes	Yes	Yes	Yes
	Hierarchical processing	Yes	Yes	-	-	Yes
	Language comprehension	Yes	-	Yes	-	Yes
Application	Cognitive tasks and games	[9, 14, 13, 11]	[253, 252, 255]	[77, 69]	[134, 185, 209]	
	Conversation and tutoring	[78, 221, 12]	[184]		[179]	[37, 38]
	Process control and robotics	[262, 261, 125]	[255]	[74, 72, 73]	[210, 129]	[39]

DE = declarative, PR = procedural, BPE = Bayes parameter estimation, TRT = top-down rule translation
RER = rule extraction refinement, CSM = constraint satisfaction mapping, * = representation-dependent

Table 2.6: Other prominent examples of hybrid cognitive architecture

Architecture	Aspect	Description
Shruti [237]	Design focus	Biologically-inspired model of human reflexive inference
	Memory	Localist-distributed
	Learning	Bottom-up (back-propagation)
4CAPS [116]	Design focus	Parallel connectionist-like production rule system
	Memory	Symbolic-connectionist
	Learning	Bottom-up (ensemble)
SCAN [286]	Design focus	Scanning comprehension of natural language phrases
	Memory	Localist-distributed
	Learning	Bottom-up (recurrent propagation)
ISACA [227]	Design focus	Self-aware agent modeling human cognitive growth
	Memory	Symbolic-connectionist
	Learning	Bottom-up (Hebbian law + reinforcement)
BRAINN [28]	Design focus	Integration of rule-based and similarity-based reasoning
	Memory	Symbolic-connectionist
	Learning	Bottom-up (Hebbian law)
4D/RCS [5]	Design focus	Reference model architecture for multi-agent systems
	Memory	Symbolic-connectionist
	Learning	Bottom-up (localized error-correction)
Novamente [153]	Design focus	Integrative architecture modeling general intelligence
	Memory	Symbolic-connectionist
	Learning	Top-down (evolutionary programming), bottom-up

2.4.2 System Comparisons

A comparative list of the memory, learning, capabilities, and applications of the hybrid architecture examples described in section 2.4.1 is presented in Table 2.5.

2.4.3 Other Architectures

Table 2.6 lists other similar or related hybrid architectures proposed in the literature, and summarizes their design focuses, memory organizations, and learning methodologies. These and the previous five architectures have been evaluated quite successfully in various task domains. Nevertheless, several salient features remain to be addressed in order to achieve a more powerful, human-like general intelligence. The three most important features are the ability to consolidate the knowledge acquired, to handle tasks of increasing complexities, and to self-monitor and -improve over time. Further discussions on these aspects shall be provided in section 2.5.

2.5 Discussion and Direction

The previous sections show that there is no lack of good ideas and models of cognition. Many excellent projects have been formulated, some developed for many years and some are just starting. Yet, no cognitive architecture appears to have been used in large-scale real-world applications. Grand challenges and detailed roadmap that lead to human competency level should be formulated to guide the research. One plausible direction is to formulate common general task repositories, each of which demands a variety of cognitive functionalities, based on which different cognitive architectures can be evaluated. For instance, integrative models of human performance are of great interest in the defense and aerospace industries, and a recent project on the Agent-Based Modeling and Behavior Representation (AMBR) resulted in quantitative data comparing the performance of humans and cognitive architectures in a simplified air traffic environment [85]. Some efforts have been expended on the evaluation of software agents, such as those presented in the AAAI Workshop on "Evaluating Architectures for Intelligence" [119]. Ideas ranged from using in-city driving environment as a testbed for cognitive architectures, to measuring incrementality and adaptivity components of general intelligent behaviors.

Perhaps a measure of "cognitive age" can be established, with a set of tasks that children at a given age can solve. Problems should be divided into several groups: vision and auditory perception, language understanding, common-sense reasoning, abstract reason-

ing, probing general knowledge about the world, learning, problem solving, imagination, creativity. Solving all problems from a given group that children at some age can solve will qualify a cognitive system to the next grade in this group. It is expected that some systems will show advanced age in some areas, and less in others. For example, solving problems requiring vision may demand adding specialized computer vision modules, while mathematical reasoning in many reasoning systems may be more advanced than in children. Experts on human intelligence largely agree to the original Gardner's proposal [81] that seven kinds of intelligence should be distinguished: logical-mathematical, linguistic, spatial, musical, bodily-kinesthetic, interpersonal and intrapersonal intelligence, perhaps extended by emotional intelligence and few others.

Such analysis should help understand what type of intelligence is expected from embodied cognitive robots and what limitations the symbolic approaches have. Brooks made a good point that elephants do not play chess [32], and expressed hope that a robot with integrated vision, hearing and dextrous manipulation controlled by large-scale parallel computer will learn to 'think' by building on its bodily experiences to accomplish progressively more abstract tasks [33]. His Cog project [33] based on grounding the meaning of concepts in deep embodiment has many followers, though after many years it stays only at the level of reactive intelligence and there is no good idea to extend it to higher cognitive levels. While behavioral intelligence in robots may be difficult to achieve without embodiment, experiences with this approach have not been very encouraging so far. Elephants are intelligent, but cannot learn law or be personal assistants.

The survey presented above shows several trends that will most likely dominate the cognitive architecture research. First, many hybrid architectures are already out there and biological inspirations are becoming increasingly important, leading to the domination of *Biologically-Inspired Cognitive Architecture* (BICA). Even the hardcore symbolic architecture proponents now base further extensions of their architectures on inspirations from the brain, focusing largely on the roles of cortex and limbic systems. Second, there may be many BICA architectures, but several key features need to be preserved. Different types of memory and learning, such as episodic vs. semantic, declarative vs. procedu-

ral, etc., are certainly important, as already stressed by several cognitive architectures reviewed in this chapter. Yet another crucial facet is the formulation of systematic and plausible procedures by which different memory modules can interact in a complementary manner, and thus produce the desired collective behaviors.

The previous review also prompts several important features that are needed to address the shortcomings of the current approaches, and to develop a more powerful yet practical general intelligence. The first feature is about *knowledge consolidation*, i.e., the ability to automatically acquire, compress and crystallize knowledge through interactions with the environment. Such capacity is especially useful for gracefully dealing with novel situations that can cause radical interference with the existing knowledge base and, by extension, for deriving simple and efficient model that can cope well with large amount of information and uncertainty. The realization of such feature involves two important requirements: mechanisms for dynamically constructing and reducing knowledge representation, and those for reinstating or refreshing the previously acquired knowledge in order to minimize forgetting. Several emergent and hybrid architectures, such as IBCA and LIDA, have incorporated consolidation mechanisms in their framework, but so far they do not fully address the two aforementioned requirements. A more complete account of the consolidation paradigm shall be provided in chapters 4 and 5.

Another related feature desirable in cognitive architectures supporting general intelligence is the *scalability* to cope with tasks and situations of varied complexity levels. While some symbolic, emergent and hybrid architectures (e.g. ICARUS, Cortronics, GWCA) already address this aspect to some extent, it is never treated as explicit or primary goal. The notion of scalability can be related to that of efficiency in formal analysis of algorithms, where the space and time efficiency of a system is a function of task complexity, operation length, environmental uncertainty, and other factors. A common case of scalability issue in the contemporary approaches arises with cognitive architectures that learn over time. As new information comes in and is added to their memories, the knowledge retrieval and acquisition processes in these systems become slower [173]. In this respect, the knowledge consolidation (particularly the construction and reduction) mechanisms

would provide an effective way to improve the scalability of the existing architectures. This point will be further elaborated in chapters 4 and 5, as well.

Last but not least, it is expected for any cognitive architecture to be able to know what it is doing and what it is supposed to do. Such capability points to the notion of *metacognitive function*, which involve higher-level aspects of cognition such as emotion, self-awareness and self-improvement. These aspects have been stressed by several architectures such as IBCA, GWCA, CLARION and LIDA, but have only been addressed partially and there is no integrated account for them at the moment. To this end, there is a need to develop generic set of methodologies that would allow a cognitive architecture to autonomously bootstrap its behavioral performances over time, independent of the type or configuration of its constituent modules. One plausible means to achieve this is to implement a generic *meta-learning algorithm* that can automatically select a proper method or module from existing repertoire to solve a given task, or create a new one if the stereotyped behaviors are inadequate [57, 56]. Such approach would allow optimizing different architectural modules, based on either external stimuli from the environment or feedbacks from own internal processes. In extension, this would provide a mechanism for integrating and/or switching between different types of computational methods, paving the way toward a supervisory or meta-optimization framework [215, 56].

2.6 Summary

This chapter presented a survey of prominent cognitive architectures modeling human intelligence. A simple taxonomy of symbolic, emergent, and hybrid systems has been proposed to evaluate various cognitive architectures. Representative examples of each class were then given, highlighting their design properties, capabilities, as well as strengths and limitations. Several guidelines to address the open issues in the current approaches have also been outlined, with emphasis on the aspects of knowledge consolidation, scalability and metacognition. Our own approach to realize some of these ideas consists of building an integrated neuro-cognitive architecture, which is elaborated in the next chapter.

Chapter 3

Integrated Neuro-Cognitive Architecture

3.1 Brain Inspirations

A novel brain-inspired hybrid framework, dubbed *Integrated Neuro-Cognitive Architecture* (INCA), is elaborated in this chapter which focuses on the aspects of knowledge consolidation, scalability and metacognition, as put forward in Chapter 2. Before going into its details, it is worthwhile to look at the key psychological and neuroscientific principles of learning and memory in the brain that serve as the inspirational basis for developing the architecture. A popular means used in neuroscience to identify functional localization in the brain is by showing a *double dissociation* between behaviors [121, 62]. A dissociation exists when lesion in one brain area P is shown to impair a function X but not Y. In turn, a double dissociation occurs when one can further show that lesion in a different region Q affects function Y but not X. The wide adoption of such methodology has resulted in a taxonomical framework of learning and memory, which is illustrated in Figure 3.1 and detailed in sections 3.1.1 and 3.1.2. Further discussion on the mechanisms by which learning and memory systems interact is given in section 3.1.3.

3.1.1 Memory

Broadly speaking, there are two types of memory system: *short-term memory* (STM) and *long-term memory* (LTM) [64]. STM exhibits a limited capacity, and so degrades quickly if its content is not maintained via rehearsal or transferred into LTM. Conversely, LTM has a theoretically unlimited capacity and is fairly stable (though still subject to

forgetting). Sections 3.1.1.1 and 3.1.1.2 discuss the aspects of LTM and STM respectively.

3.1.1.1 Long-Term Memory

An inherent confusion in the study of human memory systems is the cornucopia of the terms that the researchers use to describe components of the LTM. A widely established division within LTM is between *declarative* and *procedural memories*, as advocated by Squire and Cohen [247]. In essence, declarative memory refers to a storage for verbalizable information associated with facts and events. On the other hand, procedural memory is concerned with working skills or procedures that cannot typically be expressed verbally. Yet another established class of LTM is *emotional memory*, which governs expression and learning of emotional responses (drives or intentions) to stimuli [62].

Schacter [229] introduced yet another classification dimension of LTM, distinguishing between *explicit* and *implicit memories*. Explicit memory denotes a memory store whose contents can be recalled by conscious efforts. This type of memory is flexible and typically involves association of multiple bits and pieces of information. In this, conscious awareness is usually implied, as is intention to remember. Conversely, implicit memory can be recalled unconsciously and is often involved in learning reflective perceptual or motor skills, such as priming, skill learning, and conditioning. Unlike explicit memory, implicit memory is more rigid and tightly connected to the original stimulus conditions under which learning occurred. Due to their similarities, explicit and implicit memories are often associated with declarative and procedural memories, respectively [121].

Further distinction can be made within declarative memory between *episodic* and *semantic memories*, as proposed by Endel Tulving [274]. The former refers to a repository for specific episodes or events (usually autobiographical) that are coupled with some spatial and temporal references. Semantic memory, on the other hand, retains general facts about the world that are independent of spatial or time references. There is much debate about the existence of separate episodic and semantic systems within the brain. Although researchers first described them as physically disparate, there is now a consensus that the two overlap or depend on one another to some degree [121, 200].

Declarative Memory. Various neurological and psychological studies have shown that many forms of declarative memory depend on the medial temporal lobe area in the brain, particularly the *hippocampal region* and its surrounding cortical circuits (i.e., entorhinal, perirhinal and parahippocampal cortices) [62, 121]. A pioneering study of a neurological patient named H.M. have exemplified what happens in the absence of the hippocampus and its related structures [171, 232]. It was shown that H.M. almost completely failed to learn new materials, though his remote autobiographical and semantic memories were intact. This leads to the idea that the hippocampal region plays a role as mediator for *consolidation* of transient memories into a permanent, long-term store [247, 200]. Subsequent studies on amnesic and healthy patients [62, 248] have also shown that the hippocampal region is crucial for the formation of both episodic and semantic memories.

Another key structure that serves as long-term repository for declarative knowledge is the *association area of the neocortex* [121, 62]. It is also shown that lesions in different parts of association cortex give rise to specific defects in either semantic or episodic memory. In particular, semantic memory lesions can be related to the association *posterior cortex*. Several studies have indeed shown that our experience of knowledge as seamless, orderly, and cross-referenced database is a product of integration of multiple, possibly overlapping representations at distributed posterior cortical sites [121]. On the other hand, lesions in association cortical areas may also interfere with our capacity to recall episodic memories. The areas that seem specialized for long-term storage of episodic knowledge are those of the *frontal cortex*. These areas work together with other areas of the neocortex to allow recollection of when and where a past event occurred [121].

Procedural Memory. A vast array of coordinated behaviors that we execute everyday, such as playing a sport or riding a bicycle, are the product of procedural memory. This form of memory has been believed to rely on the *cerebellum* and *basal ganglia* in the brain [121, 62]. The cerebellum has been perceived as a regulator of motor movements. Studies on the vestibulo-ocular reflex and eyeblink conditioning have shown that the cerebellum encodes the timing and amplitude of learned movements [30]. It forms an internal model of a controller or control subject, and performs an error-correction learning to gradually

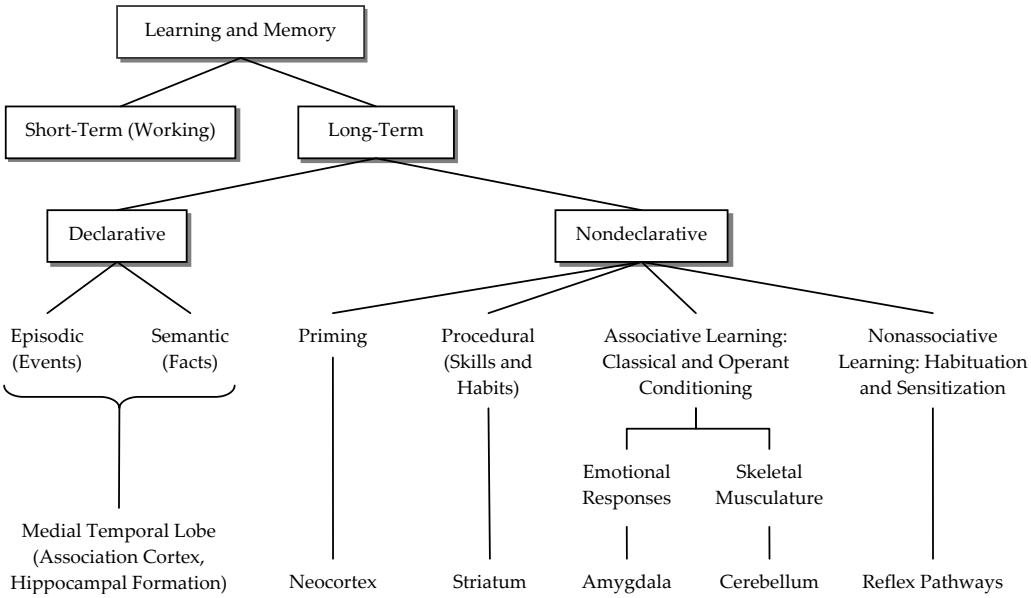


Figure 3.1: Human learning and memory systems (adapted from [121])

reduce the difference between its outputs and the external (desired) control feedbacks when similar input stimuli are supplied [4]. Modern studies have also shown that the cerebellum has a broader role in several key cognitive functions, including attention and processing of language, music, and other sensory temporal stimuli [218].

Another major brain system for procedural memory is the *basal ganglia*, and in particular the *striatum*. The striatum plays a key role in habit and skill learning that requires resolution of competition among multiple input or response options, especially in tasks involving the learning of response sequences [93, 62]. This role becomes more crucial in higher-level cognitive tasks where temporal sequencing of information is involved, as in the case of reinforcement learning paradigms. Whether the functional roles of the basal ganglia and cerebellum in temporal sequencing are identical, or really quite different, is unclear. It is evident that these areas have different structures, and some studies have suggested dissociations in the roles of these systems [62]. However, it is also clear that these structures all modify their circuitries in the service of procedural memory.

Emotional Memory. Perhaps the best example of emotional memory is illustrated by the classical fear conditioning studies by Joseph LeDoux and Michael Davis [145, 47]. The studies focused on learning of fearful responses to a simple auditory stimulus, involving a complex set of cortical and subcortical areas in widespread areas of the brain, and in

particular the *amygdala*. Chiefly, it was found that the amygdala intervenes between the regions related to the somatic expression of emotion and the cortical areas related to fear; it thus mediates both inborn and acquired emotional responses. Damage to the amygdala would indeed cause selective impairments in emotional perception and appreciation, as well as emotional expression in humans and animals [62].

The amygdala also appears to be involved in the modulation of long-term memory consolidation, in which emotional arousal following any learning event influences the strength of the subsequent memory for that particular event [70, 62]. Recent findings on animal training in a variety of classical conditioning tasks have found that drugs injected to the amygdala after training affect the animals' subsequent retention of the task [70]. That is, if a drug that activates the amygdala is injected, the animals had better memory for the task. Conversely, supplying a drug that inactivates the amygdala yielded lesioned memory. Nevertheless, despite the importance of the amygdala in modulating memory consolidation, learning can occur without it, though such learning seems impaired as in fear conditioning impairments following an amygdalar damage [128].

3.1.1.2 Short-Term (Working) Memory

The relationship between STM and LTM has been debated for many years. Experiments on cognitive tasks have shown that the two measure different capacities of memory [22, 121]. Nevertheless, there is a general consensus that many aspects of LTM depends on STM. In this case, the two can be regarded as components of a single system that are linked in a serial fashion, whereby information entering LTM must inevitably flow through STM. The notion of STM as a component of LTM has led to the idea of *working memory* [22, 23], a distinct system that encompasses some of the capacity limitations of STM, and a dynamic system that affects attention and metacognitive (executive) functions.

In this regard, working memory serves to direct temporary storage of incoming information in various tasks, from reading to math and problem solving. The prominent Baddeley's model [23] conceptualizes working memory as having four components: a *central executive* for supervising and coordinating the components of several slave systems, a *phonological loop* for auditory-verbal processing often associated with the linguistic

areas of the left brain hemisphere, a *visuospatial sketchpad* for manipulating visual and spatial images from both the environment and LTM that relate to the right brain hemisphere, and an *episodic buffer* for a temporary interface between the slave systems (i.e., phonological loop and visuospatial sketchpad) and LTM.

The phonological loop and visuospatial sketchpad have been linked to the lateralized modalities in the brain and to the *frontal cortex*, particularly the *prefrontal area*. Goldman-Rakic's [91] groundbreaking work in primate models of working memory points to the prefrontal cortex as the areas that holds information online while it is processed. In this study, monkeys are tested for their ability to recall the position of food in one of two food wells after a short delay of several seconds. Subsequent brain imaging studies have also supplied confirmation in humans that the prefrontal cortex activates during working memory tasks [114]. This also suggests that the the functioning of the working memory is closely related to attention and metacognitive processes.

3.1.2 Learning

Neuroscientific investigations on the learning aspects of the brain usually involve exposing experimental subjects to controlled sensory experiences. Such studies have identified two major classes of learning: *nonassociative* and *associative learning* [267, 121], as described in sections 3.1.2.1 and 3.1.2.2 respectively.

3.1.2.1 Nonassociative Learning

Nonassociative learning occurs when an animal or a person is exposed repeatedly to a single type of stimulus. Two most commonly known forms of non-associative learning are *habituation* and *sensitization*. Habituation refers to a decrease in response to a benign stimulus that is presented repeatedly, while sensitization is a process by which an intense or noxious stimulus can enhance responsiveness to subsequent stimulus presentation [120, 121]. A sensitizing stimulus may override the effects of habituation, and this process is called *dishabituation*. Sensitization and dishabituation are independent from the relative timing of the intense and weak stimulus, i.e., no association between the two stimuli is required. Both habituation and sensitization are respondent conditioning processes, and

are related to biology and reflexes and sometimes overlapped by operant learning.

Many aspects of habituation and sensitization can be related to those of *unsupervised* (*competitive*) *learning* in neural modeling, which aims at identifying the salient statistical features of particular input patterns by adapting their internal representations [126, 132]. In this framework, a set of neurons compete with one another for the right to respond to (or represent) an input pattern. Sensitization can accordingly be viewed as a process by which a neuron is made to produce a stronger response to a pattern, whereas habituation is a process by which the neuronal response to patterns other than the current pattern is strengthened [126]. Together, the two complementary processes form a powerful adaptive mechanism that enables humans to continuously learn changing patterns.

3.1.2.2 Associative Learning

Associative learning results from the conjunction of two (or more) entities, and consists of two forms: *classical* and *operant conditioning* [121]. The gist of classical conditioning is learning to associate between a *conditioned stimulus* such as a light, tone or tactile stimulus, which normally produces either no overt or a weak response usually unrelated to the response that will eventually be learned (i.e., *conditioned response*), and an *unconditioned stimulus* such as food or an electric shock, which yields a strong, overt and innate response that arises without learning (i.e., *unconditioned response*) [204]. Meanwhile, operant conditioning involves learning to predict the relationship between a response (i.e., behavior), such as button-pressing, and a stimulus that follows (i.e., consequence of the behavior), such as food [268, 242]. Despite the different kinds of association involved in classical and operant conditioning, the rules governing the two are quite similar, suggesting that they may use the same neural mechanisms [121].

The above notion of associative conditioning also has a close relationship to the well-known *reinforcement* and *supervised learning* paradigms. For instance, the links between an influential reinforcement paradigm known as Temporal Difference (TD) learning, and classical conditioning, are close and widely recognized [258]. In this, the computational goal is to ensure that the early-occurring CS becomes a predictor of the later-occurring US after learning (i.e., prediction problem), creating a situation where temporal differences

of a value function need to be evaluated. Reinforcement learning also provides a good model of operant conditioning, though the formal, detailed argument for this is not yet made (the closest so far is perhaps [25]). The supervised learning paradigm, on the other hand, can be viewed as a special case of reinforcement learning, where the (US) feedbacks are explicit/instructive such that no trial-and-error process is involved [24, 130].

3.1.3 Consolidation and Inference

The previous sections have shown the various aspects of learning and memory systems in the brain. But how do these systems interact to produce cognition? Two fundamental interaction mechanisms are considered in this thesis, as also established in neuroscience. The first is *consolidation* mechanism, which refers to the process by which memories are acquired and crystallized. It is well known that some experiences are quickly forgotten, whereas others are retained for a lifetime. It is also evident that various forms of interference or brain lesion can erase memories that were recently acquired, but have less effect on those gained remotely before the interfering event or injury [171, 165, 162]. These suggest that memories are initially labile and eventually become resistant to loss, implying that a consolidation occurs during which memories take on a permanent form.

It is held that the final repository of the consolidated memories is the cortex, and that the hippocampus provides an initial storage to organize or mediate the formation of permanent memories in specific cortical sites [7, 162, 117]. The main substrate for consolidation in the cortex is believed to involve a hippocampal-based rehearsal process during slow-wave sleep (and possibly rapid eye movement sleep and/or quiet wakefulness) of the activity patterns reflecting past active behavioral states [288]. Recent studies have also shown that amygdala plays a crucial role in mediating the modulating influences of drugs and hormones on memory consolidation, since lesion in amygdala would block the effects of these modulators [165]. Another fear conditioning test on rats found that consolidated fear memory, when reactivated, enters a changeable state that requires protein synthesis afresh for new consolidation (or reconsolidation) of the old memory [181].

The second key mechanism for interaction of different learning and memory systems is the *inference* process, which pertains closely to the *perception-action cycle* described in

[79, 80]. The cycle essentially refers to the circular flow of information occurring between an organism and its environment in the course of a sensory-guided sequence of behavior towards a goal. Each action in the sequence causes changes in the environment that are processed in a bottom-up manner via the perceptual hierarchy (including the occipital, temporal and parietal lobes), and lead to the processing of action in a top-down fashion via the executive hierarchy (including the frontal lobe) toward the motor effectors. This action causes new environmental changes, which leads to new sensory input and so on. All stages of processing generate internal feedback on earlier stages, which serves to monitor and modulate incoming signals at every stage. This cybernetic cycle has been expanded in humans to include speech, reasoning, and executive processes, but remains the primary causal mechanism of behavior and neural function [79, 80].

3.2 Related Computational Models

This section presents several prominent computational models that emulate some of the brain aspects previously described and that later provide inspirations for the formulation of the learning memory modules in INCA and their interactions. Note that this section serves to provide introductory information and so the listing of models given here is by no means complete. More in-depth treatment of the models (and their extensions) can be found later in chapters 4 and 5, when discussing the INCA modules in greater detail.

Perhaps the most established brain-inspired computational model is that of the *association* (*posterior*) *cortex*. The Multi-Layer Perceptron (MLP) network, first proposed in [163], is an example of such cortical model that has been widely used in pattern recognition tasks, whereby the hidden layers of the network capture to some extent the non-linear, hierarchical traits of the association cortex. Many extensions have been developed over the years to develop more comprehensive models of the cortex, the most popular approach of which is based on hierarchical probabilistic generative models [104, 102, 49]. Another approach that functionally models the localized learning aspects stemming from the columnar organization of the cortex has been proposed recently in [266, 188].

Yet another popular target of computational modeling is the *cerebellum* circuit, chiefly

due to its regular, homogeneous anatomical structure. An influential and established theoretical account of the cerebellum is the Marr-Albus-Ito model [159, 3], which described how the cerebellum, specifically its climbing fibers, transmits moment-to-moment changes in sensory information for movement control. Moreover, it was shown that the cerebellum participates in motor learning by detecting error in one movement and changing its program for the next movement. Building upon this theory of cerebellar function, a network model named Cerebellar Model Articulation Controller (CMAC) was developed that employs error-correction signals to drive the learning and memory skills [4]. CMAC has been recognized for its localized learning and generalization traits, and as such has widespread use in many applications, particularly adaptive control systems [170].

Several attempts have also been made to model the functional aspects of the *hippocampal* system, which roughly fall into two major groups: models that focus on slow, sequential learning to construct semantic representation over multiple training trials (e.g. [230, 86], and models that focus on rapid storage and retrieval of episodic memories within a single trial (e.g. [180, 236]). Despite this opposition, it is becoming apparent that the two views are partially correct and reflect different aspects of the hippocampal structures [62, 167]. A reconciliation of the two can be made by considering the role of the hippocampal system in encoding sensory information that contribute to our experiences or episodic memories, and then sequentially linking them via their common elements to form flexible semantic memories [62]. A transient learning memory modeling approach that epitomizes this consensus view has been reported in [275, 197].

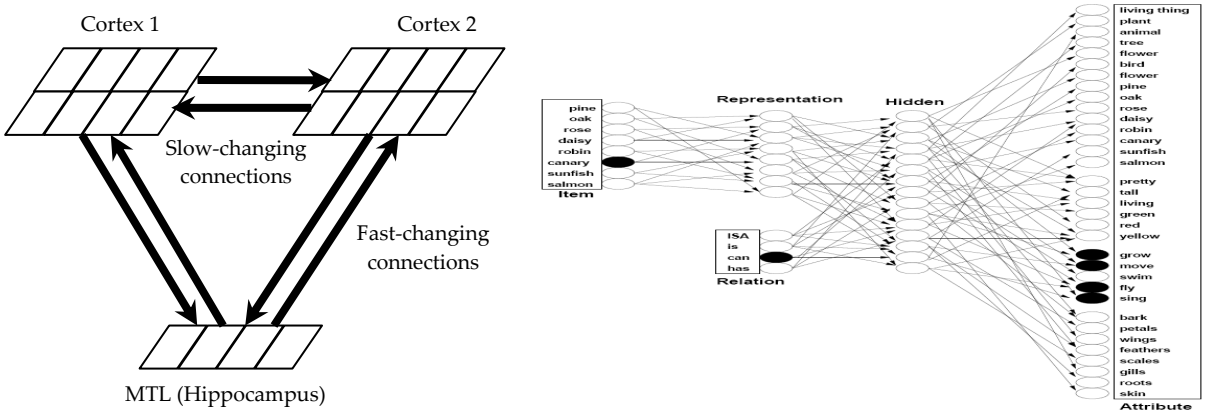
Additionally, some efforts have been made to build computer-based models of emotional aspects, focusing on the *amygdala* structure in the brain. For instance, Grossberg [96] devised models of conditioned emotional states based on the premise that conditioned reinforcement involves pairs of antagonistic neural processes such as fear and relief. The model suggested a mechanism by which neutral events are charged with a (positive or negative) reinforcing value, which depends on the previous activity of the model. Armony and colleagues [19] developed another network model of emotional learning and memory that shares some similarities to Grossberg's models. The model investigates processing

in two parallel sensory transmission pathways to the amygdala from the auditory thalamus and auditory cortex in a fear conditioning situation, suggesting in turn a close link between the amygdala and the attentional system in the midbrain.

In the context of working memory, a number of computational models of the *frontal cortex* and *basal ganglia* have been developed to simulate their functions. A notable model was proposed in [53, 231], which simulates the role of neuromodulators associated with the basal ganglia and cortex (e.g. dopamine, serotonin, noradrenaline, acetylcholine) in mediating global signals that govern distributed learning processes in the brain. The model provides a robust and biologically plausible meta-learning algorithm for fine-tuning the hyperparameters or meta-parameters of any learning system in a dynamic and adaptive manner. Recently, O'Reilly and Frank [199] proposed a more elaborate biological model of frontal cortex and basal ganglia, where the former serves to hold information online for processing, and the latter performs dynamic gating to allow only task-relevant information to be maintained in the frontal cortex and to control the working memory updating therein. This model has demonstrated powerful learning abilities to deal with complex working memory, and by extension metacognitive, tasks.

Meanwhile, there are several computational accounts for the mechanisms of memory consolidation. Alvarez and Squire [7] for the first time developed a simple neural network model that exhibits the distinct operating characteristics of the cortex and hippocampus, as illustrated in Figure 3.2(a), and showed quantitatively how the model can capture the basic features of consolidation. They found that the cortex can store an immense amount of information but its connections change slowly, whereas the hippocampus has a limited storage but can change its weights to capture information rapidly. Despite its intuitiveness, it was acknowledged that the model has some serious limitations, mainly attributed to its spartan simplicity [7]. The model also requires some way of establishing or strengthening the connections between neurons in disparate cortical areas, which would not normally be expected to enjoy substantial reciprocal connections.

A more comprehensive model of consolidation was developed by McClelland and his co-workers [162]. In their model, the cortical representations involve a semantic network



(a) Simple network model capturing the cortico-hippocampal interaction [7]

(b) Semantic network modeling the organization of propositional knowledge about living things [162]

Figure 3.2: Neural network models of memory consolidation

of related items organized in parallel, multidimensional hierarchies, as per Figure 3.2(b). However, once a set of hierarchical organization is established, it is difficult to add new items smoothly; it causes *catastrophic interference* with the previously established items. McClelland's solution to this was to add a new small "hippocampus" network that can rapidly acquire representation of a new item, and then have this network slowly train the large "cortex" network. The latter was also repetitively exposed to the old materials it was built to represent, yielding an *interleaved learning* regimen that blends repetitions of old and new representations. The model thus exhibits the key feature of consolidation; the interleaving process would eventually produce an asymptotic state of the cortical representation, at which point it no longer relies on the hippocampal activations.

One issue with McClelland's model is that, regardless of how slow the cortex learns new patterns fed by the hippocampus, it does not prevent the overwriting of information already in the cortex [75]. In light of this, French [75] and Ans and Rousset [18] independently devised dual-network models comprising two continually interacting modules, one for early-processing, one for long-term storage, with information passed back and forth between the modules by means of *pseudopatterns* [222], which approximate the original patterns the modules were exposed to. This *pseudorehearsal* mechanism helps mitigate catastrophic interference by interleaving the existing cortical patterns with the new patterns being learned. Such principle can also be used to explain the aspects of rapid eye movement sleep, memory consolidation, and anterograde/retrograde amnesia [75, 18].

3.3 INCA Architectural Framework

This section elaborates the INCA framework that models the various cognitive aspects described in sections 3.1 and 3.2. Each module in INCA employs both structural and parameter self-organizing mechanisms utilizing various unsupervised, supervised and reinforcement learning methods. Unsupervised learning is done via clustering, while supervised and reinforcement learning involve mapping in the modules states to target actions and state-action pairs to a utility function, respectively. In turn, these provide support for metacognitive/reflective functions (e.g. attention shifting, affect modulation, executive control) to direct and enhance ongoing processes. The macro-level behaviors of INCA emerge from its module interactions in the form of *consolidation* and *inference cycles*, epitomizing the human knowledge acquisition and retrieval processes respectively.

The INCA framework, illustrated in Fig. 3.3, has several modules: *sensory* and *motor registers*, a *transient memory*, an *affect modulator*, distinct *declarative* and *procedural memories*, and an *executive manager*. The last three modules collectively form the system's long-term memory. The arrows in Fig. 3.3 refer to the direction of information flow between the modules, whereby the unidirectional arrows indicate one-way propagation of information and the bidirectional ones show two-way exchange of information. INCA also includes several algorithmic protocols to support inter-module communications: an *attention protocol*, a *consolidation protocol*, and a *retrieval protocol*. Apart from the two registers which are hand-coded at design time, the remaining modules are self-organizing memories and exhibit functionalities that are learned. Also, while each module maps to a key brain region, the aim is not to model accurate biological details, as it would likely pose a more complex, less-scalable design or adversely impact system performance.

Instead, the INCA modules realize the putative, functional aspects of the brain regions using a *Neuro-Fuzzy System* (NFS) modeling approach [196, 197, 151]. NFS is a hybrid symbolic-connectionist model that combines the learning capability, parallelism, and robustness of neural networks with the human-like, symbolic, and approximate reasoning abilities of fuzzy rule-based systems. That is, NFS offers the mechanism to automatically induce, from raw (numerical) data, decision logic in the form of intuitive fuzzy linguis-

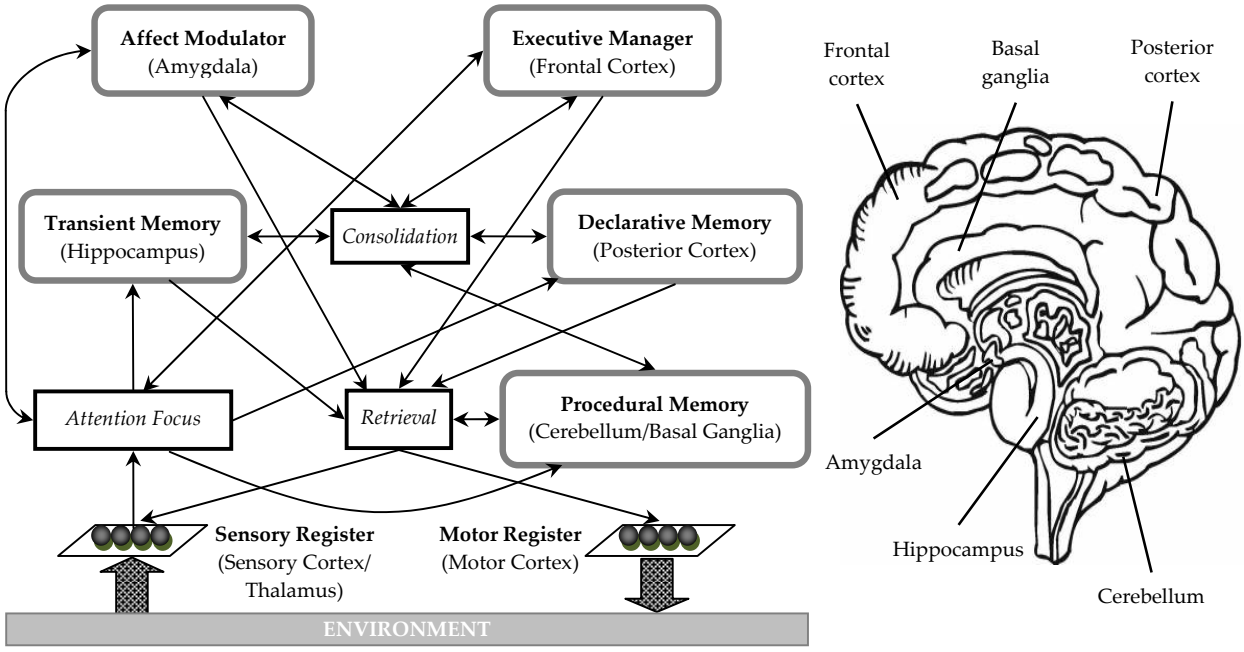


Figure 3.3: Outline of the Integrated Neuro-Cognitive Architecture (INCA)

tic rules. With this ability, NFS compares favorably to the conventional neural network, statistical pattern recognition, and machine learning approaches, which typically lack the explanatory power to describe the salient knowledge structures, association patterns, or causal dependencies in data [151, 60]. On the other hand, NFS addresses the limitations of traditional fuzzy systems, which require extensive human intervention in their design and yield static rules that cannot be further tuned after their initial setup [151].

3.3.1 Architectural Modules

This section provides a system-level description of the constituent modules in INCA, as outlined in Figure 3.3. The *sensory* and *motor registers* constitute the two elementary interfaces for the architecture to receive sensory inputs from and to carry out its motor commands into the environment, respectively. These registers implement basic pre- or post-processing procedures that emulate the functional (i.e., not biological) aspects of the reactive, primitive perceptual and action systems in the brain, respectively. At present, it is assumed that the sensory and motor registers possess the innate ability to extract relevant sensory inputs and reliably hold and execute motor commands output by INCA, or else delegate those functionalities to some other subsystems. This is a separate issue altogether, which can be addressed more effectively during deployment.

The *transient memory* is a mediator for the formation of the long-term declarative, procedural, and executive memories. It emulates the role of the hippocampal region in the brain in rapidly encoding perceptual information contributing to our experiences or episodic knowledge [200, 62]. To realize this, a transient NFS model has been developed that employs a rapid online learning mechanism to acquire knowledge from the environment. The NFS model is configured to exhibit a pattern separation trait, i.e., distinct events are encoded using separate, non-overlapping (rule) representations. Existing long-term memory content may conversely be used to direct learning or interleave the old knowledge with new information acquired in the transient memory. Further details of the transient NFS model can be found at chapter 5.

The *declarative memory* captures general facts about the world or self (i.e., the "what") that are relatively static over time. In contrast to the rapid, online learning of transient memory, it carries out a slow, localized learning process atop some rule neighborhood that reflects the stable storage in the posterior cortex [200] and is useful to capture the salient (statistical) properties of the world. A slow-learning localized NFS has been built upon this principle, whose learning and inference utilize at any time only a small portion of the knowledge (rule) base. It includes several rule base reduction mechanisms to enhance its system scalability and interpretability. The detailed description of the localized NFS model shall be presented in chapter 4.

The *procedural memory* learns and retains knowledge about working skills or procedures (i.e., the "how"). One important role of this module is to regulate the generation of motor actions. In this, trajectories of movement are learned via local error correction on embedded synaptic weights, that roughly corresponds to the functional role of the cerebellum and basal ganglia regions in the brain [3, 121, 62]. Another key function of the module is to provide modulatory (gating) signals to the executive manager, emulating the role of basal ganglia in switching between rapid updating and robust maintenance of information in the frontal cortex [200, 199]. These roles can be simulated via the localized NFS model described before, which is similar to that of the declarative memory, except that it focuses on capturing procedural knowledge.

The metacognitive functions of INCA are governed by the affect modulator and executive manager systems. The *affect modulator* is concerned with motivations and their interactions (i.e., the "why"), and its function is tailored within the executive manager. The relevance of the affect modulator to INCA operations lies in the fact that it provides a context in which goal setting and reinforcement signals are determined for the other subsystems. Cognitively, this module can be viewed as a functional analogue of the limbic circuit in the brain, especially the amygdala, which provides the goals to direct system operations and modulate the memory consolidation processes [19, 62]. A plausible implementation of this module is via the transient NFS model mentioned previously.

Lastly, the *executive manager* is used for active monitoring, control and orchestration of cognitive processes to enhance overall system performance. This role maps functionally to that of the frontal cortex in the brain [200, 121], such as to change or interrupt ongoing processes in the transient, procedural and declarative memories, tune the modules' external (free) parameters, etc. Regulation is also achieved via setting reinforcement functions on the basis of motivation signals from the affective modulator. For these purposes, the executive manager includes a number of sub-modules: a goal setting manager, a parameter tuning manager, and a reinforcement metrics manager. Each can be implemented algorithmically or via the localized NFS model described before.

3.3.2 Neuro-Fuzzy System Modeling

The NFS modeling approach adopted in INCA essentially represents both explicit and implicit knowledge [251] in an integrated fashion. The schematic illustration of an NFS that models the Newton's law $force = mass \times acceleration$ is given in Figure 3.4, which involves a dual perspective. The connectionist perspective of the model, which includes the interconnection of neurons and the weight parameters, captures the implicit knowledge of the system; conversely, the symbolic perspective, that comprises fuzzy rules and their supporting fuzzy antecedent/consequent labels, corresponds to the explicit knowledge. This approach bears some similarities to the localist-distributed representation as described in [251, 252], although there is no physical separation between the connectionist and symbolic knowledge bases in the former approach.

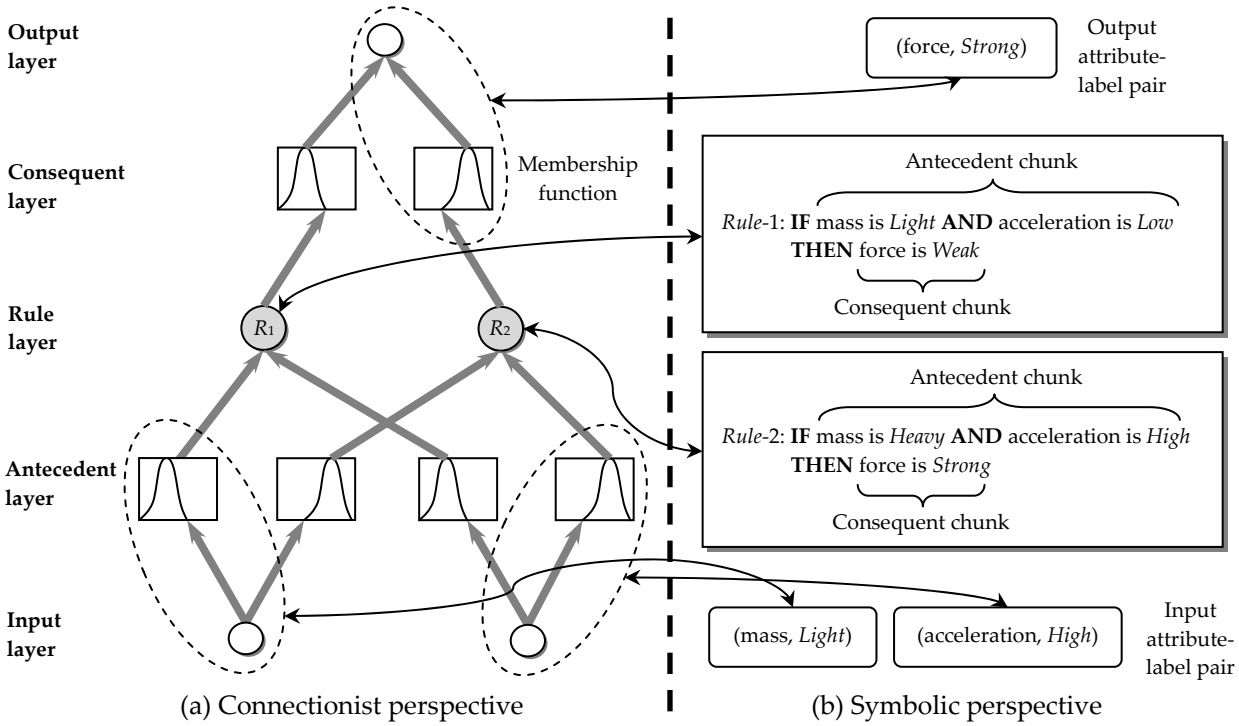


Figure 3.4: Neuro-fuzzy system modeling approach

Here an NFS model is conceptualized as having three basic components: *attribute-label pair*, *rule*, and *chunk*. The input and antecedent (output and consequent) layer neurons at the connectionist side map to the input (output) attribute-label pairs (*attribute*, *label*) at the symbolic side, where *attribute* indicates a feature of an object and *label* is the linguistic label that characterizes that object. For example, an attribute-label pair (mass, *Light*) in Figure 3.4 describes an object whose mass is light. Each *label* at the symbolic side is associated with a set of weight parameters at the connectionist side that defines a membership function (e.g. Gaussian, trapezoidal, etc). The input to the NFS is thus formed by a collection of attribute-label pairs, each capturing the state of an object at a given time. Meanwhile, each rule layer neuron at the connectionist side map to a (fuzzy) rule at the symbolic side. For example, the neuron R_1 represents a symbolic rule *Rule*-1 saying "IF mass is *Light* AND acceleration is *Low* THEN force is *Weak*".

Correspondingly, chunk refers to a conjunction of attribute-label pairs that forms a cluster in the multi-dimensional attribute space capturing the overall aspect of an object. Each rule is linked to an antecedent (condition) chunk and a consequent (conclusion) chunk. In the case of *Rule*-1, the antecedent chunk is (mass, *Light*) $\wedge$ (acceleration,

Low) and the consequent chunk is (force, *Weak*), which comprise two and one attribute-label pairs respectively. Another rule *Rule*-2, with antecedent chunk of (mass, *Heavy*) $\wedge$ (acceleration, *High*) and consequent chunk of (force, *Strong*), has a different set of labels for the same set of attributes, hence corresponding to a different object. Collectively, the set of antecedent chunks, consequent chunks, and rules forms a (fuzzy) rule base.

The relationship between the pieces of information in an NFS model can subsequently be summarized as follows. The activation levels of the input layer nodes are matched with the membership functions in the antecedent layer nodes. The degree of matching between an input node and an antecedent node (corresponding to an input attribute-label pair), called input membership degree, is calculated in this process. The membership degrees of all input attributes are then aggregated and propagated to the rule layer nodes. Next, the consequent layer nodes to which the rule layer nodes are connected are activated to compute the output membership degrees (associated with output attribute-label pairs). Lastly, the output layer nodes weigh the output membership degrees and aggregate them to derive the final output activation levels. Altogether, these steps make up the inference procedure supporting the decision-making processes in NFS. The detailed mathematical aspects of this procedure shall be presented in chapters 4 and 5.

On the other hand, the formulation of labels and rules in an NFS model is achieved via combining in numerous ways structural and parametric self-organizing mechanisms based on nonassociative (unsupervised) and associative (supervised or reinforcement) learning techniques. For example, unsupervised clustering can be performed independently on the input and output attribute spaces to derive the relevant antecedent and consequent label nodes, respectively. Rule nodes may subsequently be created using Hebbian correlation learning [103], followed by tuning of the parameters of all rules and labels via supervised learning method such as back-propagation [225], or reinforcement method such as TD-learning [258], depending on whether the nature of the feedbacks from the environment is instructive or evaluative, respectively. Further illustrations on how the NFS rule base formulation is carried out can be found in chapters 4 and 5.

3.3.3 Communication Protocols

The *attention focus protocol* in INCA realizes the concept of selective attention in human cognition. The key objective is to select the most relevant sensory features, or combine (transform) the existing features into new, more informative ones, possibly guided by the goal and/or contextual signals from the affect modulator and executive manager. This allows in turn a focused cognitive processing that reduces the computational load of the operations in the remaining INCA modules. Various feature selection and combination methods can be used to realize this role [55, 97].

The *retrieval protocol* functions to exploit the (associative) knowledge base captured in all INCA modules, and constitutes a central procedure in the INCA inference cycle. This can take various forms, ranging from aggregating the outputs of the transient and declarative/procedural/executive memories to achieve a better collective performance than that of the individual modules, to retrieving goal and contextual information from the affect modulator and executive manager to modulate the operations of the remaining modules. The results of this protocol would ultimately be transmitted to the motor register to produce actions affecting the external environment, or to the sensory register to influence the internal system operations in the next cycle.

The *consolidation protocol*, central to the INCA consolidation cycle, realizes the transfer of knowledge initially stored in the transient memory (that learns fast but is more plastic) into the long-term storage (that is more resilient yet learns slowly). Our modeling approach is specifically built upon the idea of *interleaved learning* between the transient hippocampal memory and long-term cortical memory in the brain [162, 75, 18], modulated by the affect signals from the amygdala memory. In this scheme, new information is first kept wholesale in the hippocampus and then continually read into the cortex and interleaved with the previously acquired knowledge. Such procedure has the benefit of allowing the cortex to be optimized for the gradual discovery of shared structures (experiences), while enabling the hippocampus to readily absorb new information (exploration) without interfering with the existing knowledge.

The basic infrastructure on which the above three key protocols operate involves a

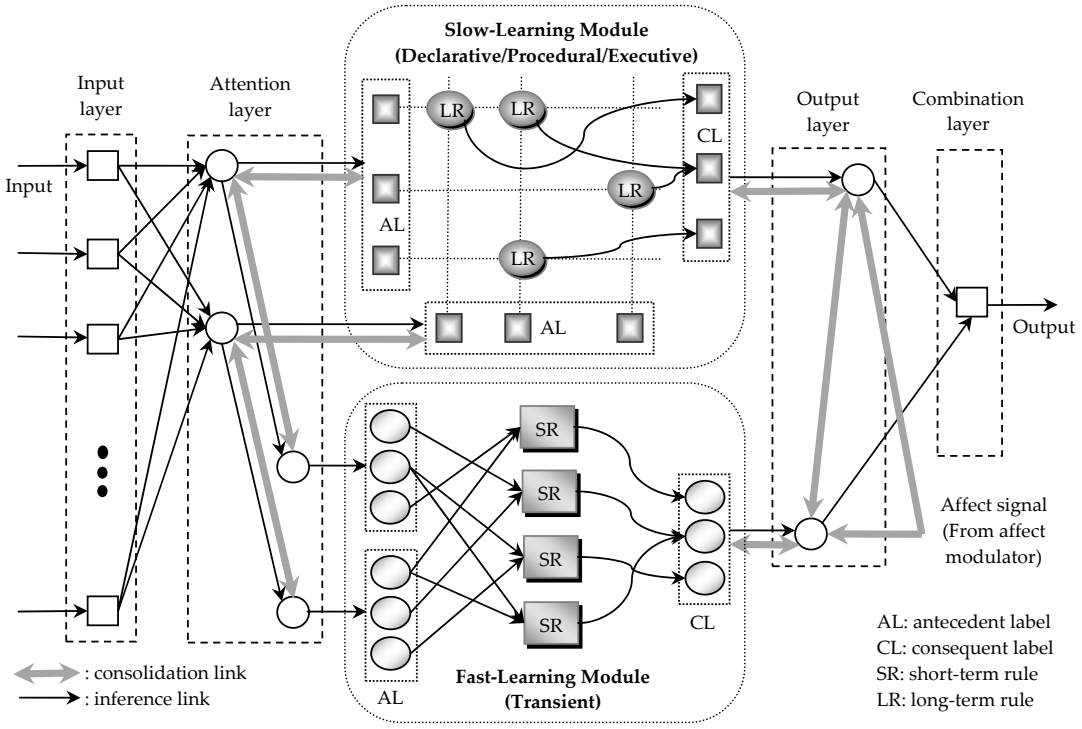


Figure 3.5: Dual consolidation network in INCA

Dual Consolidation Network (DCN), comprising a slow-learning module (SLM) and a fast-learning module (FLM), realized using the slow-learning localized NFS and fast-learning transient NFS discussed earlier, respectively. DCN also includes an attention layer on which the attention focus protocol is carried out to reduce the problem dimensionality, and a combination layer that aggregates the output signals from SLM and FLM to support partly the retrieval protocol. Meanwhile, the communication between the two modules is achieved via the consolidation and inference links, which form the key pathways for the consolidation and retrieval protocols, respectively. Figure 3.5 shows an example of DCN architecture comprising two (attended) inputs and an output. Further details about the DCN model can be found in chapter 5.

3.4 Operational Cycles

The macro-level operations of the proposed INCA framework are governed by its consolidation and inference cycles, which may run in parallel and interact through the system's knowledge base. For illustration purposes, the two operational cycles are described here using an open-ended scenario of automated driving skill acquisition by an intelligent car.

Preliminary research has been conducted that employed the INCA procedural memory to realize operational maneuvers (e.g. reverse or parallel parking, U-turn, lane-following, etc), and more recently a tactical decision-making module to execute a sequence of maneuvers [203, 202]. In extension, this section presents several cases whereby the acquisition of higher-level cognitive functions is required to realize sophisticated, human-like driving skills. The goal of this work is thus not only realize autonomous driving capabilities, but also understand (and model) the acquisition of complex cognitive skills.

In this example, the embodied INCA-based car agent operates in a virtual traffic world, and has innate senses and effectors corresponding to the sensory and motor registers respectively. The space-time of the environment is continuous. The sensory register realizes several primitive sensing functions to extract features of interest from the environment (e.g. edge detection from visual inputs, low-pass noise filter for auditory inputs). Meanwhile, the effectors are realized in the motor register that sends out driving control signals to the environment (e.g. steering angle, acceleration/brake efforts, gear shifting).

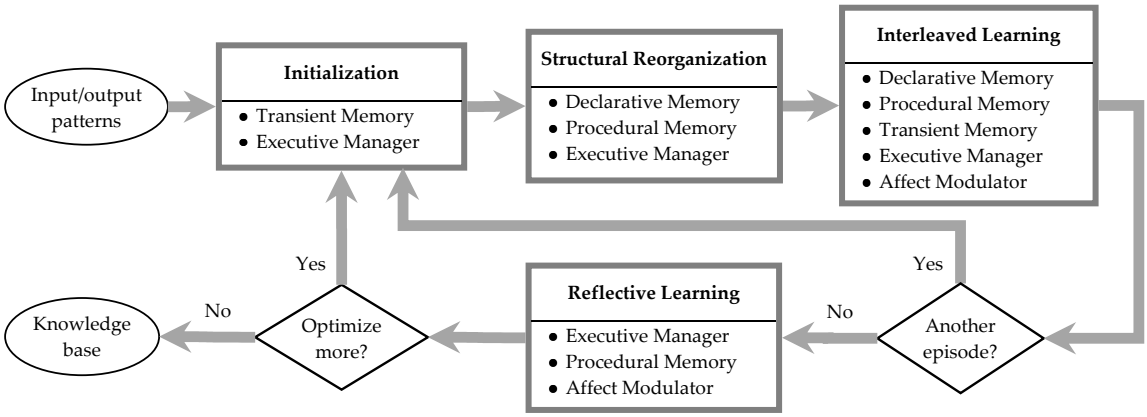
3.4.1 Consolidation Cycle

The consolidation cycle in INCA constitutes the primary mechanism for acquisition, transfer and synthesis of the entire knowledge base, carried out internally at a recurring time (periodical- or event-based) whenever acquisition/updating of knowledge is needed. Our general assumption is that the task at hand consists of various experiences and can be decomposed into independent data chunks occurring at different time periods (episodes). The major steps of the consolidation cycle are summarized in Figure 3.6(a), where the modules involved at each step are listed in bullets.

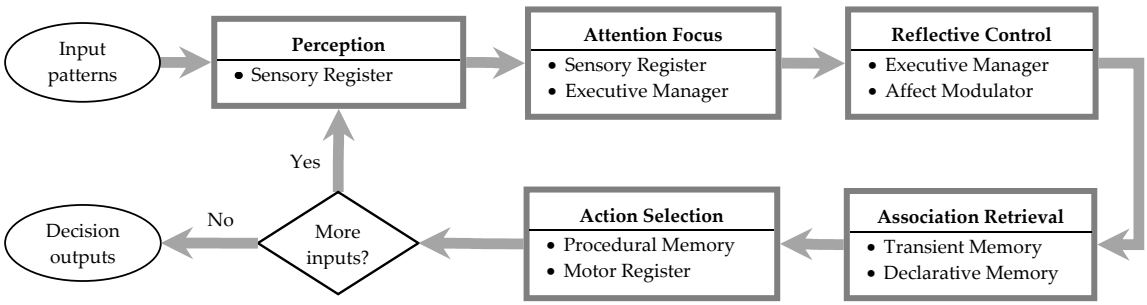
Initialization. The attention focus protocol is first executed to select or combine relevant sensory features, possibly directed by the executive manager, e.g. learning to attend to the left car mirror stimulus before changing to a left road lane. Based on these attended features, the transient memory performs a rapid online learning procedure to adapt both its structure and parameters on the fly for every new data pattern from the environment. Also, the transient memory may receive patterns reinstated from the long-term mem-

ory (termed *pseudopatterns*) that reflect part of the existing knowledge therein. Here a pseudopattern is derived based on the parameters (e.g. the centroid coordinates of the antecedent and consequent labels) of a rule randomly selected in long-term memory. In turn, if presentation of a pattern causes the transient memory to expand beyond maximum capacity, replacement procedure (e.g. deletion of the least recently used rule) is carried out to keep the memory bounded (as is the hippocampus). In our car example, this concerns rapid encoding in the transient memory of the recent driving traces and/or some previous experiences reinstated from the long-term driving rule base.

Structural Reorganization. Associative learning is carried out using pseudopatterns reinstated from the current transient memory representation to craft the coarse rule base structure in the long-term memory. It begins by creating many (linguistic) label neurons, then removes those with low significance degrees, and finally allocates space for the (fuzzy) rules based on the surviving neurons. This process emulates the initial development stage of human cortical memories, where coarse structure is first laid out by an activity-independent process [121]. Active neurons then stabilize via the uptake of trophic factors, whereas the inactive ones eventually die. In our car example, this concerns the initial formation of fuzzy labels such as (*Mild, Steep*) for road angle, or (*Far, Near*) for distance to obstacle, and the corresponding rules generated, e.g. "IF road angle is *Steep* and distance to obstacle is *Near* THEN decision is *Slow down*." This initial structure is then reduced by discarding rule antecedent links that do not contribute crucially to the outputs, and then pruning duplicate rules having the same antecedents. As a result, a highly intuitive and concise set of rules can be attained. For example, the above driving rule simplifies to "IF distance to obstacle is *Near* THEN decision is *Slow down*", as the road angle becomes insignificant when the distance to obstacle is critical. Further reorganization may involve a chaining of rules to devise explicit plans (as in AI planning methods) and a segmentation of action sequences to build hierarchies of sequential behaviors [250]. They form the primary mechanisms for prediction and imagination, on which higher-level capabilities are built. In our car scenario, these concern devising hierarchical structures to support various car anticipation and navigation functions.



(a) Consolidation cycle



(b) Inference cycle

Figure 3.6: The two operational cycles in INCA

Interleaved Learning. This phase involves a rehearsal of the pseudopatterns (approximating some new and old actual patterns) generated from the transient memory to tune the weight parameters of the long-term memory in an offline manner. The pseudopatterns are derived in the same way as that from the long-term memory in the initialization phase. Feeding together the new and old information into the long-term memory would subsequently lead to an interleaved learning regimen that allows new information to be incorporated very gradually, while refreshing the existing (old) knowledge so as to not forget it. This procedure ultimately conforms to the idea that catastrophic interference can be minimized by interleaving old patterns as new patterns are learned [162, 75, 18], and the hippocampus is there to provide initial storage of patterns in a way that avoids interference. In the car example, interleaved learning involves reinstatements of the recent and old driving patterns stored and using them together to tune the long-term semantic driving rules. Such procedure is particularly useful to gracefully handle novel traffic events that are inconsistent with the existing (long-term) driving expertise.

Reflective Learning. This phase serves to evaluate the performances of the INCA modules and then optimize their operations with respect to the task at hand. It constitutes the core metacognitive mechanism in INCA, involving the executive manager, affect modulator, and procedural memory, to allow the system to be self-aware (know what it is doing) and self-improve (know what to do) over time. This can be realized via a search in the space of the modules' external (free) parameters, simulating in a way the role of neuromodulators (e.g. dopamine, acetylcholine, etc) within the frontal cortex and basal ganglia circuits in providing global regulatory signals for learning in the brain [53, 231]. This provides in turn a generic meta-optimization approach that can automatically tune the (other) INCA modules independently of their internal configuration, architecture, or learning method, and that allows evaluating and switching between different (sub)modules. For example, a search can be performed to tune the free parameters of a car module specialized in parking maneuver, so that it improves over time, from the early, erratic execution phase to the expert stage.

3.4.2 Inference Cycle

The inference cycle in INCA realizes a serial flow of reasoning processes, starting from perception and ending with action, which is performed continuously atop the current system's knowledge base. It offers a plausible description of the human knowledge exploitation processes, and shares some similarities with the perception-action cycle described in [80] as well as the nature of system engineering life cycles. Figure 3.6(b) summarizes the inference cycle, which comprises several steps.

Perception. This phase is concerned with primitive information processing based on senses. Specifically, input stimuli from the environment are first kept and encoded in the INCA sensory register. A basic computational analysis is subsequently performed on stimuli containing visual, auditory, and other forms of perceptual information in order to extract the attributes/features of interest relevant to the current task. For instance, the sensory register in our driving example performs task-specific edge detection on its visual inputs to extract the road angle as well as active vision-based depth estimation to

compute the distance to the nearest object ahead.

Attention Focus. The features extracted in the perception phase are filtered via the attention focus protocol, possibly directed by the executive manager and affect modulator. This process can be viewed as a form of competition for attention based on how informative a feature subset or combination compares to others. The attended input features are then broadcast as cues for knowledge retrieval and further processing in the remaining INCA modules. One instance of attention focus procedure in the car scenario is to decide whether to attend to the visual inputs from the left, right, and rear mirrors, before initiating a lane changing maneuver.

Reflective Control. In this phase, the INCA executive manager and affect modulator work jointly to orchestrate the operations of the transient, declarative, and procedural memory modules. This may be in the form of setting goal context for the three modules, rerouting their operations, altering their inference mode, or setting their free parameters. In effect, these enable the system to become aware of and regulate its present state and performance. In the car example, these concern driving mechanisms at a tactical level (e.g. to determine which operational driving module can best handle a maneuver or traffic situation, to set relevant goal information, etc.)

Association Retrieval. The associative knowledge stored in the transient and declarative memories is invoked and aggregated in some ways (via the retrieval protocol) based on the attended stimuli and signals from the executive manager and affect modulator. This may be achieved at different complexity levels via either a basic reactive inference, that generates decisions by directly invoking the association rules, or deliberative inference, that plans decision/action sequences to achieve a goal. An example of the former is the firing of lane switching rules to overtake a slow car ahead, while that of the latter is a composite decision sequence to navigate to a highway exit.

Action Selection. The aggregated outputs from the transient and declarative memories are propagated to the motor register for the execution of motor commands in the

environment or back to the sensory register for the execution of internal actions in the next cycle. They may be transmitted directly or indirectly via the procedural memory. For example, if the aggregated decision output suggests to switch to the next lane before negotiating a road exit, and sensors indicate that it is vacant, the procedural memory would initiate a turn of the steering wheel. These commands are then either executed in the motor register to change the position of the car on the road, or to the sensory register to provide driving proprioception-related information.

3.5 System Evaluation

3.5.1 Cognitive Plausibility

The plausibility of the proposed INCA framework may be evaluated based on a number of cognitive aspects and/or principles that have been established in cognitive sciences. The first evaluation dimension discussed here is memory organization. As outlined in section 3.3.1, the INCA organization explicitly incorporates separate declarative and procedural memory modules, which capture knowledge about general facts pertaining to the world or self and knowledge about skills or procedures, respectively. In addition, INCA includes a distinct affect modulator subsystem for providing intention or goal signals in conjunction with the emotional and metacognitive functions of the system. This modular organization conforms largely to the established distinction among as well as the major brain correlates of declarative, procedural and emotional memories, as elaborated in section 3.1.1.

The second dimension based on which the plausibility of the INCA design can be evaluated is the learning mechanism employed. Section 3.1.2 has distinguished between the nonassociative and associative learning paradigms, and discussed how they are related to the aspects of unsupervised, supervised and reinforcement learning methods widely adopted in cognitive modeling. On the other hand, each INCA module employs a variety of structural and parameter self-organizing mechanisms based on unsupervised, supervised and reinforcement learning procedures, as was explained in section 3.3.2. The design of these procedures, as the later chapters of this thesis show, is largely inspired by and exhibits strong correspondences with the established aspects of learning mechanisms

in the human brain, thereby justifying their cognitive plausibility.

The design of INCA may be further evaluated in terms of the information processing levels that it accommodates. Sloman [243] argued that the human mind can be viewed as a control system involving three levels of abstraction: *reactive*, *deliberative*, and *reflective*. The reactive level is characterized by instantaneous, automatic responses, and operates in a similar manner as a lookup table. At the deliberative level, options and plans are laid out and evaluated via hypotheses (imagination) conducted as internal simulation of the interactions with the external world. Such a system, working atop reactive processes, can plan ahead and make informative decisions. At the highest level, reflective control functions to maintain, monitor, and evaluate the processes invoked and decisions made at the deliberative level, relative to the current goal context. In INCA, reactive mechanisms are embedded within all modules, while deliberative mechanisms may be realized atop the reactive processes within the declarative and procedural memories. Finally, the reflective mechanisms are implemented in the executive manager and affect modulator.

Yet another evaluation criterion of INCA is concerned with the flow of information processing within the architecture. In section 3.4, information processing in INCA was described as consisting of repetitive journeys through the inference and consolidation cycles, which blend the *serial* and *parallel* forms of processing. That is, multiple inference (or consolidation) cycles may proceed at a given time, although they would be funneled through a serial 'bottleneck' when making use of the system's (limited) resources. This idea is closely related to the global workspace theory of information flow [21], which postulates that a serial procession of states emerges from the interactions of many disparate, specialized parallel processes. An implication of this concept is that consciousness imposes seriality on otherwise concurrent/parallel processes.

Finally, INCA may be evaluated in terms of the formal protocol through which its constituent modules interact. With reference to section 3.4 (and the previous paragraph), INCA formally includes in its design specification a consolidation cycle and an inference cycle to regulate its module interactions and thus its macro-level behaviors. These two operational cycles have their conceptual roots at the chief neuroscientific findings of the

human consolidation and perception-action cycles, respectively, as elaborated in section 3.1.3. Such procedures inherently offer a sound principle for unifying different processes and mechanisms of cognition in a systematic and well-defined manner.

3.5.2 Comparisons with Other Architectures

Table 3.1 summarizes the comparisons of design features between the proposed INCA framework and some representative cognitive architectures previously reviewed in chapter 2, including SOAR [138], ICARUS [140], Cortronics [105], IBCA [200], ACT-R [13], CLARION [250], and LIDA [73]. Comparisons are first made in terms of salient design properties, namely architecture type (symbolic, emergent, hybrid), memory organization (declarative, procedural, emotional), learning mechanism (nonassociative, associative), cognitive abstraction level (reactive, deliberative, reflective), information processing flow (serial, parallel), and formal protocol for interaction of modules, the descriptions of which have already been provided in section 3.5.1 as well as chapter 2.

From the table, it can be seen that INCA offers a comparatively comprehensive design that encompasses a broad range of cognitive aspects. Some conceptual similarities between INCA and the other architectures can be inferred, e.g. INCA has declarative and procedural memories as with all the other architectures, INCA has emotional memory (i.e., the affect regulator) similar to ACT-R, LIDA and CLARION, etc. One unique feature that most (if not all) of the other architectures do not share, however, is that INCA realizes *both* consolidation and inference cycles to systematically guide its knowledge self-organization and exploitation processes, respectively. Such approach is closely related to that of the LIDA architecture, which defines its overall operations in terms of a cognitive cycle. This cycle, however, focuses largely on the inference process, whereas consolidation is considered as a secondary mechanism, i.e., it is not entirely clear what the consolidation process in LIDA is composed of or how exactly it is done. By contrast, INCA incorporates an explicit segregation between the consolidation and inference processes, thus providing a more comprehensive framework for modeling human cognition.

Table 3.1 also compares the cognitive architectures in terms of the capabilities they support, the definitions of which were presented in chapter 2 (see section 2.2.2). Again,

it is shown that INCA accommodates a relatively wide variety of capabilities. Language comprehension is one area that the current INCA specification do not address directly, but it is possible to build a natural language processing system for analysis of unstructured text data by connecting the learning memory modules in INCA with the vector-based model of language [157]. One way to achieve this was recently described in [58], whereby enhancement of feature space representing association of concepts is achieved by iterative spreading of activation to include all related concepts of the same semantic type, followed by a reduction process to identify the most informative features including those that do not appear in the original text. The final feature space can then be linked to the INCA memory modules to achieve proper document categorization etc. Further details of this methodology are, nevertheless, beyond the scope of the current thesis.

In a nutshell, the proposed INCA framework provides a considerably rich set of design properties and capabilities as compared to that of the other cognitive architectures listed. However, it should be noted that INCA does not attempt to compete with the existing approaches in AI and cognitive sciences. Instead, it aspires to provide a complementary methodology to address some of the key issues in the current approaches, especially with regard to bridging the gap from biologically-based to symbolic-based approaches, and to build towards the upper level on the state of the art and its developments.

3.6 Summary

This chapter described the INCA framework, beginning with discussions on some brain inspirations and key computational models that pertain to its formulation. The organization and communication protocols of the modules in the architecture were then elaborated, followed by description on the two operational cycles governing the overall system behaviors. The architecture was then evaluated based several cognitive aspects, and its features were compared with those of other notable cognitive architectures. As the first step to build INCA, a novel localized NFS model shall be described in the next chapter that provides the infrastructure for realizing the long-term (declarative, procedural, executive) memory modules within the architecture.

Table 3.1: Comparisons between INCA and other established cognitive architectures

Architectural Feature	SOAR	Icarus	IBCA	Cortronics	ACT-R	CLARION	LIDA	INCA
Design property								
1) Architecture type	Symbolic	Symbolic	Emergent	Emergent	Hybrid	Hybrid	Hybrid	Hybrid
2) Memory organization	DE,PR	DE,PR	DE,PR	DE,PR	DE,PR,EM	DE,PR,EM	DE,PR,EM	DE,PR,EM
3) Learning mechanism	NAS,AS	AS	NAS,AS	NAS	NAS,AS	AS	AS	NAS,AS
4) Abstraction level	RA,DB,RF	RA,DB	RA,DB,RF	RA,DB	RA,DB,RF	RA,DB,RF	RA,DB,RF	RA,DB,RF
5) Processing flow	SE	SE	PA	PA	SE,PA	PA	SE,PA	SE,PA
6) Interaction protocol	-	-	-	-	-	-	CG	CS,IN
Supported capability								
1) Perception	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
2) Problem solving	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
3) Reactivity	Yes	Yes	Yes	-	Yes	Yes	Yes	Yes
4) Executive regulation	Yes	-	Yes	-	Yes	Yes	Yes	Yes
5) Emotion	-	-	-	-	Yes	Yes	Yes	Yes
6) Parallel processing	-	-	Yes	Yes	Yes	Yes	Yes	Yes
7) Hierarchical processing	Yes	Yes	Yes	Yes	Yes	Yes	-	Yes
8) Language comprehension	Yes	-	-	Yes	Yes	-	Yes	Yes

DE = declarative, PR = procedural, EM = emotional, NAS = nonassociative, AS = associative, RA = reactive, DB = deliberative, RF = reflective
SE = serial, PA = parallel, CG = cognitive cycle, CS = consolidation cycle, IN = inference cycle

Chapter 4

Reduced Localized Network Model for Knowledge Extraction

4.1 Knowledge Extraction

This chapter presents a localized NFS model of the cortical system in the brain that is used to realize the long-term memory modules in INCA. The proposed localized model is designed to accomplish two important objectives: to achieve satisfactory performance in associative (i.e., classification or regression) tasks, and to extract useful and comprehensible knowledge structures from data. The first goal, which involves creating predictive models that offer the best fit to the data being modeled, has been the primary focus in statistical pattern recognition, neural networks, and machine learning research. However, in many applications such as medical diagnosis or financial engineering, such black-box models are not satisfactory and the ability to extract salient knowledge structures, association patterns, or causal dependencies from data is of greater importance.

Recent interests in the knowledge extraction paradigm have led to the development of various computational techniques, a popular example of which involves encapsulating knowledge induced from data using a set of expressive, logical rules [60, 151, 183, 100]. In this respect, the self-organizing neuro-fuzzy system (NFS) modeling adopted in the INCA framework provides a promising and suitable approach to extraction of fuzzy rules from data. Armed with the learning capability, parallelism, and robustness of neural network and the human-like, symbolic, and approximate reasoning capacities of fuzzy system, the NFS approach offers an effective means to automatically induce decision logic from raw numerical data in the form of highly intuitive fuzzy linguistic rules [151, 183, 100].

Contemporary approaches to NFS learning and recall broadly consist of two groups: *globalized* and *localized*. In the former, learning and inference are accomplished via a global activation of the entire rule base (i.e., the underlying network). Globalized NFSs, such as [112, 149, 123, 41, 223, 152, 16], typically exhibit good accuracy and generalization performances, since all network parameters are utilized to compute the output for any given input. However, the acquisition of new information in these systems affects all parameters and may cause a catastrophic interference with (or forgetting of) knowledge previously gained. Moreover, the intrinsic network plasticity and the transient nature of the acquisition process often render unstable learning (e.g. oscillation or divergence) that may be undesirable in some applications, such as controller design and signal processing.

An attractive alternative to the globalized method is the localized NFS, whose learning and inference processes involve at a time the activation of only a small portion of the entire rule base defined within certain neighborhood or spatial constraints. One prominent family of localized NFSs is the Fuzzy Cerebellar Model Articulation Controller (FCMAC), first proposed in [115], and the related variants [191, 110, 190, 241, 249, 208, 188, 270]. These systems are well-known to be more resilient to interference and can provide more efficient knowledge recall and learning than the globalized type. However, as only local information is used in learning and inference, one major issue is their lack of an overall view of the domain data, leading possibly to generalization deficiency. As such, an accurate representation of the overall data usually requires large number of rules and/or features, resulting in increased memory requirements and degraded system interpretability.

The model presented in this chapter aims at addressing the aforementioned issues in localized NFSs by exploiting current understanding of cognitive processes in the brain. In particular, neuroscience studies have established that the neural development of human long-term memory from birth, essentially that of the cortical systems, involves two overlapping stages [121, 63]. In the first stage, the basic cortical memory structure and coarse connections are laid out as a result of neurogenesis, producing a large, excessive formation of neurons. In the second stage, these initial structure and connections are reduced and refined via various activity-dependent experiences occurring throughout one's life span.

Active neurons gradually stabilize via the update of neurotrophic factors, whereas the losing neurons eventually perish and the extraneous connections get pruned. These two processes enable humans to consolidate the knowledge learned, compressing information into a compact memory while keeping its semantic content sufficiently accurate.

A new localized NFS model termed the *Reduced Fuzzy Cerebellar Model Articulation Controller* (RFCMAC) is proposed in this chapter that implements the brain inspiration discussed above. Specifically, RFCMAC includes in its learning procedure a *label generation* and a *rule generation* phase, each composed of construction and reduction steps that model the two-stage cortical development process. These are then followed by a *parameter tuning* phase for further refinements of the crafted rule base structure. Compared to conventional localized NFSs, a distinct feature of RFCMAC lies precisely in the provision of effective structural reduction mechanisms to induce a concise and interpretable rule base, while at the same time improving the system's accuracy/generalization. This approach enables the system to deal with large high-dimensional data and to enhance robustness by pruning spurious or redundant fuzzy rules and labels that typically stem from noise or outliers. It is also worth noting that, to the best of our knowledge, RFCMAC is a new and the first FCMAC that employs rule base reduction mechanisms.

Research on FCMAC has yielded a variety of models that provide some inspirations in developing RFCMAC. In [191], for instance, a self-organizing FCMAC model capable of adapting its structure and parameters in real time was developed. Hwang and Hsu [110] proposed an FCMAC with reinforcement learning ability based on stochastic actor-critic model to compute the optimal actions. Meanwhile, to improve the interpretability and address the rigid structural limitations in classical FCMAC, Ng *et al.* [188] devised an FCMAC realizing Compositional Rule of Inference (FCMAC-CRI), and a similar model realizing Yager Inference (FCMAC-Yager) was developed in [241]. Further improvements were made in [190] by integrating Bayesian Ying-Yang learning into FCMAC to derive good input fuzzy sets that can more accurately capture the input data distribution. To improve the learning speed, Su *et al.* [249] proposed a non-uniform credit assignment scheme for updating the FCMAC weights. More recently, Peng *et al.* [208] built a recur-

rent FCMAC for dynamic tasks that has feedback connections capturing the dynamics of a system via time delays. Regardless, in contrast to RFCMAC, all these models still lack a comprehensive account for reduction mechanisms, and their abilities to extract concise, intuitive rules and handle large, high-dimensional data have yet to be demonstrated.

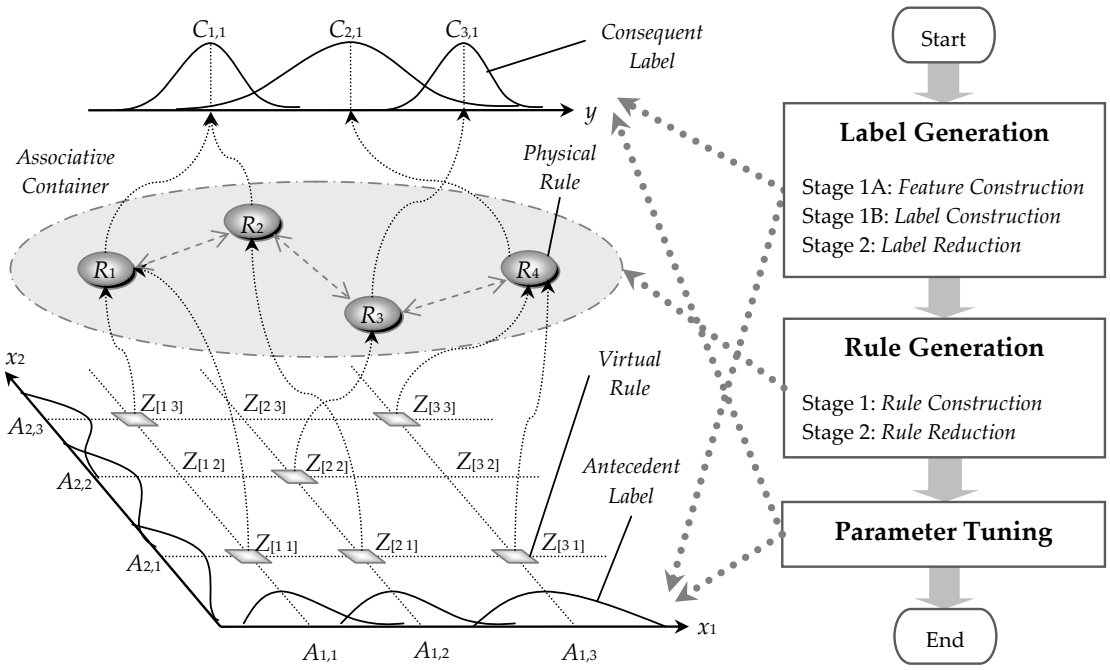
Meanwhile, comparisons can be made with several contemporary NFSs incorporating reduction mechanisms. Chakraborty and Pal [41] developed a feature and rule reduction technique based on certainty factor. This approach, however, requires multiple rule base tuning phases that yield rather slow learning, whereas RFCMAC uses only a single phase. In [16], an evolving fuzzy system is proposed that can dynamically discard ambiguous or old rules, but its reduction method does not yet account for removal of inconsequential input features and labels as in RFCMAC. Ang and Quek [15] proposed the Rough Set-Based Pseudo Outer Product (RSPOP) that employs a consistency measure derived from rough set theory to reduce input features and subsequently discard redundant remaining rules that have inconsequential input labels. Unlike RFCMAC, however, RSPOP does not warrant optimal input-output mapping, for there is no tuning phase after the reduction step. Recently, Liu *et al.* [152] devised the Hebb Rule Reduction (HRR) that addresses this issue by tuning the rule base parameters after feature and label reductions are performed. Yet its reduction is not extended to omit redundant remaining rules, unlike RSPOP and RFCMAC. Furthermore, these models still do not scale very well to large datasets and, being globalized NFS type, remain subject to catastrophic interference.

The architectural framework of the RFCMAC system is first described in section 4.2. Section 4.3 subsequently details its learning procedure, accompanied by a pedagogical illustration in section 4.4. Next, section 4.5 presents some experimental studies conducted to validate the proposed system. Section 4.6 finally concludes this chapter.

4.2 System Architecture

4.2.1 Connectionist Structure

The RFCMAC system is a fuzzy associative memory that is built upon the original CMAC [4] which models the physiological and localized learning traits of the cortical circuits in



(a) Connectionist structure (b) Learning procedure

Figure 4.1: Overview of the proposed RFCMAC system

the brain, and expanded to model fuzzy linguistic rules to provide system interpretability similar to [241]. It realizes the *Mamdani model* of fuzzy rule [156], where the input and output feature spaces are partitioned into a set of antecedent and consequent fuzzy sets/labels, respectively, in a non-uniform manner. The model is defined in (Eq. 4.1)

$$\begin{aligned} &\text{IF } x_1 \text{ is } A_{1,j} \text{ and} \ldots x_i \text{ is } A_{i,j} \text{ and} \ldots x_I \text{ is } A_{I,j} \\ &\text{THEN } y_1 \text{ is } C_{l,1} \text{ and} \ldots y_m \text{ is } C_{l,m} \text{ and} \ldots y_M \text{ is } C_{l,M} \end{aligned} \qquad \text{(Eq. 4.1)}$$

where x_i is the i^{th} input feature of interest (e.g. velocity and distance to obstacle in the case of a car control system [203, 202]), y_m is the m^{th} output feature (e.g. brake control), and I and M are the total number of inputs and outputs respectively, $A_{i,j}$ is the j^{th} antecedent label of the ith input (e.g. *Slow, Fast* and *Near, Far* for the velocity and distance inputs, respectively), and $C_{l,m}$ is the l^{th} consequent label of the mth output (e.g. *Weak, Strong* for the brake control output). An example of RFCMAC architecture with two inputs x_1, x_2 and an output y is shown in Figure 4.1(a).

The Mamdani model adopted in RFCMAC essentially belongs to the *linguistic fuzzy modeling* (LFM) approach [146], which focuses on good human interpretability of the underlying model that is paramount for knowledge mining and analysis tasks. Another

major approach is the *precise fuzzy modeling* (PFM) [146] that aims at obtaining high modeling accuracy/precision. A prime example of this approach is the Takagi Sugeno fuzzy model [263], which replaces the output fuzzy sets in the Mamdani model with an additive function of the input features. However, this functional expression of the rule consequents does not lend itself well to human comprehension, and is thus incompatible with the interpretability goal of the RFCMAC system (as mentioned in section 4.1). This provides the motivation behind employing the Mamdani model in the RFCMAC system.

In the RFCMAC architecture, an arbitrary virtual rule cell is denoted as $Z_{[j_1,\ldots,j_i,\ldots,j_I]}$, where $[j_1,\ldots,j_i,\ldots,j_I]$ refers to the *address* of the rule cell. The total number of possible rules in RFCMAC is thus $K = \prod_{i=1}^{I} J_i$, where J_i is the number of antecedent labels for the ith input. This shows that the number of rules depends heavily on the dimensionality of the inputs and the number of labels in each input. In practice, however, only a small portion of the rule space is actually used, and allocating memory for all K rules may be unreasonable, especially for large datasets. A method is thus needed to map this large virtual rule space into a physically small storage, analogous to the mapping between virtual and physical memory spaces in computer systems.

While it is possible to use hash map, for instance, this approach imposes the risk of rule collisions, where distant rules are mapped into the same location, possibly yielding undesirable generalization and degraded performance [283]. In light of this issue, RFCMAC stores its physical rules using an *associative container* data structure instead, which is implemented as simple bidirectional linked list that supports both forward and backward traversal of its elements. This data structure ensures that only the physical rules relevant to the current domain data are stored and that each rule is unique, thus eliminating collisions. The only tradeoff (as compared to hashing) is slower neighborhood invocation due to the need to trace from the first list element to know the rules corresponding to the activated antecedent labels. However, this extra time needed is generally negligible, as the number of physical rules is usually small in practice.

This idea is illustrated in Figure 4.1(a), where virtual rules are mapped into physical rules using an associative container. Each physical rule R_k in the container has access via

the bidirectional links to its adjacent rules i.e., trained physical rules whose corresponding virtual rule cells are located closest to that of R_k. Such access allows a fast traversal from one physical rule to its neighboring rules, whose antecedent labels are invoked during the inference process. Figure 4.1(a) also shows that some virtual rules can map to the same physical rule (e.g. $Z_{[11]}$, $Z_{[13]}$ map to R_1). This does not imply a collision but indicates that the antecedent links from some inputs are omitted (e.g. R_1 says: "IF x_1 is $A_{1,1}$ THEN y is $C_{1,1}$", ignoring input x_2). This omission results from the rule reduction phase of the RFCMAC learning process and will be detailed in section 4.3.2.

4.2.2 Inference Scheme

The RFCMAC architecture is generic and can accommodate various fuzzy inference schemes. For illustration and experimentation purposes, the so-called Yager inference scheme [124] is adopted in the current work, yielding the RFCMAC-Yager system. In this scheme, the system output is computed based on the degree of dissimilarity between input and rule antecedent. Its key feature is that, when the input perfectly matches the rule antecedents, the final output exactly matches the rule consequents. Such deduction process is conceptually sound, for it maps closely to the material implication in classical Boolean logic while corresponding to the humans' intuitive reasoning [195, 241]. The basic idea and illustrative example of the Yager inference can be found in Appendix A.

In this work, Gaussian membership function (MF) is used to describe the antecedent and consequent labels in the RFCMAC-Yager system. The computation of the final crisp output y_m at the m^{th} dimension based on the Yager inference is defined in (Eq. 4.2)

$$y_m = \frac{\sum_{k \in \mathbf{S}} \frac{m_{(l,m)_k} f_k}{\sigma_{(l,m)_k}(2-f_k)}}{\sum_{k \in \mathbf{S}} \frac{f_k}{\sigma_{(l,m)_k}(2-f_k)}} \qquad \text{(Eq. 4.2)}$$

where $\mathbf{S}$ is the set of (physical) rule indices being selected, $m_{(l,m)_k}$ and $\sigma_{(l,m)_k}$ respectively denote the centroid and width of the MF in the consequent label $C_{(l,m)_k}$ linked to the selected rule R_k with firing strength f_k, as per (Eq. 4.3)

$$f_k = \prod_{i=1}^{I} \left(1 - d_{(i,j)_k}\right) \qquad \text{(Eq. 4.3)}$$

The term $d_{(i,j)_k}$ in (Eq. 4.3) essentially computes the degree of dissimilarity between the input x_i and antecedent label $A_{(i,j)_k}$ (whose MF is defined by the centroid $m_{(i,j)_k}$ and width $\sigma_{(i,j)_k}$) connected to rule R_k, as per (Eq. 4.4)

$$d_{(i,j)_k} = 1 - \exp\left(-\frac{\left(x_i - m_{(i,j)_k}\right)^2}{\sigma^2_{(i,j)_k}}\right) \tag{Eq. 4.4}$$

Meanwhile, the (localized) selection of rules in RFCMAC-Yager is determined based upon the neighborhood set $\mathbf{N}_i$ of the selected, relevant antecedent labels, as per (Eq. 4.5)

$$\mathbf{N}_i = \begin{cases} \{1\} & \text{if } m_{i,1} > x_i \\ \{J_i\} & \text{if } m_{i,J_i} < x_i \\ \{j, j+1\} & \text{if } \exists j : m_{i,j} \leq x_i \leq m_{i,j+1} \end{cases} \tag{Eq. 4.5}$$

Here each instance $[j_1, \ldots, j_i, \ldots, j_I]$ of the index combination set $\{\mathbf{N}_1, \ldots, \mathbf{N}_i, \ldots, \mathbf{N}_I\}$ uniquely addresses a virtual rule $Z_{[j_1,\ldots,j_i,\ldots,j_I]}$, which then maps to a selected physical rule R_k (where $k \in \mathbf{S}$). In the context of associative container representation, the rule index k is identified by traversing from the first list elements until a rule node that is linked to the set of selected antecedent nodes $A_{i,j}$ (where $j \in \mathbf{N}_i$) is found.

4.3 Learning Procedure

The learning procedure of the RFCMAC system consists of three phases: *label generation*, *rule generation*, and *parameter tuning*, as elaborated in sections 4.3.1-4.3.3 respectively. Figure 4.1(b) provides an overview of the procedure along with its correspondence to the connectionist structure, whereby each of the first two phases further comprises construction and reduction steps modeling the two-stage neural development explained in section 4.1. A complexity analysis of the learning procedure is also presented in section 4.3.4.

4.3.1 Label Generation

The label generation phase aims at identifying the input features that are most relevant to the final outputs (i.e., classification/approximation made by the system), and subsequently partitioning the data space into a set of input/output fuzzy labels. This phase comprises the following three steps:

Stage 1A: Feature Construction. This step serves to select a (sub)set of input features that are highly correlated to the output but uncorrelated to one another. Starting from an empty subset, features are added one at a time until D features are selected, where D is empirically set as $1 \leq D \leq I$, and the feature subset selected in each step is one that maximizes a correlation function [99] given in (Eq. 4.6)

$$C_F = \frac{n\overline{R_{fy}}}{\sqrt{n + n(n-1)\overline{R_{ff}}}} \qquad \text{(Eq. 4.6)}$$

where C_F is the *merit* of a feature subset $\mathbf{F}$ containing $n \leq D$ features, $\overline{R_{fy}}$ is the average correlation between features $f \in \mathbf{F}$ and output y, and $\overline{R_{ff}}$ is the average correlation among different features $f \in \mathbf{F}$. The numerator of (Eq. 4.6) indicates how relevant a feature subset is, while the denominator denotes how much redundancy exists among the features. Consequently, by adding each time a feature that maximizes C_F, a set of unique, informative features with low redundancy is obtained. The chosen features are used from step 2 onwards, and their orders are kept for the later rule generation phase.

In (Eq. 4.6), the degree of correlation between any two features A and B is computed using a *symmetrical uncertainty* measure [214], which is chosen due to its simplicity and low bias for multi-valued features. The measure is expressed in (Eq. 4.7)

$$R_{AB} = \frac{2I(A,B)}{H(A) + H(B)} \in [0,1] \qquad \text{(Eq. 4.7)}$$

where $I(A,B) = H(A) + H(B) - H(A,B)$ is the *information gain* (also called *mutual information*) between features A and B, $H(A)$ is the entropy of A, and $H(A,B)$ is the joint entropy of A and B. To obtain $H(A)$ and $I(A,B)$, the probability distributions of A and B are typically computed a priori via discretization [67] or kernel estimator [201]. However, this estimation is rather expensive, with a time complexity up to $O(N^3)$, where N is the number of data instances. For simplicity, normal distribution is assumed instead to avoid the expensive distribution estimation. While this assumption may give less accurate estimation for highly non-normal distribution, it is expected that this can compensated by the rule reduction process in the later learning phase (see section 4.3.2), which allows further refinement of input features in order to more accurately reflect the

actual data distribution. The notion of differential entropy [144] in (Eq. 4.8) is used in RFCMAC, and the corresponding information gain is given in (Eq. 4.9)

$$H(A) = \frac{1}{2}\left(1 + \log\left(2\pi\sigma_A^2\right)\right) \tag{Eq. 4.8}$$

$$I(A,B) = -\frac{1}{2}\log\left(1 - \rho_{AB}^2\right) \tag{Eq. 4.9}$$

where σ_A^2 is the variance of feature A and ρ_{AB} the Pearson's correlation between A and B [90]. In this manner, the complexity of computing $H(A)$ and $I(A,B)$ reduces to $O(N)$.

Stage 1B: Label Construction. The initial antecedent and consequent labels are crafted in this step. For each p^{th} training data point ($p \in \{1,\ldots,N\}$), if the overall similarity $S^{(p)}$ between the inputs and selected antecedent labels, computed in (Eq. 4.10) based on the dissimilarity degree $d_{i,j}^{(p)}$ as defined in (Eq. 4.4), is lower than a split threshold δ, then a new label is created at every input and output dimension.

$$S^{(p)} = \frac{1}{D}\sum_{i=1}^{D}\left(1 - \min_{j\in\{1,\ldots,J_i\}}\left\{d_{i,j}^{(p)}\right\}\right) \tag{Eq. 4.10}$$

The centroid of the newly generated label is initialized to the current data point, and the width is set in proportion to the scale of the respective feature. In essence, δ controls the coverage of data by the labels and affects the number of initial labels (and initial rules). Its value is chosen empirically as $0 < \delta \leq 1$. Larger δ yields more labels, and vice versa.

When $S^{(p)} \geq \delta$, Hebbian learning [103] is performed based on the current data point to update the winning antecedent and consequent labels $A_{i,j*}$ and $A_{l*,m}$, having the lowest dissimilarities $d_{i,j*}^{(p)}$ and $d_{l*,m}^{(p)}$ with respect to input x_i and target output t_m, respectively. The weights $w_{i,j*}^{(p)}$ of $A_{i,j*}$ and $w_{l*,m}^{(p)}$ of $C_{l*,m}$ are updated using (Eq. 4.11)-(Eq. 4.12)

$$\Delta w_{i,j*}^{(p)} = \frac{1}{DM}\left(\sum_{i'=1}^{D}(1 - d_{i',j*}^{(p)})\right)\left(\sum_{m'=1}^{M}(1 - d_{l*,m'}^{(p)})\right) \tag{Eq. 4.11}$$

$$\Delta w_{l*,m}^{(p)} = \frac{1}{DM}\left(\sum_{i'=1}^{D}(1 - d_{i',j*}^{(p)})\right)\left(\sum_{m'=1}^{M}(1 - d_{l*,m'}^{(p)})\right) \tag{Eq. 4.12}$$

where $\Delta w_{i,j*}^{(p)} = w_{i,j*}^{(p+1)} - w_{i,j*}^{(p)}$ and $\Delta w_{l*,m}^{(p)} = w_{l*,m}^{(p+1)} - w_{l*,m}^{(p)}$. For every new label created, its weight is initially set as one. The accumulated weights $W_{i,j} = w_{i,j}^{(N)}$ and $W_{l,m} = w_{l,m}^{(N)}$ are later used to rank the antecedent and consequent labels in each dimension, respectively.

Stage 2: Label Reduction. Antecedent (or consequent) labels in each dimension are successively evaluated based on their accumulated weights, starting from the highest rank. In each evaluation, among the labels in every input, the label $A_{i,j'}$ is chosen whose centroid is the nearest to that of label $A_{i,j}$ being evaluated ($j' \neq j$). If their overlapping degree exceeds a merging threshold α (empirically set as $0 < \alpha \leq 1$), the two labels are combined as a new label $A_{i,j"}$ and $A_{i,j'}$ is deleted, thus improving interpretability; otherwise, $A_{i,j'}$ remains. In sum, α governs the tradeoff between accuracy and interpretability. Higher α can improve accuracy but may reduce interpretability, and vice versa.

The centroid and width of $A_{i,j}$ and $A_{i,j'}$, $(m_{i,j}, \sigma_{i,j})$ and $(m_{i,j'}, \sigma_{i,j'})$, are then merged into $(m_{i,j"}, \sigma_{i,j"})$ via (Eq. 4.13)-(Eq. 4.14), and the new label weight is defined in (Eq. 4.15)

$$m_{i,j"} = \frac{m_{i,j}W_{i,j} + m_{i,j'}W_{i,j'}}{W_{i,j} + W_{i,j'}} \tag{Eq. 4.13}$$

$$\sigma_{i,j"} = \frac{\max\{m_{i,j} + \sigma_{i,j}, m_{i,j'} + \sigma_{i,j'}\} - \min\{m_{i,j} - \sigma_{i,j}, m_{i,j'} - \sigma_{i,j'}\}}{2} \tag{Eq. 4.14}$$

$$W_{i,j"} = W_{i,j} \tag{Eq. 4.15}$$

In this process, the overlapping degree between $A_{i,j}$ and $A_{i,j'}$ is given in (Eq. 4.16)

$$\varphi(A_{i,j}, A_{i,j'}) = \max\left(\frac{|A_{i,j} \cap A_{i,j'}|}{|A_{i,j}|}, \frac{|A_{i,j} \cap A_{i,j'}|}{|A_{i,j'}|}\right) \tag{Eq. 4.16}$$

where $|A_{i,j}|$, $|A_{i,j'}|$ and $|A_{i,j} \cap A_{i,j'}|$ are computed based on the centroids and widths of the labels via (Eq. 4.17)-(Eq. 4.18) respectively, assuming $m_{i,j} \geq m_{i,j'}$, as in [150]

$$|A_{i,j}| = \sqrt{\pi}\sigma_{i,j} \tag{Eq. 4.17}$$

$$\begin{aligned} |A_{i,j} \cap A_{i,j'}| = {} & \frac{h^2\left(m_{i,j'} - m_{i,j} + \sqrt{\pi}\left[\sigma_{i,j} + \sigma_{i,j'}\right]\right)}{2\sqrt{\pi}\left(\sigma_{i,j} + \sigma_{i,j'}\right)} \\ & - \frac{h^2\left(m_{i,j'} - m_{i,j} + \sqrt{\pi}\left[\sigma_{i,j} - \sigma_{i,j'}\right]\right)}{2\sqrt{\pi}\left(\sigma_{i,j} - \sigma_{i,j'}\right)} \\ & + \frac{h^2\left(m_{i,j'} - m_{i,j} - \sqrt{\pi}\left[\sigma_{i,j} - \sigma_{i,j'}\right]\right)}{2\sqrt{\pi}\left(\sigma_{i,j} - \sigma_{i,j'}\right)} \end{aligned} \tag{Eq. 4.18}$$

where $h(x) = \max\{0, x\}$. The merging of consequent labels at the output side is the same as in (Eq. 4.13)-(Eq. 4.16), except that $A_{i,j}$ is replaced with $C_{l,m}$. In a special case when an input feature has only one label, that feature is deleted and D decreased by 1.

4.3.2 Rule Generation

In this phase, a set of rules linking identified antecedent and consequent labels is constructed. The rules are then reduced/compressed based on certain consistency measure to produce a concise yet accurate rule base. The steps are detailed hereafter.

Stage 1: Rule Construction. The identified antecedent and consequent labels are first sorted based on their centroids in an ascending order, so as to support the neighborhood selection process in (Eq. 4.5). Then for each p^{th} data point, a physical rule R_k, if not already existing in the associative container, is created at the address pointed by the set of currently selected winning antecedent labels (from all inputs) giving the minimum degrees of dissimilarity. This new rule is then linked to all consequent labels in all outputs, with all link weights initially set to zero. As only one rule can be created for each data point, the number of rules K is constrained as $K \leq N$.

Hebbian learning [103] is then carried out for each p^{th} data point to update the weight $w_{k,l,m}^{(p)}$ linking the most influential rule R_k (addressed by the current winning antecedent labels) and the winning consequent label $C_{l,m}$ in each m^{th} output, as defined in (Eq. 4.19)

$$\Delta w_{k,l,m}^{(p)} = f_k^{(p)} \prod_{m'=1}^{M} \left(1 - d_{l',m'}^{(p)}\right), \forall l, l' \in \{1, \ldots, L_m\} \qquad \text{(Eq. 4.19)}$$

where $\Delta w_{k,l,m}^{(p)} = w_{k,l,m}^{(p+1)} - w_{k,l,m}^{(p)}$, $f_k^{(p)}$ is the firing strength of rule R_k, $d_{l',m}^{(p)}$ is the dissimilarity degree between $C_{l,m}$ and target output $t_m^{(p)}$, and L_m is the number of consequent labels in the m^{th} output. The final consequent label $C_{(l,m)_k}$, to which R_k should be linked, is subsequently determined using (Eq. 4.20):

$$C_{(l,m)_k} = C_{l*,m} \mid W_{k,l*,m} = \max_{l \in \{1,\ldots,L_m\}} \{W_{k,l,m}\} \qquad \text{(Eq. 4.20)}$$

where $W_{k,l,m} = w_{k,l,m}^{(N)}$ is the accumulated link weight for N data points. Afterwards, rule links to all the non-winning consequent labels (i.e., $\forall l \neq l*$) are discarded.

Stage 2: Rule Reduction. More input features are eliminated in this step using a rule consistency criterion adopted from rough set theory [205]. Specifically, for each i^{th} input feature, consistent rules are identified which, when their antecedent links to the i^{th}

input are omitted, will have the same remaining antecedent and consequent labels. That is, a rule R_k is consistent with another rule R'_k when it satisfies (Eq. 4.21)

$$\left(\mathbf{A}_k - \{A_{(i,j)_k}\} = \mathbf{A}_{k'} - \{A_{(i,j)_{k'}}\}\right) \wedge \left(\mathbf{C}_{(l,m)_k} = \mathbf{C}_{(l,m)_{k'}}\right) \quad \text{(Eq. 4.21)}$$

where $\mathbf{A}_k = \{A_{(1,j)_k}, \ldots, A_{(i,j)_k}, \ldots, A_{(I,j)_k}\}$ and $\mathbf{C}_k = \{C_{(l,1)_k}, \ldots, C_{(l,m)_k}, \ldots, C_{(l,M)_k}\}$ are the sets of all antecedent and consequent labels of R_k.

The i^{th} input feature is thus discarded when *all* rules R_k in memory meet criterion (Eq. 4.21). Consequently, duplicate rules with the same residual set of antecedent labels $\mathbf{A}_k - \{A_{(i,j)_k}\}$ are deleted. Otherwise, a partial feature reduction is performed for each rule R_k as long as removal of the i^{th} input of R_k does not lead to an ambiguous rule $R_{k'}$ i.e., $\mathbf{A}_k - \{A_{(i,j)_k}\} = \mathbf{A}_{k'} - \{A_{(i,j)_{k'}}\}$ but $\mathbf{C}_{(l,m)_k} \neq \mathbf{C}_{(l,m)_{k'}}$. Subsequently, any one or more consistent/duplicate rules matching R_k as per (Eq. 4.21) are discarded, and the inconsequential inputs of R_k are deleted. These steps ensure that each physical rule is stored only once, for memory efficiency, while enhancing the generalization ability of the rules. It must be noted that a different order of feature reduction may result in different final rule sets. A reasonable heuristic in this case is to perform reduction starting from the least significant input feature, selected last in the feature construction step.

The rough set-based rule reduction in RFCMAC bears some similarities to that in RSPOP [15], but they differ in several ways. RSPOP uses two separate phases of feature and rule reduction, whereas RFCMAC combines the two in a single phase. Also, the RFCMAC reduction is only concerned with the consistency criterion in (Eq. 4.21), whereas each RSPOP reduction phase additionally checks if certain objective measure deteriorates before reduction can take place. The latter, however, requires multiple passes of data that are rather slow and may degrade the conciseness of the final rule base. In contrast, RFCMAC tries to reduce as many rules as possible, only after which the rule base parameters are tuned. As shown later by the experiments in section 4.5.2-4.5.3, this method is able to derive a much more compact rule base, while still giving good output generalization. The RFCMAC parameter tuning procedure is described in section 4.3.3.

4.3.3 Parameter Tuning

The parameter tuning phase involves an iterative *Localized Least Mean Square* (LLMS) procedure, which is an iterative version of the LMS algorithm [287] adapted to the RFCMAC architecture for locally updating the kernel parameters of the antecedent and consequent labels selected in the current neighborhood. The procedure attempts to minimize the error criterion defined in (Eq. 4.22)

$$E^{(p)} = \frac{1}{2}\sum_{m=1}^{M}\left(t_m^{(p)} - y_m^{(p)}\right)^2 \tag{Eq. 4.22}$$

where $t_m^{(p)}$ and $y_m^{(p)}$ are the m^{th} target and system outputs for the p^{th} data point respectively. The LLMS-based tuning in RFCMAC involves successive corrections, one step at a time, to the kernel parameters of the selected antecedent label $A_{i,j}$ and consequent label $C_{l,m}$ in the negative direction of their gradients, as defined in (Eq. 4.23)-(Eq. 4.24)

$$\theta_{i,j}^{(p)} = \theta_{i,j}^{(p-1)} - \eta\frac{\partial E^{(p)}}{\partial \theta_{i,j}^{(p)}} \tag{Eq. 4.23}$$

$$\theta_{l,m}^{(p)} = \theta_{l,m}^{(p-1)} - \eta\frac{\partial E^{(p)}}{\partial \theta_{l,m}^{(p)}} \tag{Eq. 4.24}$$

where η is the learning rate and $\theta_{i,j}^{(p)}$ ($\theta_{l,m}^{(p)}$) is either the centroid $m_{i,j}$ ($m_{l,m}$) or width $\sigma_{i,j}$ ($\sigma_{l,m}$) of $A_{i,j}$ ($C_{l,m}$). A high η value produces fast learning at the expense of stability, whereas a low value yields stable but slow learning [61, 151]. To ensure a good compromise between learning speed and stability and to avoid introducing a new user (free) parameter to the system, η is empirically set as $1/N$ in this work (such that $0 < \eta \leq 1$).

Resolving (Eq. 4.24) based on the definitions in (Eq. 4.2)-(Eq. 4.4), the updating formulae for the consequent parameters $m_{l,m}$ and $\sigma_{l,m}$ are given in (Eq. 4.25)-(Eq. 4.26)

$$\Delta m_{i,j}^{(p)} = \frac{\eta}{M|\mathbf{S}|}\left(\frac{x_i^{(p)} - m_{i,j}^{(p)}}{(\sigma_{i,j}^{(p)})^2}\right)\sum_{m=1}^{M}\left(\frac{t_m^{(p)} - y_m^{(p)}}{D_m^{(p)}}\right)\sum_{k\in\mathbf{S}}\left(\frac{m_{(l,m)_k}^{(p)} - y_m^{(p)}}{\sigma_{(l,m)_k}^{(p)}}\right)\left(\frac{f_k^{(p)}}{(2-f_k^{(p)})^2}\right) \tag{Eq. 4.25}$$

$$\Delta \sigma_{i,j}^{(p)} = \frac{\eta}{M|\mathbf{S}|}\left(\frac{(x_i^{(p)} - m_{i,j}^{(p)})^2}{(\sigma_{i,j}^{(p)})^3}\right)\sum_{m=1}^{M}\left(\frac{t_m^{(p)} - y_m^{(p)}}{D_m^{(p)}}\right)\sum_{k\in\mathbf{S}}\left(\frac{m_{(l,m)_k}^{(p)} - y_m^{(p)}}{\sigma_{(l,m)_k}^{(p)}}\right)\left(\frac{f_k^{(p)}}{(2-f_k^{(p)})^2}\right) \tag{Eq. 4.26}$$

where $\Delta m_{i,j}^{(p)} = m_{i,j}^{(p+1)} - m_{i,j}^{(p)}$ and $\Delta \sigma_{i,j}^{(p+1)} = \sigma_{i,j}^{(p+1)} - \Delta\sigma_{i,j}^{(p)}$. Similarly, the updating

formulae for the antecedent parameters $m_{i,j}$ and $\sigma_{i,j}$ are given in (Eq. 4.27)-(Eq. 4.28)

$$\Delta m_{l,m}^{(p)} = \frac{\eta}{|\mathbf{S}_{l,m}^{(p)}|} \left(\frac{t_m^{(p)} - y_m^{(p)}}{D_m^{(p)} \sigma_{l,m}^{(p)}} \right) \sum_{k \in \mathbf{S}_{l,m}^{(p)}} \frac{f_k^{(p)}}{2 - f_k^{(p)}} \tag{Eq. 4.27}$$

$$\Delta \sigma_{l,m}^{(p)} = \frac{\eta}{|\mathbf{S}_{l,m}^{(p)}|} \left(\frac{\left(t_m^{(p)} - y_m^{(p)}\right)\left(y_m^{(p)} - m_{l,m}^{(p)}\right)}{D_m^{(p)} (\sigma_{l,m}^{(p)})^2} \right) \sum_{k \in \mathbf{S}_{l,m}^{(p)}} \frac{f_k^{(p)}}{2 - f_k^{(p)}} \tag{Eq. 4.28}$$

where $\Delta m_{l,m}^{(p)} = m_{l,m}^{(p+1)} - m_{l,m}^{(p)}$, $\Delta \sigma_{l,m}^{(p)} = \sigma_{l,m}^{(p+1)} - \sigma_{l,m}^{(p)}$, and $\mathbf{S}_{l,m}^{(p)}$ is the set of selected rule indices that are linked to $C_{l,m}$. The complete derivations of all the above updating equations can be found in Appendix B.

An important difference between the above updating procedure and that of contemporary FCMAC systems is that the former covers not only consequent labels but also antecedent labels, which help to further improve the accuracy of the input-output mapping. Also note that the shifting of antecedent centroid $m_{i,j}$ due to (Eq. 4.23) may result in an alteration of the activated input neighborhood set computed in (Eq. 4.5), when the same data point is presented to the system in the next training epoch. This may lead to unwanted spikes in the learning curve that can hamper the learning stability. To resolve this issue, a simple regularization procedure is done on the $m_{i,j}^{(p+1)}$ obtained via (Eq. 4.23), as per (Eq. 4.29)

$$m_{i,j}^{(p+1)} = \begin{cases} x_i^l & \text{if } m_{i,j}^{(p+1)} < x_i^l \wedge m_{i,j}^{(p+1)} \neq x_i^{min} \\ x_i^r & \text{if } m_{i,j}^{(p+1)} > x_i^r \wedge m_{i,j}^{(p+1)} \neq x_i^{max} \\ m_{i,j}^{(p)} & \text{otherwise} \end{cases} \tag{Eq. 4.29}$$

where x_i^l and x_i^r are the nearest input points on the left and right of centroid $m_{i,j}^{(p+1)}$, and $x_i^{min} = \min_{p \in \{1,\ldots,N\}} x_i^{(p)}$ and $x_i^{max} = \max_{p \in \{1,\ldots,N\}} x_i^{(p)}$ are the minimum and maximum values of the i^{th} input, respectively.

The above tuning process is repeated until the convergence criterion (Eq. 4.30) is met

$$\left| \sum_{p=1}^{N} E_{t-1}^{(p)} - \sum_{p=1}^{N} E_t^{(p)} \right| \leq \varepsilon \sum_{p=1}^{N} E_{t-1}^{(p)} \tag{Eq. 4.30}$$

where $E_t^{(p)}$ is the cost for the p^{th} data point at the t^{th} training epoch and ε is a small threshold value (e.g. 10^{-5}), or until a maximum number of epochs T_{max} is reached (i.e., $t = T_{max}$). The ε and T_{max} thus constitute complementary parameters to avoid excessive training epochs that may otherwise yield overfitting and degraded generalization.

Table 4.1: Complexity analysis of the RFCMAC learning procedure

Learning phase	Time complexity	Description
Label generation		
1) *Feature construction*	$O(N \times D \times I \times M)$	Compute pairwise input-output and input-input feature correlations, and add selected features one at a time
2) *Label construction*	$O(N \times (\sum_{i=1}^{D} J_i + \sum_{m=1}^{M} L_m))$	For each data point, update Hebbian strengths of the winning input/output labels, or create new labels
3) *Label reduction*	$O(\sum_{i=1}^{D} J_i + \sum_{m=1}^{M} L_m)$	Combine the labels in all input and output dimensions when their overlapping degrees are sufficiently high
Rule generation		
1) *Rule construction*	$O(N \times (\sum_{i=1}^{D} J_i + \sum_{m=1}^{M} L_m))$	For each point, determine based on the winning input labels to create a rule linking them to output labels
2) *Rule reduction*	$O(K^2 \times (D + M))$	Perform pairwise rule comparison to check whether they are consistent, and then remove duplicate rules
Parameter tuning	$O(N \times K \times T_{max} \times (D + M))$	Update kernel parameters of the labels belonging to the selected rules for all data points in every epoch

4.3.4 Complexity Analysis

A summary of the worst time complexity of the three learning phases, employing the notations adopted in the previous sections, is given in Table 4.1. It can be seen that the main computational load lies in the feature construction and parameter tuning steps, especially when the number of inputs I and number of data points N are large. Nevertheless, the complexity of the feature construction phase remains fairly low, as it is governed by the parameter D that is usually much smaller than I (i.e., $D \ll I$). On the other hand, as the parameter tuning in RFCMAC is localized and only updates selected labels in the current neighborhood, the complexity of the step is relatively low compared to globalized learning methods. This is also notably faster than conventional FCMAC methods, such as [191, 110, 241, 190, 188, 270], since the tuning is done at a much reduced feature/rule space after the label generation and rule generation phases are performed.

Meanwhile, the space complexity of the proposed system is defined in (Eq. 4.31), which is essentially the sum of the numbers of input/output labels and (physical) rules.

$$O\left(K + \sum_{i=1}^{D} J_i + \sum_{m=1}^{M} L_m\right) \qquad \text{(Eq. 4.31)}$$

As noted in section 4.3.2, it must be that $K \leq N$, though most of the time it is much

	x_1	x_2	x_3	x_4	x_5	y
$p=1$	5	0	1	2	3	1
$p=2$	5	0	1	3	4	1
$p=3$	5	1	1	2	3	2
$p=4$	5	1	2	2	3	2
$p=5$	5	1	1	3	3	2
$p=6$	5	0	2	2	4	2

Figure 4.2: A simple dataset with redundant input features

smaller (i.e., $K \ll N$), implying in turn a total (worst) space complexity as per (Eq. 4.32).

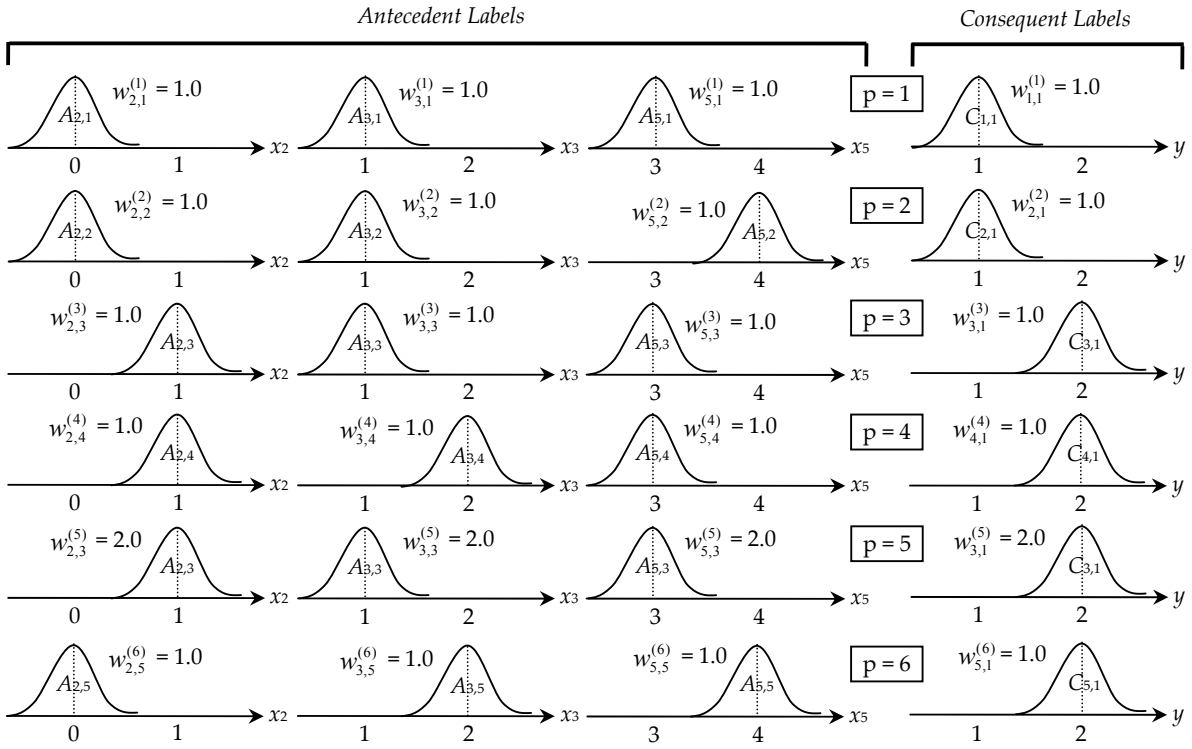
$$O\left(N + \sum_{i=1}^{D} J_i + \sum_{m=1}^{M} L_m\right) \quad \text{(Eq. 4.32)}$$

This requirement is considerably low nonetheless, which again can be attributed to the reduction processes in the label generation and rule generation phases. This benefit is appealing when compared to classical FCMAC systems.

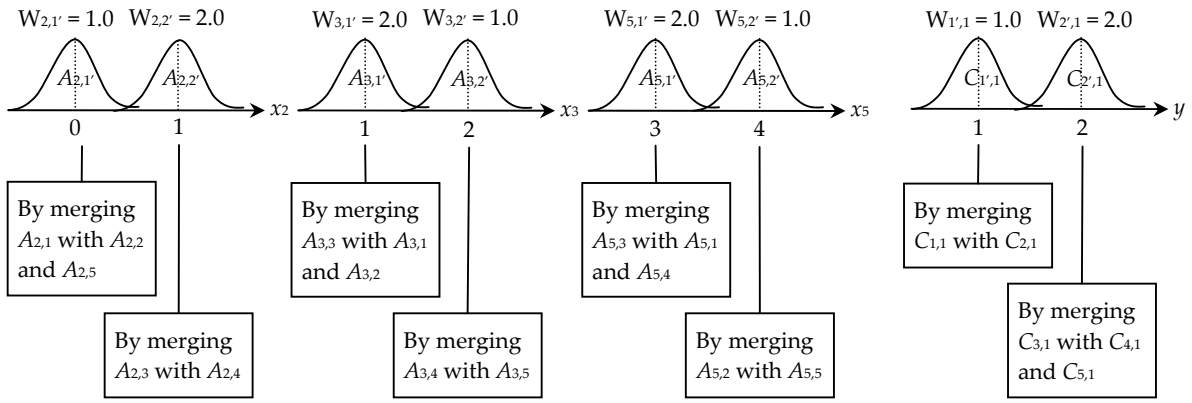
4.4 Pedagogical Example

A pedagogical example is given to illustrate the working of the RFCMAC learning procedure. Consider the simple data in Figure 4.2, comprising five nominal input features x_1-x_5 and an output y. In this case, x_2 and x_3 are the two features relevant to y, whereas x_1, x_4 and x_5 are irrelevant. There also exist three indispensable rules that best describe the data, and it is expected for RFCMAC to discover them. For visualization purposes, the parameter D is set as three here. Learning begins with the *feature construction* step selecting x_2, x_3 and x_5 in this order based on (Eq. 4.6) (x_2 is the first rank and so on).

The subsequent *label construction* step involves determining whether a new fuzzy label needs to be created for each p^{th} data point, as illustrated in Figure 4.3(a). For simplicity, the figure only shows either the newly generated, or existing (winning), labels, having the lowest dissimilarities $d_{i,j}^{(p)}$ (or $d_{l,m}^{(p)}$) and Hebbian weights $w_{i,j}^{(p)}$ (or $w_{l,m}^{(p)}$) for $p \in \{1, ..., 6\}$. Here the splitting threshold δ is set as 0.5, and the initial width of the Gaussian MF set to a small value so as to minimize the overlaps between neighboring MFs. For $p = 1$ to 4 and 6, a new label is created at every input/output dimension, as the overall similarity $S^{(p)}$ defined in (Eq. 4.10) falls below δ; conversely, no new label is formed for $p = 5$, since the 5^{th} data point is the same as the 3^{rd} (i.e., $S^{(5)} = 1$), viewed from x_2, x_3 and x_5. In



(a) Label construction (Stage 1B)

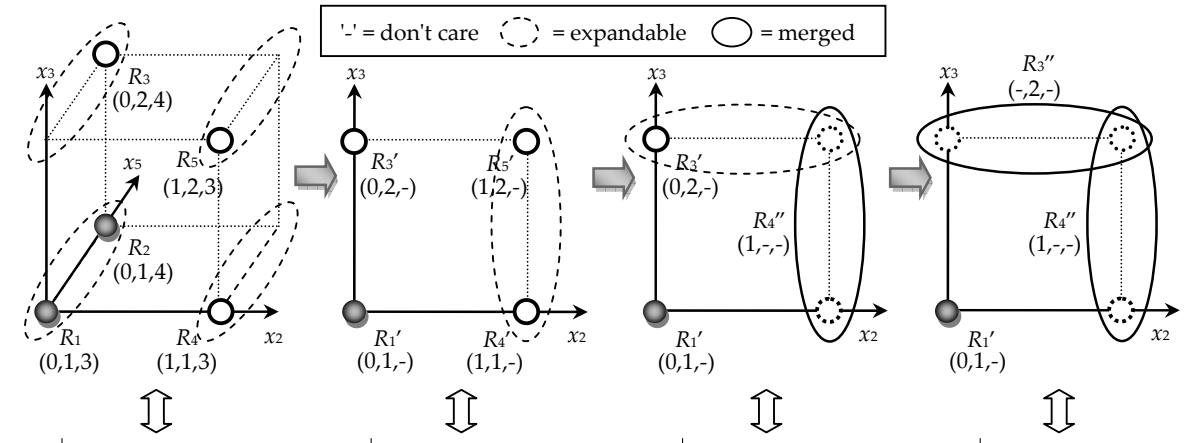


(b) Label reduction (Stage 2)

Figure 4.3: Example of label generation in RFCMAC

the first case, duplicate (or very similar) input and output labels may emerge (e.g. $A_{2,1}$ and $A_{2,2}$, $C_{1,1}$ and $C_{2,1}$, etc) that have identical (or very similar) MFs. This is possible mainly because the computation of $S^{(p)}$ involves all, rather than one, input dimensions.

Label reduction is carried out afterwards to merge similar labels, as per (Eq. 4.13)-(Eq. 4.14), taking into account the weights $W_{i,j} = w^{(6)}_{i,j}$ and $W_{l,m} = w^{(6)}_{l,m}$ accumulated from $p = 1$ to 6. Figure 4.3(b) illustrates this step, where the callouts show how the final labels $A_{i,j'}$ and $C_{l',m}$ are derived. For instance, the antecedent label $A_{2,3}$ has the same MF as $A_{2,4}$, so they can be merged into a new label $A_{2,2'}$. Here $A_{2,3}$ has a larger accumulated



	x_2	x_3	x_5	y
R_1	0	1	3	1
R_2	0	1	4	1
R_3	0	2	4	2
R_4	1	1	3	2
R_5	1	2	3	2

(a) Original rules (Stage 1)

	x_2	x_3	x_5	y
R_1'	0	1	-	1
R_3'	0	2	-	2
R_4'	1	1	-	2
R_5'	1	2	-	2

(b) x_5 reduction (Stage 2)

	x_2	x_3	x_5	y
R_1'	0	1	-	1
R_3'	0	2	-	2
R_4"	1	-	-	2

(c) x_3 reduction (Stage 2)

	x_2	x_3	x_5	y
R_1'	0	1	-	1
R_3"	-	2	-	2
R_4"	1	-	-	2

(d) x_2 reduction (Stage 2)

Figure 4.4: Example of rule generation in RFCMAC

weight $W_{2,3} = 2.0$, for the 5^{th} data point equals the 3^{rd}, as explained before. Thus, $A_{2,2'}$ will have a weight $W_{2,2'} = 2.0$. Meanwhile, for any other labels having identical MF and accumulated weight, the resulting merged label will have exactly the same parameters, e.g. the merged label $A_{2,1'}$ is identical to its constituents $A_{2,1}$, $A_{2,2}$ and $A_{2,5}$.

Next, the *rule construction* step is done via (Eq. 4.19)-(Eq. 4.20), yielding five (physical) rules R_1-R_5 in this order, as shown in Figure 4.4(a). The dark and white circles denote the rules for $y = 1$ and 2 respectively, while the (x, y, z) labels the centroid coordinates of their respective antecedent labels. *Rule reduction* is then carried out that starts by examining x_5 (i.e., the lowest-ranked input feature). It is found that x_5 can be removed entirely, for all rules meet (Eq. 4.21). The RFCMAC memory representation thus reduces from three- to two-dimensional, as per Figure 4.4(b). R_3-R_5 expand their territories to uncovered memory region giving R_3'-R_5', while R_1 and R_2 are merged into R_1' since they are consistent. These improve the rule generalization. The next consistency test using (Eq. 4.21) shows that removing links to x_3 from rules R_4' and R_5' does not yield ambiguity; so, they are merged into R_4", as per Figure 4.4(c). For the same consistency reason, it is also valid to remove link to x_2 from R_3', yielding R_3". This gives the three

Table 4.2: Parameter configurations of RFCMAC-Yager for the experiments

Parameter	Nakanishi Data			Water Plant Data		Leukemia Data	
	Nonlinear	Chemical	Stock	2-Class	3-Class	Independent Test	Leave-One-Out
Maximum inputs D	2	1	2	5	6	3	4
Split threshold δ	0.7	0.4	0.2	0.7	0.3	0.3	0.3
Error threshold ε	10^{-5}	10^{-5}	6×10^{-3}	10^{-4}	10^{-4}	10^{-3}	10^{-5}

final rules: R_1', R_3", R_4", as in Figure 4.4(d). Parameter tuning is done afterwards, which stops after one epoch as the error (Eq. 4.22) is already zero, also satisfying (Eq. 4.30).

4.5 Experimental Studies

4.5.1 Simulation Setup

A number of experimental studies have been conducted to validate the RFCMAC-Yager system, the three most illustrative of which are reported in this chapter: the *Nakanishi regression benchmark* [182], *wastewater treatment plant* [20], and *acute leukemia diagnosis* [92]. These experimental validations serve to demonstrate the efficacy of the proposed model as an implementation prototype for the long-term memory modules in INCA as well as in dealing with complex task domains. In all simulations, the merging threshold α and maximum training epochs T_{max} are fixed as 0.5 and 15,000 respectively, while the other parameters are chosen empirically for each case study, as per Table 4.2. Detailed results and analysis of the case studies are given in sections 4.5.2-4.5.4. The corresponding estimates of learning time required by RFCMAC-Yager can be found in Appendix C.

4.5.2 Nakanishi Regression Benchmark

The Nakanishi datasets [182] are explored here to provide an initial evaluation for the function approximation and knowledge reduction abilities of the RFCMAC system. Three datasets are available: the *nonlinear system*, *human operation of a chemical plant*, and *daily price in a stock market*, each split into 3 similar groups: A, B and C [182]. In this study, A and B are used as the training set, while C the testing set. The results are then compared with those of several regression methods: Multilayer Perceptron (MLP) [225], Radial Basis Function (RBF) [213], Support Vector Regression (SVR) [244], Interval-Valued Compositional Rule of Inference (IV-CRI) [182], Adaptive-Network-based Fuzzy

Table 4.3: Fuzzy rules identified by RFCMAC-Yager for Nakanishi datasets

(a) Nonlinear system example

	x_1	x_2	y
R_1	*High*	*High*	*Low*
R_2	*High*	*Low*	*Medium*
R_3	*Low*	-	*High*

(b) Chemical plant example

	x_3	y
R_1	*Very Low*	*Very Low*
R_2	*Low*	*Low*
R_3	*Rather Low*	*Rather Low*
R_4	*Rather High*	*Rather High*
R_5	*High*	*High*
R_6	*Very High*	*Very High*

(c) Stock market example

	x_2	x_4	y
R_1	-	*Very High*	*Very Low*
R_2	-	*High*	*Very Low*
R_3	*High*	*Medium*	*Low*
R_4	*Low*	*Medium*	*High*
R_5	-	*Low*	*High*
R_6	*Very Low*	*Medium*	*High*
R_7	-	*Very Low*	*Very High*
R_8	*Very High*	*Medium*	*Very High*

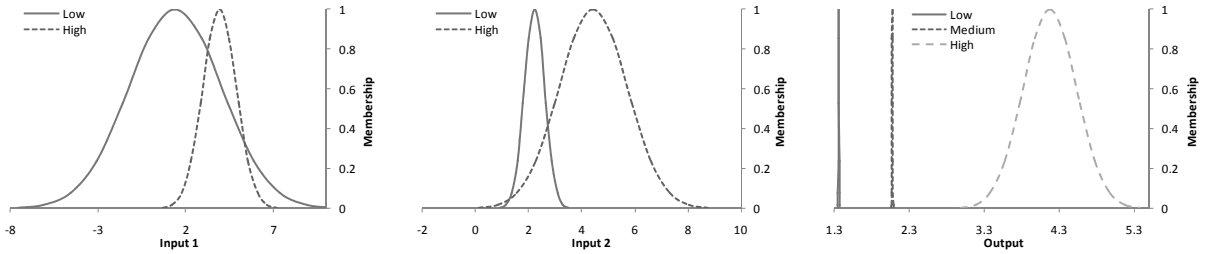
Inference System (ANFIS) [112], Evolving Fuzzy Neural Network (EuFNN) [123], RSPOP realizing CRI (RSPOP-CRI) [15], HRR [152], and FCMAC-Yager [241].

The nonlinear system in [182] is described using (Eq. 4.33)

$$y = \left(1 + x_1^{-2} + x_2^{-1.5}\right)^2, \quad 1 \leq x_1, x_2 \leq 5 \qquad \text{(Eq. 4.33)}$$

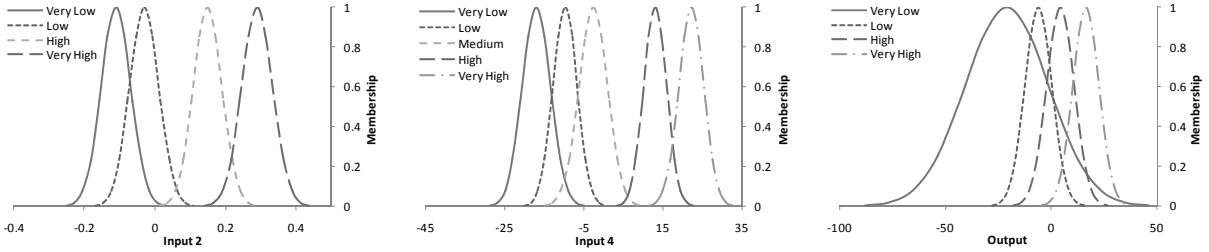
which involves two input features x_1, x_2, and an output feature y. Meanwhile, the actual data used in this study includes two redundant, noisy inputs x_3 and x_4. Comparisons can be accordingly made among the systems listed which employs feature selection methods i.e., IV-CRI, RSPOP-CRI, HRR, and RFCMAC-Yager. All these systems select inputs x_1 and x_2 as desired, except for RSPOP-CRI which still keeps x_4 (albeit partially). The membership functions identified by RFCMAC-Yager are shown in Figure 4.5(a). As can be seen, two antecedent labels are generated in x_1 and x_2, while y has three consequent labels. The RFCMAC-Yager system generates a total number of three rules, listed in Table 4.3(a), which accurately capture the inverse relationship between the inputs and output in (Eq. 4.33). For instance, rule R_1 states "IF x_1 is *High* AND x_2 is *High* THEN y is *Low*.", and R_3 states "IF x_1 is *Low* THEN y is *High*." In the latter, the omission of links to x_2 stems from the partial feature reduction performed in the rule reduction step.

The chemical plant dataset in [182] contains five inputs and a single output, labeled x_1, x_2, x_3, x_4, x_5 and y respectively. In this example, the feature selection in IV-CRI [182] omits x_2, x_4 and x_5, while RSPOP-CRI and HRR delete x_1, x_2, x_5 and x_1, x_2, x_4, x_5 respectively. RFCMAC-Yager achieves the same result as HRR (i.e., retaining only x_3), and crafts six antecedent and consequent labels in both input x_3 and output y



(a) Nonlinear system example

(b) Chemical plant example

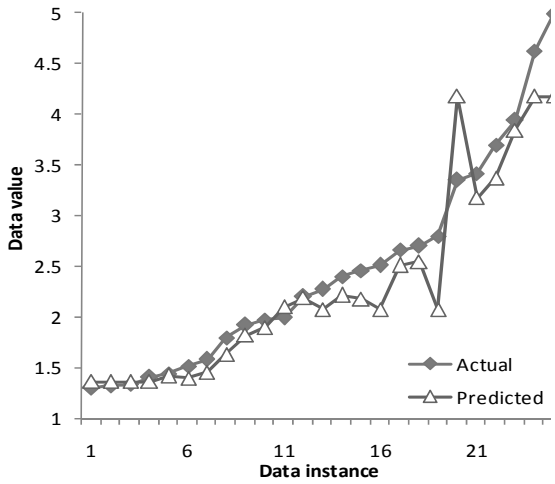


(c) Stock market example

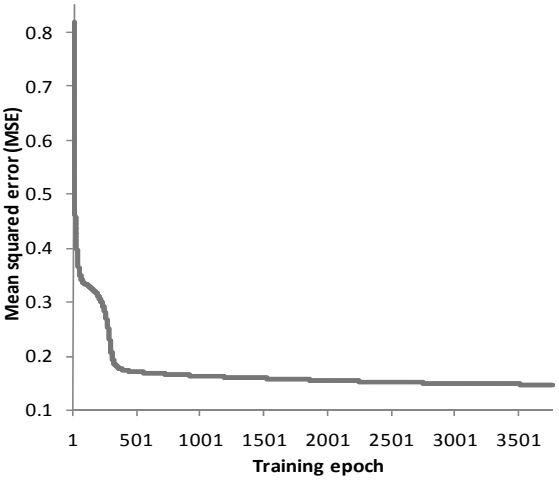
Figure 4.5: Fuzzy labels crafted by RFCMAC-Yager for Nakanishi datasets

respectively, as per Figure 4.5(b). It subsequently generates six rules, as listed in Table 4.3(b). The monotonic mapping of the rules in the table essentially suggests that x_3 and y are approximately linearly correlated to one another.

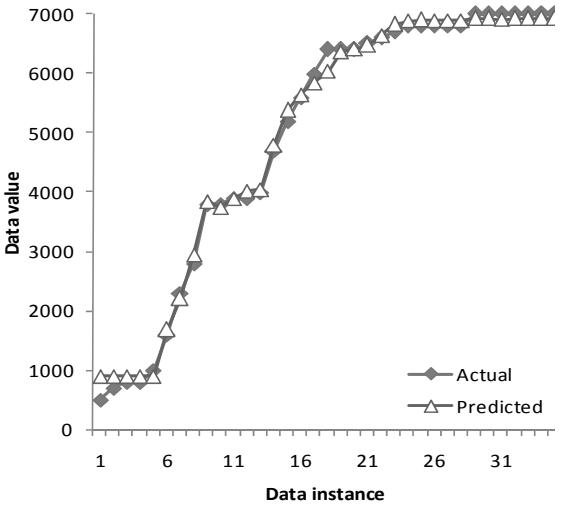
The stock market price prediction dataset consists of ten inputs and one output, termed x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10} and y respectively, as described in [182]. The feature selection procedures used in IV-CRI and RSPOP-CRI preserve inputs x_4, x_5, x_8 and x_4, x_5, x_7, x_8, x_9 respectively, while HRR omits x_2 and x_5. Meanwhile, RFCMAC-Yager keeps only two inputs: x_2 and x_4, which is smaller than the other methods. Input x_2 is the one discarded by the other methods. However, RFCMAC-Yager shows that x_2 is in fact informative, as reflected later by its relatively good approximation results. Five antecedent labels are then generated in both x_2 and x_4, and four consequent labels are crafted in y, as per Figure 4.5(c). Table 4.3(c) lists in turn the eight rules identified in this example, which are again very intuitive and concise.



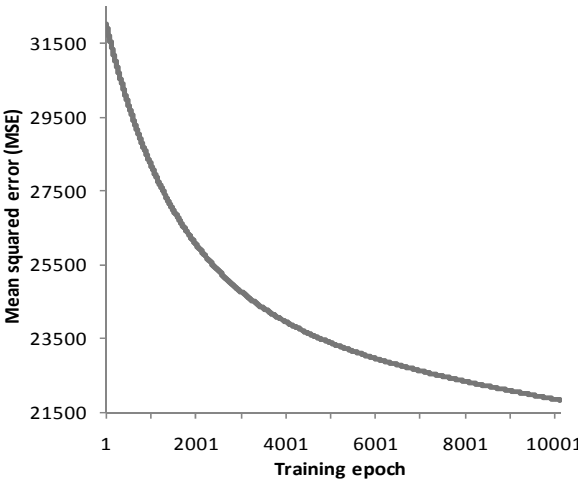
(a) Nonlinear system (System prediction)



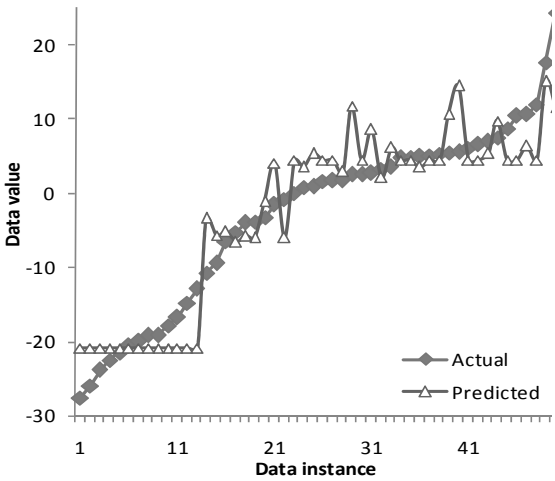
(b) Nonlinear system (Learning convergence)



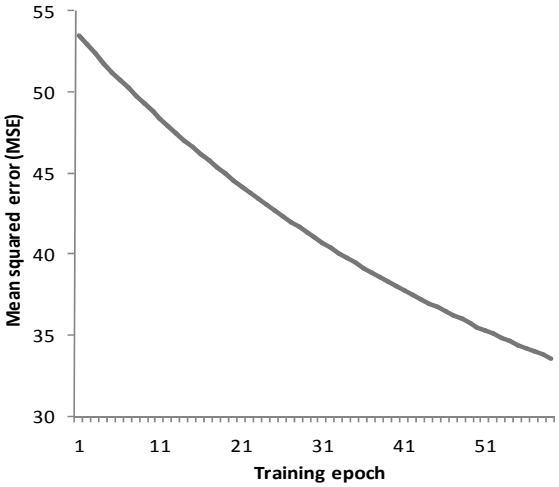
(c) Chemical plant (System prediction)



(d) Chemical plant (Learning convergence)



(e) Stock market (System prediction)



(f) Stock market (Learning convergence)

Figure 4.6: Results of RFCMAC-Yager on Nakanishi datasets

Traces of the system prediction and training convergence for all three examples are given in Figure 4.6(a), (c), (e) and (b), (d), (f) respectively. The former shows generalization performance on unseen testing data while the latter shows a stable, non-increasing learning curve that warrants training convergence (corresponding chiefly to the parameter tuning phase described in section 4.3.3). Consolidated simulation results are shown in Table 4.4, involving four metrics: the number of selected features, number of rules, Pearson's correlation R [90], and test mean square error (MSE). As seen, RFCMAC-Yager gives the smallest set of features for all examples. Its number of rules is also the smallest for the first example, second smallest for the second example (excluding IV-CRI as its rules are coded by hand), and smallest for the last example (excluding IV-CRI again). On the other hand, RFCMAC-Yager yields the best R and MSE for the first two examples, and second best for the last. While HRR gives a better result in the latter case, it has more features and rules, and hence less interpretability. In addition, thanks to its reduction mechanisms, RFCMAC-Yager compares favorably to its predecessor FCMAC-Yager in terms of memory requirements and prediction performances. In sum, these results show the salient interpretability and generalization traits of the proposed system.

4.5.3 Water Plant Monitoring

This experiment concerns a case study of wastewater treatment plant data [20], conducted to evaluate the classification performance of the RFCMAC-Yager in an ill-structured domain with missing feature values and unbalanced class distribution. The task involves a historical plant dataset collected over 527 days (i.e., samples) with one series of measurements per day. There are 38 different real-valued input features observed daily: 9 features for representing plant inputs, 6 and 7 features for inputs to the primary and secondary settlers respectively, 7 features for plant outputs, and 9 features for plant performances. Each day is categorized into one of the 13 classes associated with the status of the plant, some indicating normal operation of varying types, others denoting faults at various parts of the plant. The detailed feature information can be found in [20], while the distribution of the classes is presented in Table 4.5.

However, as all faults in the actual plant occur for a brief period (only 1-4 days) and

Table 4.4: Benchmark results on Nakanishi datasets

Predictor	Nonlinear System Example				Chemical Plant Example				Stock Market Example			
	SF	#Rules	R	MSE	SF	#Rules	R	MSE	SF	#Rules	R	MSE
MLP	All	-	0.874	0.396	All	-	0.999	1.703×10^4	All	-	0.696	143.015
RBF (Gaussian)	All	-	0.857	0.403	All	-	0.921	8.803×10^5	All	-	0.881	33.420
SVR	All	-	0.832	0.363	All	-	0.998	1.833×10^4	All	-	0.816	49.438
IV-CRI	x_1, x_2	6+	0.609	0.706	x_1, x_3	5+	0.993	2.581×10^5	x_4, x_5, x_8	4+	0.661	93.020
ANFIS	All	*	0.853	0.286	All	*	0.780	2.968×10^6	All	*	0.875	38.062
EFuNN	All	*	0.720	0.566	All	*	0.946	7.247×10^5	All	*	0.756	72.542
RSPOP-CRI	x_1, x_2, x_4	17	0.856	0.383	x_3, x_4	14	0.983	2.124×10^5	x_4, x_5, x_7-x_9	29	0.922	24.859
HRR	x_1, x_2	7	0.911	0.185	x_3	3	0.998	2.423×10^4	x_1, x_3, x_4, x_6-x_{10}	20	0.947	15.128
FCMAC-Yager	All	7	0.557	0.827	All	19	0.927	7.856×10^5	All	42	0.652	167.003
RFCMAC-Yager	x_1, x_2	3	0.957	0.109	x_3	6	0.999	1.656×10^4	x_2, x_4	8	0.931	19.597

SF = selected features, R = Pearson's correlation, MSE = mean squared error, + = manually set, - = not applicable, * = not specified

Table 4.5: Distribution of output classes in water plant dataset

Original Dataset		2-Class Dataset		3-Class Dataset		#Samples
Category	Value	Category	Value	Category	Value	
Normal situation	1	Normal	1	OK	1	275
Secondary settler problems, type 1	2	Faulty	2	Faulty	3	1
Secondary settler problems, type 2	3	Faulty	2	Faulty	3	1
Secondary settler problems, type 3	4	Faulty	2	Faulty	3	4
Normal situation with good performance	5	Normal	1	Good	2	116
Solids overload, type 1	6	Faulty	2	Faulty	3	3
Secondary settler problems, type 4	7	Faulty	2	Faulty	3	1
Storm, type 1	8	Faulty	2	Faulty	3	1
Normal situation, low influent	9	Normal	1	OK	1	69
Storm, type 2	10	Faulty	2	Faulty	3	1
Normal situation	11	Normal	1	OK	1	53
Storm, type 3	12	Faulty	2	Faulty	3	1
Solids overload, type 2	13	Faulty	2	Faulty	3	1

are resolved immediately, one main issue is the lack of training samples for the fault cases. Following [239], therefore, the fault cases were collated to form two (i.e., *Normal* and *Faulty*) and three (i.e., *OK*, *Good* and *Faulty*) broader categories, also given in Table 4.5. This results in two datasets, labeled 2-class and 3-class respectively. Additionally, in order to handle the missing feature values, *class mean imputation* method is used [118]. That is, the data are grouped based on the output classes, and in each group, the mean value of every feature is computed to fill in the missing values in that feature. It is not advisable to simply discard/rule out samples with missing values here, due to the presence of instrumental failures or other occasions disrupting the measurement.

Evaluation of the RFCMAC-Yager is subsequently done using a *stratified* 3-fold cross-validation (CV) procedure, i.e., partitioning the data into 3 separate groups of train and test sets, each retaining the output class proportion as in the original data. Figure 4.7 shows in turn the fuzzy labels crafted by RFCMAC-Yager for the 2-class data in CV1. In this, the system finds that the status of the plant depends solely on 3 features: *suspended solids to primary settler* (SS-P), *biological demand of oxygen to secondary settler* (RD-DBO-S), and *sediments* (RS-SED-G). Meanwhile, the rule set pertaining to the semantic interpretations of the labels in Figure 4.7 is given in Table 4.6. As can be seen, the rules are short, comprehensible, and fairly rational. For instance, rules R_2, R_3 and R_4 indicate that a fault is expected if either the SS-P level is high or RD-DBO-S level is low

Table 4.6: Fuzzy rules of RFCMAC-Yager for 2-class water dataset (CV1)

	SS-P	RD-DBO-S	RS-SED-G	Class
R_1	*Low*	*High*	*High*	*Normal*
R_2	-	*Low*	-	*Faulty*
R_3	*High*	-	-	*Faulty*
R_4	-	-	*Low*	*Faulty*

or RS-SED-G level is low; otherwise, a normal operation is assumed, as per R_1.

The results of RFCMAC-Yager on the 2-class data are summarized in Figure 4.8(a)-(b). Figure 4.8(a) shows the learning curve and convergence traits of the system (in CV1), while Figure 4.8(b) presents a *receiver operating characteristic* (ROC) plot that showcases the robustness and discriminative power of the system on the test data throughout all three CVs. The latter is obtained by varying certain decision threshold and measuring the *specificity* and *sensitivity* rates for every threshold value [48]. In the current context, specificity and sensitivity are the proportion of normal and faulty cases that are correctly classified respectively, as per (Eq. 4.34)-(Eq. 4.35). Larger area under the ROC curve indicates better performance [66]. Also, EER refers to the *error equal rate* when specificity equals sensitivity. Another related criterion (not shown in the plot but used in later benchmarking) is *precision* i.e., the proportion of predicted faulty cases that are actually correct, as in (Eq. 4.36). These metrics serve to complement the system's accuracy evaluation in relation to the unbalanced class distribution in the water data.

$$\text{Specificity} = \frac{\#\text{normal samples correctly classified}}{\#\text{actual normal samples}} \tag{Eq. 4.34}$$

$$\text{Sensitivity} = \frac{\#\text{fault samples correctly classified}}{\#\text{actual faulty samples}} \tag{Eq. 4.35}$$

$$\text{Precision} = \frac{\#\text{faulty samples correctly classified}}{\#\text{samples classified as faulty}} \tag{Eq. 4.36}$$

Comparisons are then made with several other methods: Fuzzy Rough Feature Selection (FRFS) [239], Naïve Bayes [52], MLP [225], RBF [213], SVM [211], C4.5 decision tree [216], k-Nearest Neighbors (k-NN) [44], RSPOP-CRI [15], the Generic Self-organizing Fuzzy Neural Network realizing Yager inference (GenSoFNN-Yager) [195], and FCMAC-Yager [241]. Table 4.7 shows the benchmark results using the accuracy, sensitivity, specificity and precision metrics described before, plus the average numbers of features selected

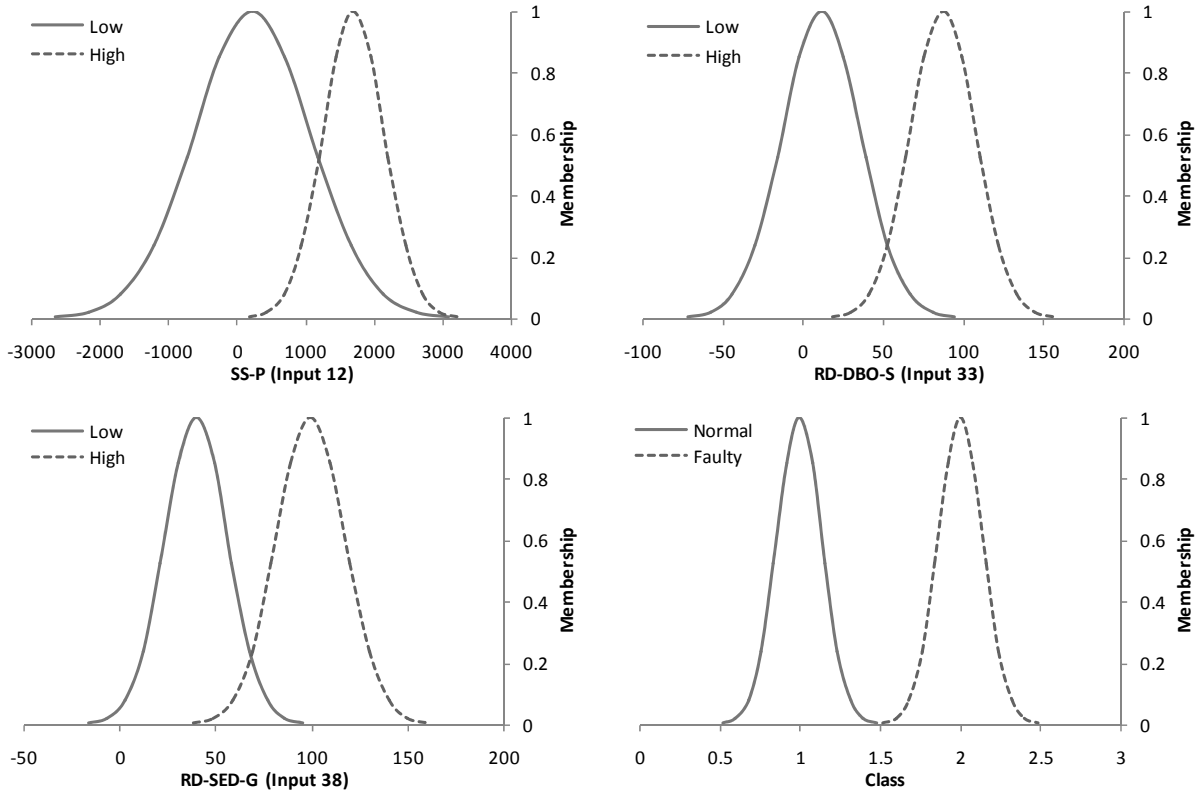
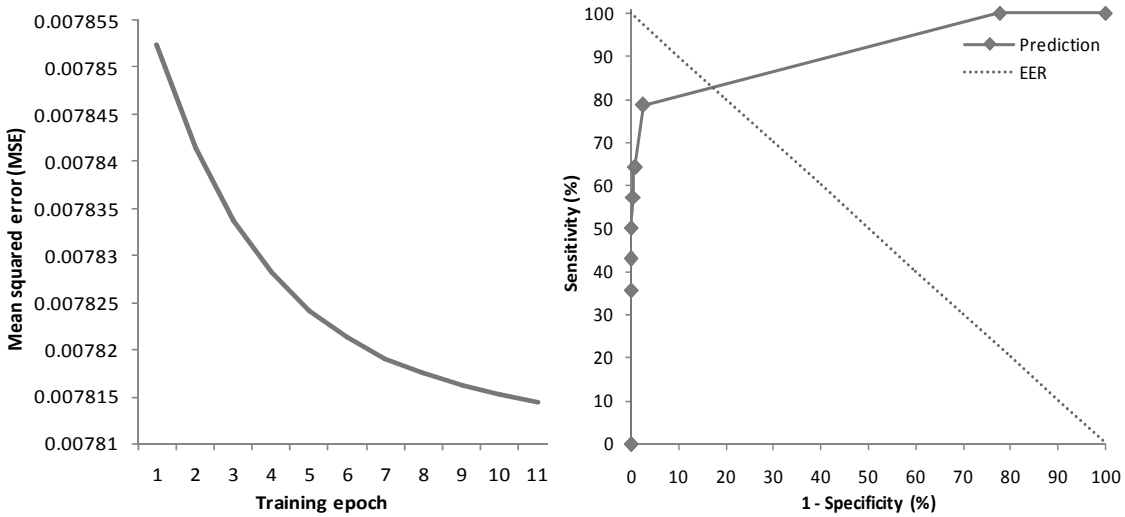


Figure 4.7: Fuzzy labels crafted by RFCMAC-Yager for 2-class water dataset (CV1)



(a) Learning convergence (CV1) (b) ROC curve

Figure 4.8: Results of RFCMAC-Yager on 2-class water dataset

and rules crafted. RFCMAC-Yager gives excellent overall performances, despite having rather low sensitivity due to the unbalanced class distribution (i.e., the faulty cases are fewer than normal ones). It also produces lesser features and rules than the other rule-based methods (excluding FRFS whose number of rules is predetermined), exemplifying

Table 4.7: Benchmark results on 2-class water dataset

Classifier	Evaluation	#Features	#Rules	Accuracy	Sensitivity	Specificity	Precision
FRFS	75%t-25%e	10.00	2+	83.90%	*	*	*
Naïve Bayes	3-fold CV	All	-	98.67%	85.71%	99.03%	70.59%
MLP (20 neurons)	3-fold CV	All	-	98.29%	42.86%	99.81%	85.71%
RBF (5 Gaussians)	3-fold CV	All	-	98.10%	57.14%	99.22%	66.67%
SVM (c = 1.0)	3-fold CV	All	-	98.48%	42.86%	100.00%	100.00%
C4.5 (p = 0.25)	3-fold CV	4.00	5.00	98.29%	64.29%	99.22%	69.23%
k-NN (k = 3)	3-fold CV	All	-	98.48%	42.86%	100.00%	100.00%
RSPOP-CRI	3-fold CV	24.00	241.33	98.29%	42.86%	99.81%	85.71%
GenSoFNN-Yager	3-fold CV	All	9.33	98.67%	85.71%	99.03%	70.59%
FCMAC-Yager	3-fold CV	All	64.33	97.53%	14.28%	99.81%	66.67%
RFCMAC-Yager	3-fold CV	2.67	3.67	98.67%	50.00%	100.00%	100.00%

75%t-25%e = 75% train-25% test, CV = cross validation, - = not applicable, * = not specified, + = manually set

Table 4.8: Benchmark results on 3-class water dataset

Classifier	Evaluation	#Features	#Rules	Accuracy	FR-OK	FR-Good	FR-Faulty
FRFS	75%t-25%e	11.00	3+	71.80%	*	*	*
Naïve Bayes	3-fold CV	All	-	85.20%	9.82%	30.17%	28.57%
MLP (20 neurons)	3-fold CV	All	-	85.58%	7.30%	28.45%	100.00%
RBF (5 Gaussians)	3-fold CV	All	-	84.25%	9.82%	32.76%	42.86%
SVM (c = 1.0)	3-fold CV	All	-	85.39%	5.29%	42.24%	50.00%
C4.5 (p = 0.25)	3-fold CV	19.00	31.00	79.89%	9.57%	54.31%	35.71%
k-NN (k = 3)	3-fold CV	All	-	82.35%	6.80%	50.86%	50.00%
RSPOP-CRI	3-fold CV	35.33	348.67	74.19%	20.65%	41.38%	42.86%
GenSoFNN-Yager	3-fold CV	All	323.33	79.51%	5.79%	61.21%	100.00%
FCMAC-Yager	3-fold CV	All	204.67	69.07%	12.59%	85.34%	100.00%
RFCMAC-Yager	3-fold CV	5.67	34.67	86.34%	7.81%	25.86%	78.57%

75%t-25%e = 75% train-25% test, CV = cross validation, - = not applicable, * = not specified, + = manually set

its superior interpretability. In this, the RFCMAC feature construction and rule reduction steps are shown to be much more effective than the RSPOP feature/rule reduction (that is subject to some objective criterion deterioration, as explained before). It is also evident that the reduction mechanisms in RFCMAC-Yager helps to achieve significant result improvements as compared to its predecessor FCMAC-Yager.

Further comparisons using the same classifiers are performed on the 3-class dataset, as given in Table 4.8. This task poses a greater challenge, as reflected in the lower resultant prediction performances. Subsequently, because the current problem involves more than two classes, *false rate* (FR) indicator is used in place of the sensitivity, specificity and precision criteria. Specifically, the notation FR-c refers to the number of class-c samples

Table 4.9: Fuzzy rules of RFCMAC-Yager for leukemia dataset

	LTC4S	Zyxin	Class
R_1	*Low*	*Low*	*ALL*
R_2	-	*High*	*AML*
R_3	*High*	-	*AML*

that are predicted wrongly, divided by the total number of class-c cases. As seen in Table 4.8, RFCMAC-Yager again displays its predictive competency, particularly in terms of accuracy and FR for class Good. In comparison to the other rule-based methods (excluding FRFS), it is clear that the RFCMAC structural reduction approach offers a major advantage. While C4.5 gives fewer rules in this case (albeit quite marginal), its knowledge base involves a larger set of features and hence poorer overall rule readability.

4.5.4 Acute Leukemia Diagnosis

To examine the ability of RFCMAC-Yager in handling high-dimensional data, an experiment is conducted using the acute leukemia microarray data [92], popularly studied in bioinformatics. The dataset contains expression profiles of 7129 genes (probe sets) collected from 72 leukemia patients, where the expression levels of each sample is measured by the Affymetrix oligonucleotide microarray. Out of these patients, 47 were identified as having *acute lymphoblastic leukemia* (ALL) and the other 25 as *acute myeloid leukemia* (AML). Following the setup in [92], the data is split into a train set of 38 samples (27 ALL and 11 AML), and an independent test set of 34 samples (20 ALL and 14 AML).

Given the ultra-high dimensionality of the dataset, feature selection and in particular the RFCMAC feature construction step provide a natural means to remove redundant or inconsequential input features (that may contribute significantly to the classification errors), and to reduce the overall computational time and memory requirements. In this experiment, the feature construction step initially selects 3 genes (i.e., $D = 3$): *Leukotriene C4 synthase* (Probe U50136), *Zyxin* (Probe X95735), and *FAH Fumarylacetoacetate* (Probe M55150), which were also selected and discussed in [92]. However, subsequent consistency evaluations in the RFCMAC rule reduction step find that the last feature (gene) can be removed entirely, leaving only the first two. The fuzzy labels

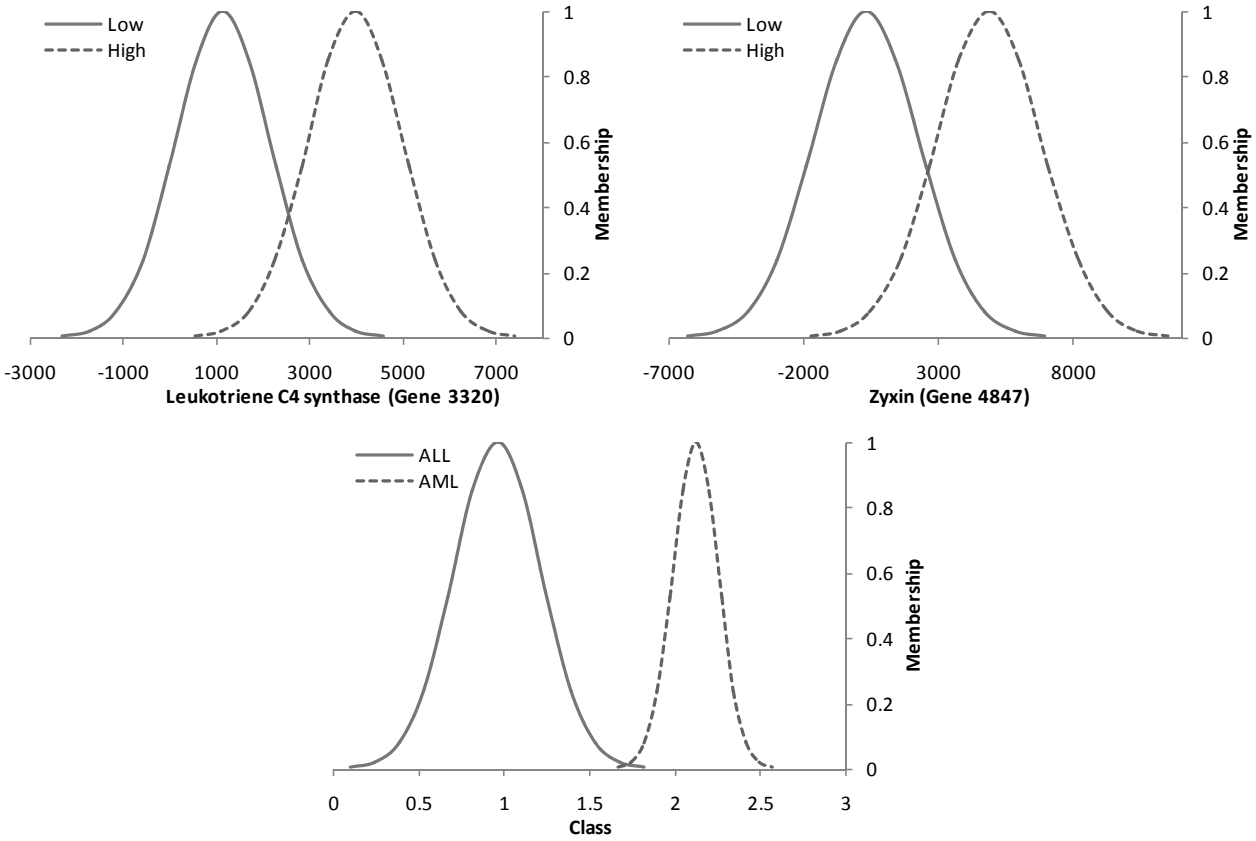
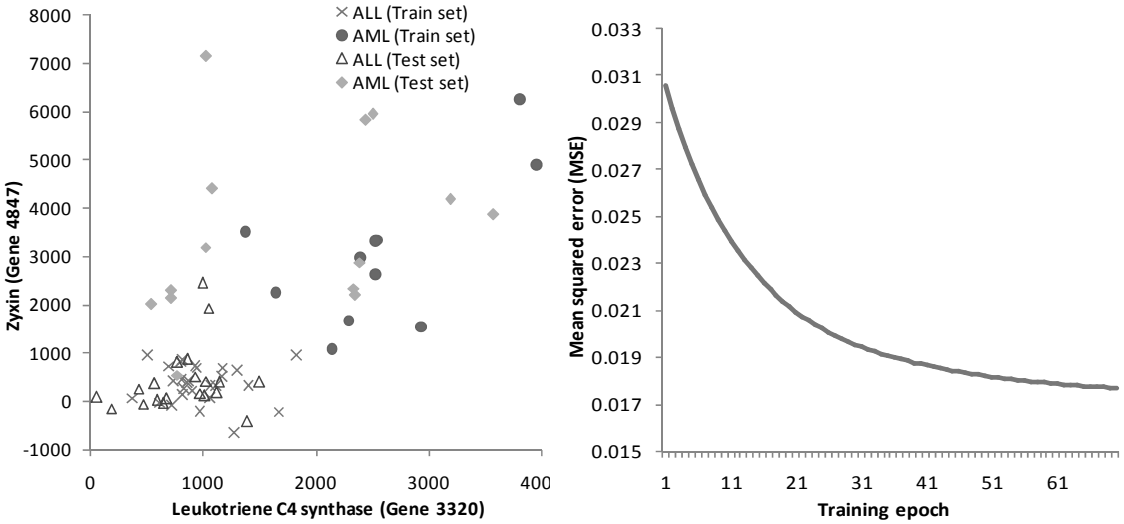


Figure 4.9: Fuzzy labels crafted by RFCMAC-Yager for leukemia dataset

associated with the two genes selected by the RFCMAC-Yager are shown in Figure 4.9, while the corresponding set of rules is provided in Table 4.9.

It can again be observed that the knowledge base formulated by RFCMAC-Yager is very simple and easy to understand. There is also biological evidence supporting the validity of the features and rules identified. For instance, a study in [292] revealed the significance of Zyxin in the prognosis of pediatric AML. Zyxin has also been reported to play a critical role in encoding a LIM domain protein for cell adhesion in fibroblast [45], and more recent studies showed that Zyxin LIM(1-2) facilitates the interaction between the CasL-HEF1 and Crk1 adaptor proteins [293], closely associated with chronic myeloid leukemia and ALL [226]. On the other hand, Leukotriene C4 synthase constitutes the key enzyme in the biosynthesis of the Leukotriene C4, which stimulates the growth of human myeloid progenitor cells and is frequently overproduced in myeloid leukemia [34]. In sum, all these facts suggest that both Zyxin and leukotriene C4 synthase are crucial for leukemogenesis, although further (wet) experiments are needed to confirm the conjecture.

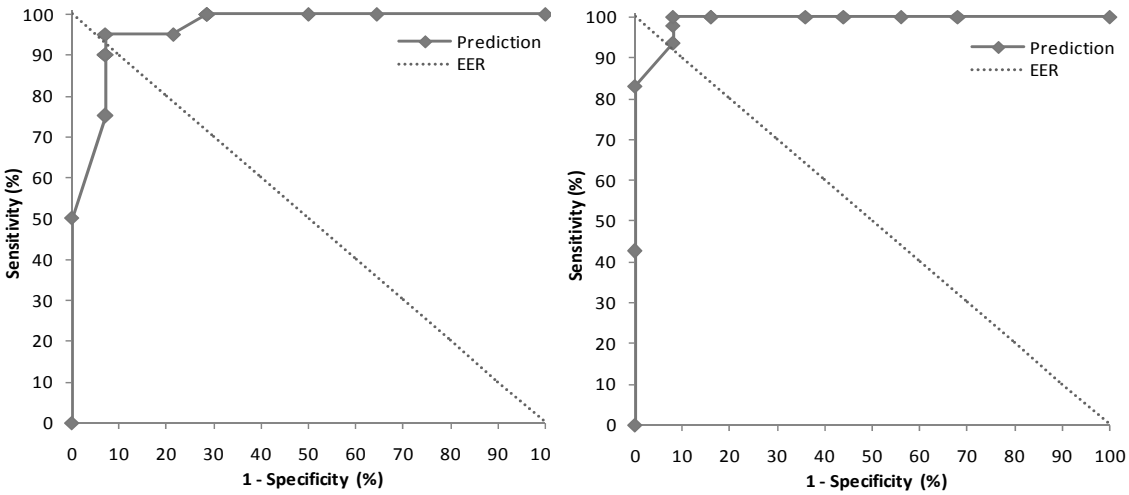


(a) Gene Zyxin versus LTC4S (b) Learning convergence

Figure 4.10: Results of RFCMAC-Yager on leukemia dataset (independent test)

A closer examination based upon the expression patterns of the selected genes depicted in Figure 4.10(a) reveals the validity and intuitiveness of the RFCMAC-Yager rules in Table 4.9. It is shown that the rules conform to the plotted distribution of gene expression levels. Here rule R_1 covers the lower left data region, where most of the ALL cases are present (in both train and test sets), whereas R_2 and R_3 capture the upper and right regions respectively, containing the majority of the AML cases. Despite its simplicity, this set of rules can provide satisfactory predictions; it yields a perfect accuracy on the train set, and 94.12% on the test set (1 mistake from ALL and AML each). The learning curve corresponding to the tuning of these rules is given in Figure 4.10(b), which again showcases the stable learning traits of the system.

The classification performance and robustness of the RFCMAC-Yager network on the (independent) test set under varying decision threshold are illustrated by the ROC plot in Figure 4.11(a), where sensitivity and specificity correspond to the ALL and AML classes respectively. The results are consolidated in Table 4.10, and shown along with some previously published results on the same test set obtained with: Voting Machine [92], SVM with Correlation Feature Selection (SVM-CFS) [211, 282], C4.5 [216, 282], Prediction Analysis of Microarrays (PAM) [269], Partial Least Squares with Logistic Discrimination (PLS-LD) [189], Emerging Patterns (EP) [148], and RBF with Median Vote Relevance (RBF-MVR) [43]. Note that comparisons with (conventional) FCMAC



(a) Independent test (b) Leave-one-out

Figure 4.11: ROC curves for the RFCMAC-Yager predictions on leukemia dataset

Table 4.10: Benchmark results on leukemia dataset (independent test)

Classifier	#Features	#Rules	Accuracy	#Errors (ALL:AML)
Voting Machine	50	-	85.29%	5(*)
SVM-CFS	1	-	91.18%	3(2:1)
C4.5	1	2	91.18%	3(2:1)
PAM	21	-	94.12%	2(*)
PLS-LD	50	-	97.06%	1(0:1)
EP	1	2	91.18%	3(2:1)
SVM-FFS	25-1000	-	88.24-94.12%	2-4(*)
RBF-MVR	50	-	97.06%	1(1:0)
RFCMAC-Yager	2	3	94.12%	2(1:1)

- = not applicable, * = not specified

systems are not made here, due to their lack of scalability in dealing with high-dimensional data. As shown, the SVM-CFS, EP and C4.5 employed only one feature, Zyxin, to perform the classification. Although Zyxin alone is sufficient to (linearly) separate the ALL and AML cases on the train set, it is no longer true for the test set, as illustrated in Figure 4.10(a). Meanwhile, RFCMAC-Yager shows that adding Leukotriene C4 synthase can indeed provide better discrimination, and generalization results, on the test set.

Comparisons can also be made with the other remaining methods i.e., Voting Machine, PAM, PLS-LD, SVM-FFS, and RBF-MVR. As shown in Table 4.10, RFCMAC-Yager achieves a high accuracy rate comparable to PAM and SVM-FFS, albeit second to PLS-LD and RBF-MVR. Despite their superior performances, both PLS-LD and RBF-MVR function as black-box predictors; no meaningful knowledge (rules) can be extracted from

Table 4.11: Benchmark results on leukemia dataset (leave-one-out)

Classifier	#Features	#Rules	Accuracy	#Errors (ALL:AML)
C4.5	2.00	2.00	73.61%	19(*)
PAM	2296.00	-	97.22%	2(*)
TSP	2.00	2.00	93.06%	5(*)
k-TSP	18.00	18.00	95.83%	3(*)
RVM	3.63	-	93.06%	5(*)
SLogReg	5.06	-	94.44%	4(*)
BLogReg	11.59	-	93.06%	5(*)
RBF-MVR	25.00	-	98.61%	1(1:0)
RFCMAC-Yager	2.92	3.92	97.22%	2(0:2)

- = not applicable, * = not specified

the systems. In this respect, the primary advantage of RFCMAC-Yager is the ability to formulate concise, interpretable rule base with significantly smaller set of features, and explain its outputs in a way highly akin to human physicians' decision making process. The interpretability of the system is also further enhanced by the use of linguistic labels in its fuzzy rules (as opposed to crisp rules in C4.5 and EP), thereby allowing the user to understand them in familiar terms and making the system verification easier.

To confirm the performance verity of the proposed system, further experiments adopting *leave-one-out* (LOO) strategy on all 72 samples (i.e., no division between the train and test sets) were conducted, owing to the small data size. This is the same as 72-fold CV, where 71 samples serve to train the system, and 1 to assess its generalization performance. The result is summarized in Figure 4.11(b), showing generalization improvement given larger training samples as compared to the previous result in Figure 4.11(a). Comparisons are then made against several published results based on the LOO setting utilizing: C4.5 [216], PAM [269], Top Scoring Pair (TSP) [82], k-TSP [265], Relevance Vector Machine (RVM) [271], Sparse Logistic Regression (SLogReg) [240], Bayesian Logistic Regression (BLogReg) [40]], and RBF-MVR [43], listed in Table 4.11. The gist of the results (especially comparisons with RBF-MVR) is similar to that of Table 4.10. In general, RFCMAC-Yager shows a good balance between interpretability and prediction accuracy, demonstrating therefore its efficacy as a clinical decision support system.

4.6 Summary

A novel reduced localized neuro-fuzzy system for knowledge extraction is presented in this chapter, termed Reduced Fuzzy Cerebellar Model Articulation Controller (RFCMAC), which incorporates rule base reduction mechanisms to improve system interpretability and generalization. The potential of RFCMAC in identifying highly concise and interpretable rule base while enhancing classification/approximation accuracy has been exemplified via many experimental results. As such, the proposed model offers the necessary features to realize long-term memory modules in the INCA framework and to address complex, large-scale real world task domains. Building upon this idea, the next chapter shall describe an extended computational model of knowledge consolidation that works based on complementary interaction between the transient and long-term memories in the brain. The model is able to rapidly acquire novel patterns without disrupting the existing knowledge base and without recourse to the original patterns already experienced, thus providing a robust and efficient means to handle large sequential learning tasks.

Chapter 5

Dual Network Model for Knowledge Consolidation

5.1 Knowledge Consolidation

In this chapter, a novel dual network model supporting consolidation processes in INCA is proposed that constitutes an extension of the RFCMAC model described in chapter 4. The knowledge extraction and in particular rule base reduction mechanisms of RFCMAC indeed provide a useful and scalable means to identify the simplest description of environmental patterns. However, this capacity only forms one element of the consolidation process. To complete the picture, there is a further need for the system to exhibit the ability to perform a full sequential learning without recourse to the original patterns it was exposed to, and without catastrophically disrupting the knowledge representation already acquired. The dual network model described hereafter is designed to fulfill this requirement, which conforms to the way humans learn a sequence of patterns.

It is well understood that memory consolidation constitutes the primary mechanism by which humans can effectively acquire and organize knowledge about their environment and about themselves [7, 162, 62]. The centerpiece of this paradigm is that storage of knowledge (especially declarative information) involves a two-stage process, whereby memories get transferred from an initial temporary repository, which exhibits fast-acting but short-lived plasticity, into a final permanent place, where learning and forgetting are much slower. The key biological account of this process, as established in neuroscientific studies, involves the interaction between the *hippocampal* and *cortical* sites in the brain, acting as the fast- and slow-learning memories respectively [7, 162].

The precise characteristic of this interaction, however, is still under active debate, particularly regarding whether the hippocampus plays a *temporary* or *permanent* role in the consolidation process [117]. From the temporary perspective, memories are initially captured in the hippocampus, and gradually consolidated into the cortex so that they no longer depend on the hippocampus [289, 295]. Conversely, the permanent view suggests that the hippocampus is always required to recall episodic information (about events with specific spatiotemporal tags), but not necessary for semantic information (about general facts independent of spatiotemporal tags) [180]. Nevertheless, there is a consensus that the main substrate for consolidation in the cortex involves hippocampal-based rehearsal of the activity patterns reflecting past active behavioral states, typically during slow-wave sleep and perhaps rapid eye movement sleep or quiet wakefulness [288]. This observation provides the inspirational basis for the work presented in this chapter.

From a computational perspective, the notion of memory consolidation is often associated with the so-called *stability-plasticity* dilemma in connectionist modeling [103, 95]. Learning in neural networks is most commonly achieved via gradient descent adaptive procedure, the prime example being the backpropagation algorithm [225]. When used in sequential learning tasks, however, such procedure may cause previously learned information to be abruptly erased in the presence of new input, leading to a severe manifestation of the sensitivity-plasticity issue known as *catastrophic interference* [164, 219]. That is, when a network that has previously learned a set of patterns is retrained on a different set of patterns, the newly acquired information may completely abolish the existing representation of the first set. As such, to avoid interference, most connectionist models rely on learning algorithms that require cycling through all of the patterns to be learned over and over, with small adjustment of connection weights made each time until the system settles to a weight-space solution for all patterns. However, this approach is implausible from a cognitive viewpoint, given the fact that humans are able to learn and consolidate without having to see again patterns already experienced.

Numerous attempts have been explored to mitigate catastrophic interference in sequential learning tasks [164, 219, 7, 162, 222, 75, 18]. The main difficulty in solving this

problem lies in the highly distributed, overlapping nature of neural network representation, which shares the same set of connection weights to capture different experiential patterns. When a new set of patterns is learned, the very weights that were already adjusted for the previously learned patterns will be modified once more. Several proposals have been made to mitigate this issue by employing a separated or sparse representation [95, 122, 137, 17]. However, such approaches result in a significant degradation in the generalization ability to extract the underlying structure of patterns learned. There is a need to have distributed representation to achieve high degree of generalization, and at the same time separated representation to eliminate catastrophic interference.

A trivial way to address catastrophic interference is via interleaved rehearsal mechanism i.e., the old information learned previously is continually refreshed/retrained and interleaved with new information being learned [162]. However, this solution is unacceptable as it requires permanent access to all individual patterns on which the system was originally trained, which do not normally repeat themselves for further learning cycles. Moreover, using the original patterns to perform rehearsal may still result in radical forgetting of old information; it is more appropriate to interleave patterns that reflect the current memory representation with new pattern being learned instead [75]. In light of these issues, an alternative mechanism called pseudorehearsal was proposed [222], which involves generating a set of artificial patterns, or *pseudopatterns*, that approximates the information contained within a given network representation. This procedure has been applied to several sequential learning tasks, and the results showed a substantial decrease of catastrophic interference compared to those obtained without it.

This pseudorehearsal method later became the inspirational basis of the dual network models developed independently in [75] and [18], that include two separate, continually interacting modules: one for capturing new patterns and the other for old patterns, with information exchanged between them via pseudopatterns. These models were shown to be able to forget gradually and appropriately perform sequential learning. Although the models can capture the division of labor between the hippocampal and cortical regions to some extent [75, 18], they do not account for the distinct representational and learning

natures of the two regions. An in-depth treatment of this was given in [200], suggesting that the hippocampus employs a sparse conjunctive representation and rapid learning to encode specific events without interference, whereas the cortex slowly learns the general structure of the environment using an overlapping distributed representation. Káli and Dayan [117] recently provided an extension of this account via a hierarchical generative model, with a new take on the roles of rehearsal in keeping episodic patterns stored in the hippocampus in register with the changing cortical representation, and in consolidating semantic information in the cortex. Nonetheless, these models are computationally rather demanding and their applications are so far limited to controlled, or small-scale, tasks.

Based on these inspirations, a novel model of hippocampal-cortical interaction, termed *Dual Consolidation Network* (DCN), has been developed and is described in this chapter. The organization of the DCN model consists of two distinct network modules: a fast-learning module (FLM) employing sparse memory representation and a slow-learning module (SLM) utilizing overlapping representation, that work interactively to build, tune and exploit the system's knowledge base. Consolidation of knowledge based on these two modules is in turn accomplished via an *awake phase*, in which FLM rapidly acquires a new pattern from the environment plus pseudopatterns reinstated from SLM which approximate the content of the old (existing) knowledge base, followed by a *sleep phase*, whereby SLM (re)organizes its rule base structure and tunes its parameters using pseudopatterns generated from FLM in order to transfer the blent (old and new) information captured in the awake phase. What essentially distinguishes DCN from other dual network models [75, 18, 200, 117], that so far work on fixed structure/topology, is the introduction of structural *construction* (expansion) and *reduction* (shrinking) mechanisms in its constituent modules. These mechanisms provide DCN with improved scalability in the face of large task domains and better adaptability to the dynamics of the environment.

The practical advantages of the proposed consolidation procedure can be justified by considering the two major types of learning: *offline* (*batch*) and *online* learning. The former assumes that all patterns to be learned are always available ("cached") and can be accessed repeatedly (e.g. the original backpropagation procedure [225]). With this

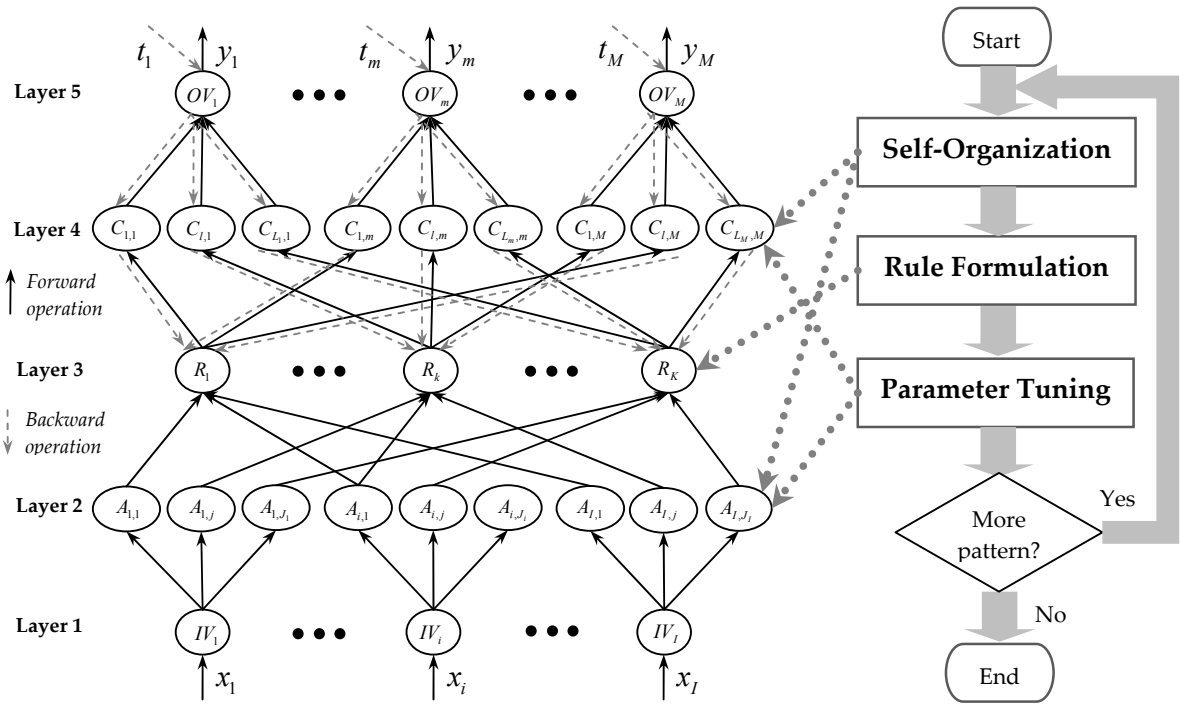
approach, one can specify an objective function for all patterns, which allows to monitor and control learning progress (e.g. error convergence) to any desired precision level. However, such technique often poses high storage requirement in the face of large data, and is not suitable for modeling dynamic real-time environments. Conversely, in online sequential learning, each pattern is discarded after it is processed. This approach is more efficient and can adapt to dynamic, time-varying tasks, but usually faces theoretical difficulties to measure learning progress and is prone to recency effects and catastrophic forgetting. The DCN consolidation procedure thus combines the best of the two camps; it supports fast online sequential learning without recourse to the actual patterns, and through its pseudopattern transfer process, allows stable learning process without suffering from catastrophic interference. By extension, this enables the so-called *representational* knowledge transfer [26], i.e., the network representation previously built for one task can be reused for learning a subsequent, related task. Such ability is of paramount interest in cognitive modeling, particularly in modeling cognitive growth.

Section 5.2 first describes the transient network model used to implement the FLM in DCN. (The SLM, on the other hand, employs the RFCMAC model described in chapter 4, and so its details are not repeated here). The full architecture and the consolidation and inference procedures of DCN are elaborated in section 5.3. Section 5.4 then presents a pedagogical cognitive task showing the basic workings of DCN, and further experimental studies on large-scale tasks are provided in section 5.5. Section 5.6 concludes this chapter.

5.2 Transient Network Model

5.2.1 Connectionist Structure

The transient network used to realize the FLM in DCN is modeled after the *Generic Self-organizing Fuzzy Neural Network* (GenSoFNN) [275, 197], which has five layers of nodes as depicted in Figure 5.1(a). Nodes in layer 1 (input layer), called input variable nodes IV_i, represent the input attributes of interest (e.g. velocity and distance in the case of vehicle control [203]). The layer 2 (antecedent layer) nodes are known as antecedent label nodes $A_{i,j}$, and correspond to linguistic labels such as *Slow*, *Fast* and *Near*, *Far*



(a) Connectionist structure (b) Learning procedure

Figure 5.1: Overview of the transient network model

for the velocity and distance inputs respectively. Nodes in layer 3 (rule layer) are called rule nodes R_k, each representing a fuzzy rule such as 'IF velocity is *Slow* and distance is *Far* THEN accelerator is *High*'. Next, the layer 4 (consequence layer) nodes are called consequent label nodes $C_{l,m}$, and refer to linguistic labels such as *Low* and *High* for the accelerator output. Finally, nodes in layer 5 (output layer), termed output variable nodes OV_m, represent the output attributes of interest (e.g. the accelerator output).

The GenSoFNN architecture implements the Mamdani model of fuzzy system [156], where the total numbers of nodes in layer 1-5 are denoted as I, J, K, L and M respectively. Every node IV_i ($i \in \{1, ..., I\}$) in layer 1 refers to a single input x_i, which is an element of input vector $\mathbf{x} = \{x_1, ..., x_i, ..., x_I\}$. Similarly, each OV_m ($m \in \{1, ..., M\}$) in layer 5 computes an output y_m belonging to the output vector $\mathbf{y} = \{y_1, ..., y_m, ..., y_M\}$. A target output vector $\mathbf{t} = \{t_1, ..., t_m, ..., t_M\}$ serves as a teaching signal to train the system. Each IV_i in layer 1 may have J_i different antecedent labels, and the total number of layer 2 nodes is $J = \sum_{i=1}^{I} J_i$. Likewise, each OV_m in layer 5 can have L_m consequent labels, and the total number of layer 4 nodes is $L = \sum_{m=1}^{M} L_m$. Finally, each $A_{i,j}$ in layer 2 and $C_{l,m}$ in layer 4 may be linked to more than one rule R_k ($k \in \{1, ..., K\}$) in layer 3.

5.2.2 Learning Procedure

The GenSoFNN model previously described employs an online sequential learning mechanism to generate its fuzzy rule base [275, 197]. That is, links and nodes that define the rule base are constructed and/or adjusted on the fly for every data pattern presented to the network. This evolving learning procedure is designed to ensure that only relevant fuzzy rules and labels are created, and that the system can continuously adapt to environmental changes while maintaining a consistent rule base. Figure 5.1(b) provides an overview of the learning procedure, which consists of three phases: *self-organization*, *rule formulation*, and *parameter tuning*, all taking place in a single pattern presentation.

Self-Organization. In this phase, the (unsupervised) *Discrete Incremental Clustering* (DIC) algorithm [275] is employed to generate the input and output fuzzy labels. The algorithm creates fuzzy labels in a local fashion, whereby the number of labels in one input/output dimension need not be equal to that in another input/output dimension. Moreover, DIC ensures that each label is uniquely represented by one fuzzy set, and the order of all labels is retained throughout the training process (e.g. a new label B created to the right of an existing label A will not appear on the left of A during training, and vice versa). This is essential to ensure the consistency of the rule base. Briefly, the DIC algorithm for the input section proceeds as follows. First, for each input dimension, the index $Winner_i \in \{1, ..., J_i\}$ of the best-fit antecedent label is computed via (Eq. 5.1)

$$Winner_i = \arg \max_{j \in \{1,...,J_i\}} \{\mu_{i,j}(x_i)\} \qquad \text{(Eq. 5.1)}$$

where $\mu_{i,j}(x_i)$ is the fuzzy membership value of antecedent label $A_{i,j}$ corresponding to the current input x_i. Next, the algorithm checks whether $\mu_{i,j}(x_i) > \lambda$, where $\lambda \in (0, 1]$ is a user-specified membership threshold. If so, the kernel of the winning antecedent label $A_{i,Winner_i}$ is expanded to widen its coverage, with an expansion rate defined by a plasticity parameter $\beta \in [0, 0.5]$ and tendency parameter $TD \in [0, 0.5]$; otherwise, a new label is created with its kernel centered at x_i. The construction of the consequent labels proceeds in the same way, with the terms i, j, J_i and x_i replaced with m, l, L_m and t_m, respectively. More details about the DIC algorithm can be found in [275, 194].

Rule Formulation. This phase aims at establishing a set of fuzzy rules linking the antecedent and consequent labels crafted in the self-organization phase. In this, only one rule can be created for every data point at most. Accordingly, the total number of rules K that may be generated is constrained as $K \leq N$, where N is the total number of data points. The rule formulation phase begins with forward and backward firing operations from layer 1 to 3 and 5 to 3 respectively, and then computes the strongest firing strengths $Z_{k'}^{fwd}$ and $Z_{k''}^{bwd}$ belonging to rules $R_{k'}$ and $R_{k''}$, as per (Eq. 5.2)-(Eq. 5.3)

$$Z_{k'}^{fwd} = \max_{k\in\{1,...,K\}} \left\{ \min_{i\in\{1,...,I\}} \mu_{(i,j)_k}(x_i) \right\} \tag{Eq. 5.2}$$

$$Z_{k''}^{bwd} = \max_{k\in\{1,...,K\}} \left\{ \min_{m\in\{1,...,M\}} \mu_{(l,m)_k}(t_m) \right\} \tag{Eq. 5.3}$$

where $\mu_{(i,j)_k}(x_i)$ and $\mu_{(l,m)_k}(t_m)$ denote the membership values of labels $A_{i,j}$ and $C_{l,m}$ that are connected with rule R_k, respectively. Subsequently, based on these computations, a new rule will be created whenever the condition (Eq. 5.4) is satisfied

$$\begin{aligned}(Z_{k'}^{fwd} < \delta) \vee (Z_{k''}^{bwd} < \delta) \vee ((Z_{k'}^{fwd} \geq \delta) \wedge (Z_{k''}^{bwd} \geq \delta) \wedge [(R_{k'} \neq R_{k''}) \vee \\ (\exists i \in \{1,...,I\}(\mu_{(i,j)_{k'}}(x_i) < \delta)) \vee (\exists m \in \{1,...,M\}(\mu_{(l,m)_{k''}}(t_m) < \delta))])\end{aligned} \tag{Eq. 5.4}$$

A newly created rule is not always added into the rule base, depending on whether its inclusion affects the overall rule base consistency. That is, a new rule R_{new} is committed into the final rule base only if it maintains the consistency criterion (Eq. 5.5)

$$\begin{aligned}\forall k \in \{1,...,K\}(\forall i \in \{1,...,I\}(A_{(i,j)_k} = A_{(i,j)_{new}}) \wedge \\ \forall m \in \{1,...,M\}(C_{(l,m)_k} = C_{(l,m)_{new}}))\end{aligned} \tag{Eq. 5.5}$$

where $A_{(i,j)_k}$ and $C_{(l,m)_k}$ are the j^{th} label of the i^{th} input dimension and the l^{th} label of the m^{th} output dimension, respectively, that are linked to rule R_k.

Parameter Tuning. A least mean square (LMS)-based optimization process, similar to that of RFCMAC, is carried out in this phase for every p^{th} data point. As in RFCMAC, the goal is to minimize a cost function $E^{(p)}$ defined in (Eq. 5.6)

$$E^{(p)} = \frac{1}{2} \sum_{m=1}^{M} \left(t_m^{(p)} - y_m^{(p)}\right)^2 \tag{Eq. 5.6}$$

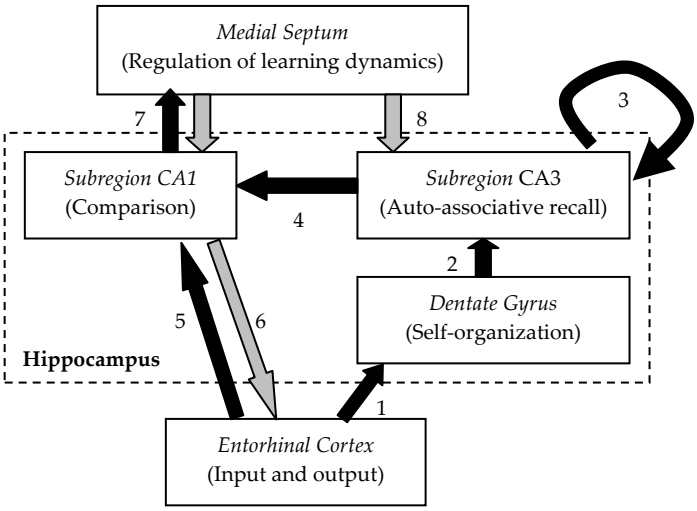


Figure 5.2: Signal pathways in the hippocampal system (adapted from [101])

where $t_m^{(p)}$ and $y_m^{(p)}$ are the m^{th} target and network outputs for the p^{th} data point respectively. Different from the localized LMS of RFCMAC, however, this procedure involves *globalized* updating of the kernel parameters of *all* antecedent and consequent labels. The error signals and updating steps propagate from layer 5 to layer 2 with respect to t_m. The updating rules for the kernel parameters $\theta_{i,j}^{(p)}$ of layer 2 nodes and $\theta_{l,m}^{(p)}$ of layer 4 nodes are given respectively in (Eq. 5.7) and (Eq. 5.8):

$$\theta_{i,j}^{(p)} = \theta_{i,j}^{(p-1)} - \eta \frac{\partial E^{(p)}}{\partial \theta_{i,j}^{(p)}} \qquad \text{(Eq. 5.7)}$$

$$\theta_{l,m}^{(p)} = \theta_{l,m}^{(p-1)} - \eta \frac{\partial E^{(p)}}{\partial \theta_{l,m}^{(p)}} \qquad \text{(Eq. 5.8)}$$

where η is the learning rate. If Gaussian kernel is adopted, then the term $\theta_{i,j}$ ($\theta_{l,m}$) here denotes either the centroid or width of the input (output) kernel. The full derivation for the updating rules of the kernel parameters in layer 2 and 4 can be found in [194].

A plausible correspondence between the above three-phase learning procedure and the processes in the hippocampal subregions may be justified from the conceptual model described in [101], as shown in Figure 5.2. In this model, the entorhinal cortex provides inputs to the hippocampus and transmits its outputs back to the cortex. During the self-organization phase, the dentate gyrus constructs the representation of input patterns in the entorhinal cortex, minimizing overlaps between them so as to enable *pattern separation*, and passes this representation to the subregion CA3. In a process akin to

the rule formulation phase, the subregion CA3 performs auto-associative encoding of the patterns from dentate gyrus, which provides a *pattern completion* mechanism to cope with partial and/or noisy input cues. This process is emulated through the conjunctive encoding of fuzzy rules, as per (Eq. 5.2) and (Eq. 5.3). Subsequently, the subregion CA1 *compares* the inputs from the entorhinal cortex with the outputs of CA3 and measures how well they match. An approximation to this process is given in (Eq. 5.4). Finally, a tuning phase takes place whereby the medial septum regulates the learning dynamics by influencing synaptic modification and transmission, depolarization and adaptation.

5.3 Dual Consolidation Network

5.3.1 Connectionist Structure

As mentioned in section 5.1, the DCN structure is composed of a fast-learning module (FLM) and slow-learning module (SLM), emulating the hippocampal and cortical circuits respectively. The SLM is realized using the RFCMAC model described in chapter 4, while the FLM employs the GenSoFNN model presented in section 5.2. Apart from these, DCN includes an *input layer* to receive input stimulus (features) from the environment, an *attention layer* to select or combine the signals from the input layer, an *output layer* to capture the outputs inferred by the SLM and FLM, and a *combination layer* to aggregate the two modules' outputs. A schematic illustration of DCN structure (which is adapted from Figure 3.5 in chapter 3) is provided in Figure 5.3(a).

Communication between the SLM and FLM is then achieved via *consolidation* and *inference* procedures, which provide respectively for the acquisition and exploitation of the DCN knowledge base. When the consolidation procedure is carried out, the SLM and FLM interact by exchanging input and output signals via the attention and output layers, following the pathways shown by the consolidation links in Figure 5.3(a). During the inference process, the input signals propagate from the attention layer to the output layers and the output signals from the SLM and FLM are aggregated in the combination layer, as per the inference links in Figure 5.3(a). More detailed descriptions of the consolidation and inference procedures shall be provided in sections 5.3.2 and 5.3.3, respectively.

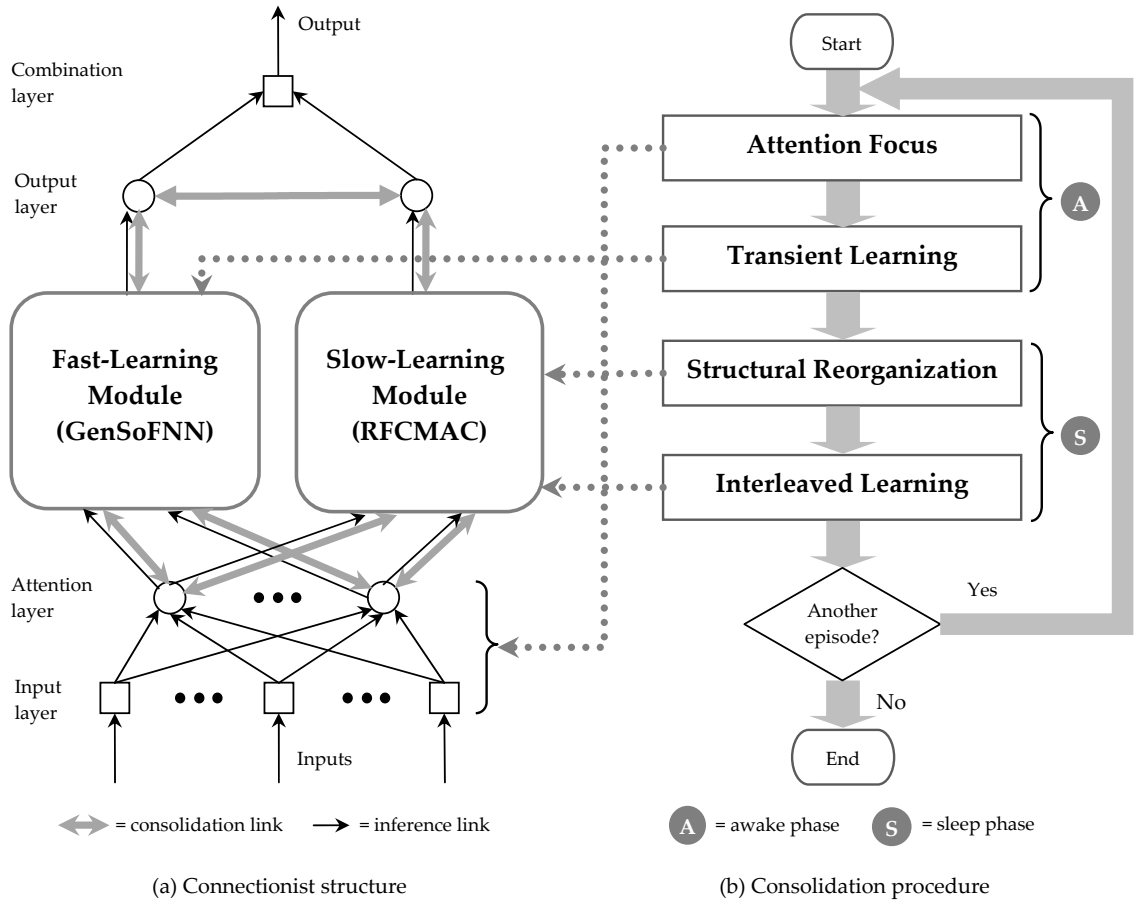


Figure 5.3: Overview of the proposed DCN framework

5.3.2 Consolidation Procedure

An outline of DCN consolidation procedure is presented in Figure 5.3(b), which involves four major steps: *attention focus*, *transient learning*, *structural reorganization*, and *interleaved learning*. The first two steps roughly correspond to the knowledge acquisition process in humans during awake phase, which involves learning of new patterns from the environment as well as reinstatement of the previously acquired information from the long-term memory. On the other hand, the last two steps approximate the rehearsal process occurring during sleep period (e.g. slow-wave sleep [288, 108]), which involves transfer of information acquired during the awake period into the long-term memory. In the DCN framework, the task at hand is assumed to consist of various experiences and be decomposable into independent data chunks occurring at different time periods, termed *episodes*. Accordingly, the proposed consolidation procedure works on a multi-episode basis, with the awake and sleep phases carried out in sequence in each episode.

5.3.2.1 Awake Phase

Attention Focus. The awake phase commences with an attention focus process, building the DCN attention layer through selecting or combining input features that contribute substantially to the final outputs. In the current implementation, this process is achieved via the *feature construction* step of the RFCMAC system, which aims at selecting a subset of features which are highly correlated with the output and uncorrelated with each other[1], as explained in chapter 4, section 4.3.1. There are $D \ll I$ input features selected in this step, where I is the total number of original features. With reference to the architecture in Figure 5.3(a), the result of feature selection is reflected by ensuring that each attention layer node has only one active link from one input layer node corresponding to a unique, selected feature, with the remaining links from the non-selected input layer nodes being disabled (i.e., by setting the link weights to 0).

Transient Learning. This step is carried out for every p^{th} data point $\mathbf{d}_e^{(p)} = (\mathbf{x}^{(p)}, \mathbf{t}^{(p)})$ coming from the environment in the e^{th} episode, where $\mathbf{x}^{(p)}$ and $\mathbf{t}^{(p)}$ are the input and target output vectors, respectively. Figure 5.4(a) presents the detailed view of this step that comprises two substages. The first substage involves reinstating (part of) the old/existing knowledge stored in SLM and supplying the reinstated pseudopatterns into FLM, while the second part involves capturing in FLM the new information $\mathbf{d}_e^{(p)}$ from the environment. In the first part, whenever the SLM rule base is not empty (i.e., $K_{SLM} > 0$), a *pseudopattern* $\psi_{SLM}^{(q)}$ $(q \in 1, ..., P)$ is generated each time from SLM using (Eq. 5.9)

$$\begin{aligned} \psi_{SLM}^{(q)} &= (\psi_{\mathbf{x}}^{(q)}, \psi_{\mathbf{t}}^{(q)}) \\ &= (\left\{m_{1,j}^k, ..., m_{i,j}^k, ..., m_{I,j}^k\right\}, \left\{m_{l,1}^k, ..., m_{l,m}^k, ..., m_{l,M}^k\right\}) \end{aligned} \qquad \text{(Eq. 5.9)}$$

where $\psi_{\mathbf{x}}^{(q)}$ and $\psi_{\mathbf{t}}^{(q)}$ are the pseudo-input and -target vectors generated in the q^{th} iteration, and $m_{i,j}^k$ and $m_{l,m}^k$ are the centroids of the antecedent and consequent labels belonging to an SLM rule R_k that is randomly selected in the q^{th} iteration, respectively. Here the rule selection is carried out with each rule being equally likely/probable.

[1]For simplicity and computational efficiency, the attention focus step is currently carried out based only on data from the first episode. A more sophisticated procedure that involves multi-episode attention focus mechanism is beyond the scope of the present work and will be investigated in the future.

Structural updating is then performed that involves determining if a new label and/or rule need to be created in FLM to cover each pseudopattern $\psi_{SLM}^{(q)}$. This is accomplished via the *self-organization* and *rule formulation* procedures of the FLM, as described in section 5.2.2. When a new rule is added and the FLM rule base size K_{FLM} exceeds a predefined capacity C, the *least recently used* rule (i.e., the oldest rule that is fired most remotely in time) is discarded. This ensures that each rule in FLM is unique, and that its rule base size is (realistically) bounded, as is the hippocampus. The above processes are repeated for P times, where $P > 0$ is a prespecified pseudopattern count, $N_e = |\mathbf{D}_e|$, and $\mathbf{D}_e = \{\mathbf{d}_e^{(1)}, ..., \mathbf{d}_e^{(p)}, ..., \mathbf{d}_e^{(N_e)}\}$ is the set of data points from the e^{th} episode.

The second substage of transient learning is essentially the same as the first, except that it is done on the actual data point $\mathbf{d}_e^{(p)}$ (instead of pseudopattern $\psi_{SLM}^{(q)}$). It must also be noted that, in order to enforce a sparse memory representation that is interference-free and can capture episodic information, no kernel expansion is performed during the *self-organization* phase of the FLM learning. That is, the kernel parameters of the antecedent and consequent labels in FLM would remain unaltered throughout the transient learning process. This is done by setting the plasticity parameter $\beta = 0$ and tendency parameter $TD = 0$. For the same reason, no *parameter tuning* step is carried out after the *rule formulation* step in Figure 5.4(a) (which contrasts with the original learning procedure described in section 5.2.2). This is achieved by damping the learning rate $\eta = 0$.

5.3.2.2 Sleep Phase

Structural Reorganization. This first step in the sleep phase is to (re)organize the structure of the SLM rule base. This is accomplished by presenting into SLM a set of pseudopatterns $\mathbf{S} = \{\psi_{FLM}^{(1)}, ..., \psi_{FLM}^{(p)}, ..., \psi_{FLM}^{(N_e)}\}$ reinstated from FLM, where each $\psi_{FLM}^{(p)}$ is generated in a similar manner as in (Eq. 5.9). An overview of this process is provided in Figure 5.4(b). Using $\mathbf{S}$, the antecedent and consequent labels in SLM are first generated and pruned via the *label construction* and *reduction* steps, respectively, corresponding to the label generation phase described in chapter 4, section 4.3.1. Subsequently, fuzzy rules are respectively created and pruned via the *rule construction* and *reduction* steps of the rule generation process elaborated in chapter 4, section 4.3.2.

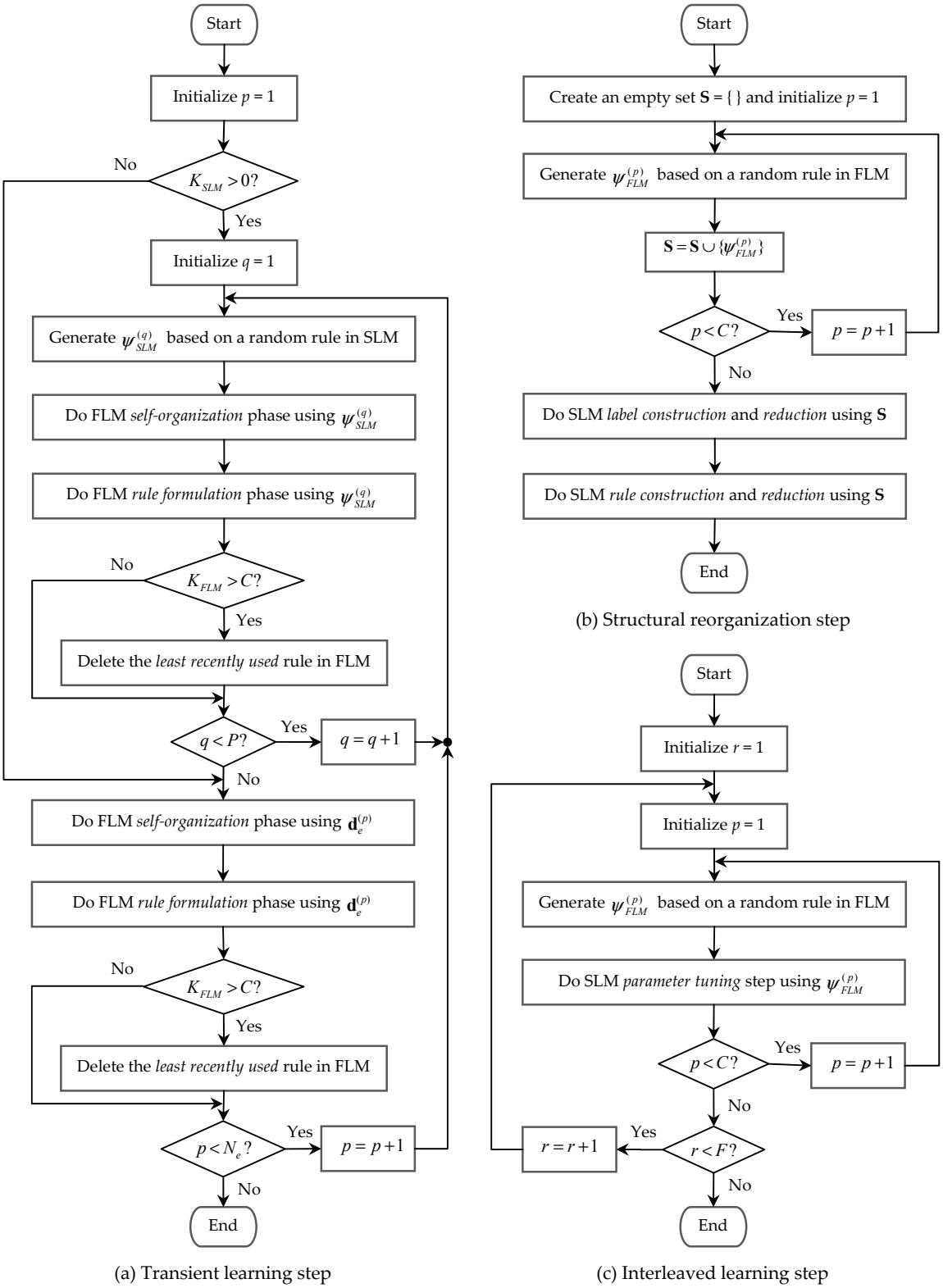


Figure 5.4: Consolidation procedure of the DCN system

Note that, since the DCN consolidation procedure deals with multi-episode learning, the result of structural reorganization in one episode would likely influence the learning process in the next episode. It is thus not advisable to directly remove during the rule

reduction step (in one episode) the consistent/duplicate rules and their respective antecedent links from physical memory, as they may actually be relevant to the subsequent episodes. To resolve this issue, pruning of consistent rules is done *virtually* instead, i.e., by disabling the rules (e.g. damping their firing strengths to 0) and the antecedent links (e.g. setting the link weights to 0) rather than deleting them immediately from the physical memory. In this way, old rules that are partially representative of the association patterns in the future episodes may be recovered and reutilized accordingly.

In addition, during the rule construction step in a given episode, it is possible that ambiguous rules be generated in the SLM which yield inconsistencies with the rule base constructed in the previous episodes. To tackle this issue, an extra checking procedure is performed prior to the rule reduction step to identify these ambiguous rules and compare their accumulated Hebbian weights $W_{k,l,m}$, as defined in (Eq. 4.19)-(Eq. 4.20)) of chapter 4, section 4.3.2. Based on this comparison, the winning rule with the strongest accumulated weight is switched on, whereas the remaining ambiguous rules are disabled (again, not physically removed, to cater for future episodes). As such, the consistency of the SLM rule base is maintained throughout a given episode. The rule reduction step is then carried out as per normal on the SLM structure, ignoring the disabled rules.

Interleaved Learning. This step involves iteratively optimizing the kernel parameters of the SLM rule base, again using pseudopatterns $\psi_{FLM}^{(p)}$ reinstated from FLM. Specifically, one-step parameter tuning process is performed for each $\psi_{FLM}^{(p)}$, based on the *Least Mean Square* (LMS) procedure described in chapter 4, section 4.3.3. This pseudorehearsal process is repeated for $F \times C$ times, where $F > 0$ is a user-specified rehearsal frequency. Figure 5.4(c) summarizes the interleaved learning process. Ultimately, feeding together the new and old information captured in the FLM to the SLM would lead to an interleaved learning regimen that allows new information to be incorporated gradually, while enabling the old information to be refreshed to prevent forgetting.

To summarize, the overall DCN consolidation procedure posits several merits from both intra- and inter-episode learning points of view. In the former case, the transient learning step occurring at the FLM allows the representative (old or new) patterns to be

stored separately in their nearly exact form i.e., without interference. This also enables full sequential learning, whereby the system can adapt its structure and parameters based on a single presentation of external/actual patterns from the environment. In addition, since the FLM represents a "cleaned" version of either the actual or pseudopatterns (i.e., similar patterns are stored only once), one can expect a fast and balanced training of the SLM during the structural reorganization and interleaved learning steps, useful to deal with large data samples and/or skewed data distribution. In extension, the combination of the above three steps facilitates a robust inter-episode consolidation to grasp new information in recent episode without interference, and to enable generalization of knowledge across episodes. As discussed, this contrasts with classical sequential learning methods (e.g. [225]), which are typically biased toward the most recent episodes.

5.3.3 Inference Procedure

The DCN inference procedure allows exploitation of the consolidated rule base by aggregating the outputs computed by SLM and FLM in the combination layer. This process is separate from the DCN consolidation procedure; the former involves combining the outputs of the two modules during recall and generalization operations, whereas the latter achieves learning by invoking only one module at any given time. In the current implementation, a simple 'gating' procedure is adopted to compute the final (consensus) DCN output vector $\mathbf{y} = \{y_1, ...y_m, ..., y_M\}$, as given in (Eq. 5.10)

$$\mathbf{y} = \begin{cases} \mathbf{y}_{SLM} & \text{if } K_{SLM} \neq 0 \\ \mathbf{y}_{FLM} & \text{if } K_{SLM} = 0 \end{cases} \qquad \text{(Eq. 5.10)}$$

where $\mathbf{y}_{SLM}$ and $\mathbf{y}_{FLM}$ are the output vectors deduced by SLM and FLM respectively, and K_{SLM} is the number of rules that are activated in SLM. That is, for a given test point, the outputs from FLM are taken as the final DCN outputs only when no rule in SLM is fired; otherwise, the SLM outputs are used. The former corresponds chiefly to the *inference hole* paradigm, which happens due to the localized inference in SLM invoking only a small portion of its rule base. Such prioritization of SLM in (Eq. 5.10) thus fosters better generalization, as it is most likely that the SLM structure obtained after consolidation has already captured largely the salient features of the domain being

modeled. On the other hand, the use of $\mathbf{y}_{FLM}$ in the 'gating' procedure chiefly provides a 'patching' mechanism to help address the inference hole issue in the SLM.

To derive the output $y_m \in \mathbf{y}_{SLM}$ from the SLM, the localized Yager inference procedure already described in chapter 4 is used, as given in (Eq. 5.11)

$$y_m = \frac{\sum_{k \in \mathbf{S}} \frac{m_{(l,m)_k} f_k}{\sigma_{(l,m)_k}(2-f_k)}}{\sum_{k \in \mathbf{S}} \frac{f_k}{\sigma_{(l,m)_k}(2-f_k)}} \tag{Eq. 5.11}$$

On the other hand, to compute the output $y'_m \in \mathbf{y}_{FLM}$ from the FLM, the inference procedure expressed in (Eq. 5.12) is adopted

$$y'_m = m_{l*,m} \mid l* = \arg \max_{l \in \{1,\ldots,L_m\}} \left\{ \max_k \left\{ Z^k_{l,m} \right\} \right\} \tag{Eq. 5.12}$$

where $m_{l,m}$ is the centroid of consequent label $C_{l,m}$, $Z^k_{l,m}$ is the firing strength of a rule R_k linked to $C_{l,m}$, and L_m is the number of labels in the m^{th} output. In effect, such winner-takes-all process yields a sparse rule base activation, allowing the FLM outputs to depict precisely the episodic information recently captured. This approach is particularly useful to boost the recall of new information that is not yet consolidated into SLM.

5.3.4 Complexity Analysis

Using the mathematical notations given in the previous sections, Table 5.1 summarizes the worst time complexities of the steps in the DCN consolidation procedure for the e^{th} episode, comprising N_e data patterns. First, the execution time of the attention focus step depends on the problem dimensionality; it is proportional to the number of (original) input features I. This complexity is relatively low nevertheless, being controlled by the parameter D that is usually configured to be much smaller than I (i.e., $D \ll I$). Following attention focus is the transient learning step, which are governed by the pseudopattern count P and maximum FLM capacity $C \leq N_e$. However, as the step is performed on the reduced D-dimensional feature space and since small P value (e.g. $P \leq 5$) is adequate for most tasks, the complexity of the step is considerably low as well.

Subsequently, the structural reorganization step imposes the shortest execution time compared to the other consolidation steps, involving only a single pass of pseudopatterns generated from the FLM. On the other hand, a longer computational time is required

Table 5.1: Complexity analysis of the DCN consolidation procedure

Learning phase	Time complexity	Description
Awake phase		
1) *Attention focus*	$O(N_e \times D \times I \times M)$	Calculate the input-output and input-input feature correlations, and add one selected input at a time
2) *Transient learning*	$O(P \times N_e \times (C \times (D+M) + \sum_{i=1}^{D} J_i^{FLM} + \sum_{m=1}^{M} L_m^{FLM}))$	For each data point, check if new label and rule need to be added in FLM, and if the rule base is consistent
Sleep phase		
1) *Structural reorganization*	$O(K_{SLM}^2 \times (D+M) + C \times (\sum_{i=1}^{D} J_i^{SLM} + \sum_{m=1}^{M} L_m^{SLM}))$	Perform Hebbian learning to construct the rules, and then find and reduce the consistent (duplicate) rules
2) *Interleaved learning*	$O(F \times C \times K_{SLM} \times (D+M))$	For each data point, update the kernel parameters of the labels belonging to the selected rules in SLM

for the interleaving learning step, which involves F iterations of rehearsal process. However, because the rehearsal involves a localized parameter tuning to update only selected antecedent/consequent labels within some neighborhood (as per the RFCMAC model), the complexity of the step remains relatively low, especially when compared against the more traditional, globalized sequential learning techniques e.g. back-propagation [225]. This is also compounded by the roles of the attention focus and structural reorganization steps in significantly reducing/simplifying the SLM rule base structure prior to tuning.

Meanwhile, the space complexity of the consolidation procedure can be computed in a straightforward manner. It is simply the sum of the memory requirements of the SLM and FLM modules in the DCN system, as defined in (Eq. 5.13)

$$O\left(K_{SLM} + K_{FLM} + \sum_{i=1}^{D}\left(J_i^{SLM} + J_i^{FLM}\right) + \sum_{m=1}^{M}\left(L_m^{SLM} + L_m^{FLM}\right)\right) \qquad \text{(Eq. 5.13)}$$

As indicated in section 5.3.2, the number of FLM rules K_{FLM} is upper bounded by C, where $C \leq N$. Similarly, the SLM rule base K_{SLM} is constrained as $K_{SLM} \leq N$. These subsequently imply a the total (worst) space complexity as given in (Eq. 5.14)

$$O\left(N + \sum_{i=1}^{D}\left(J_i^{SLM} + J_i^{FLM}\right) + \sum_{m=1}^{M}\left(L_m^{SLM} + L_m^{FLM}\right)\right) \qquad \text{(Eq. 5.14)}$$

Regardless, this requirement is considerably low, which can again be attributed chiefly to the attention focus and structural reorganization steps.

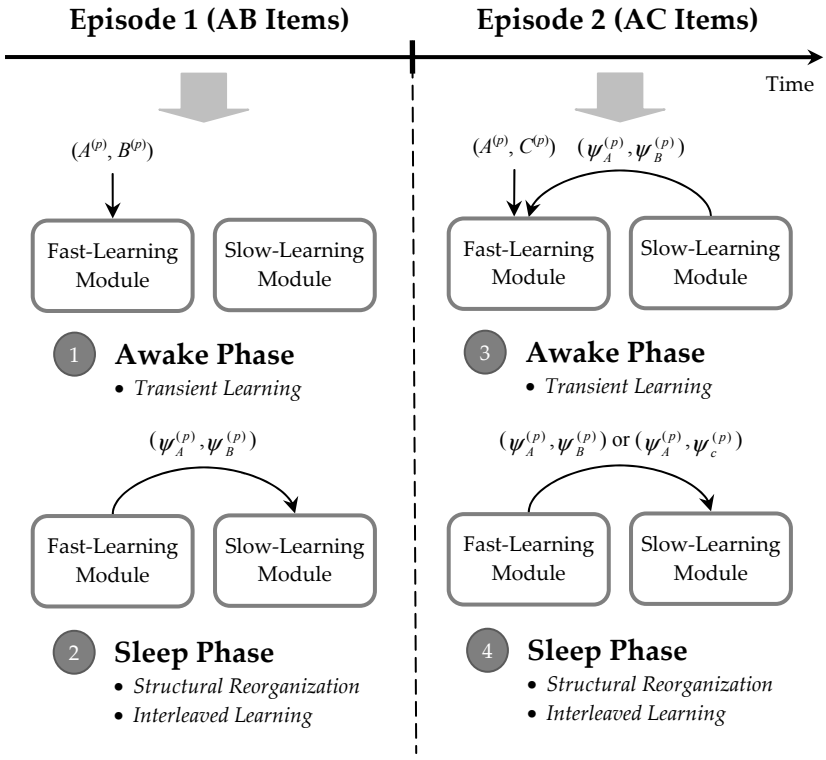
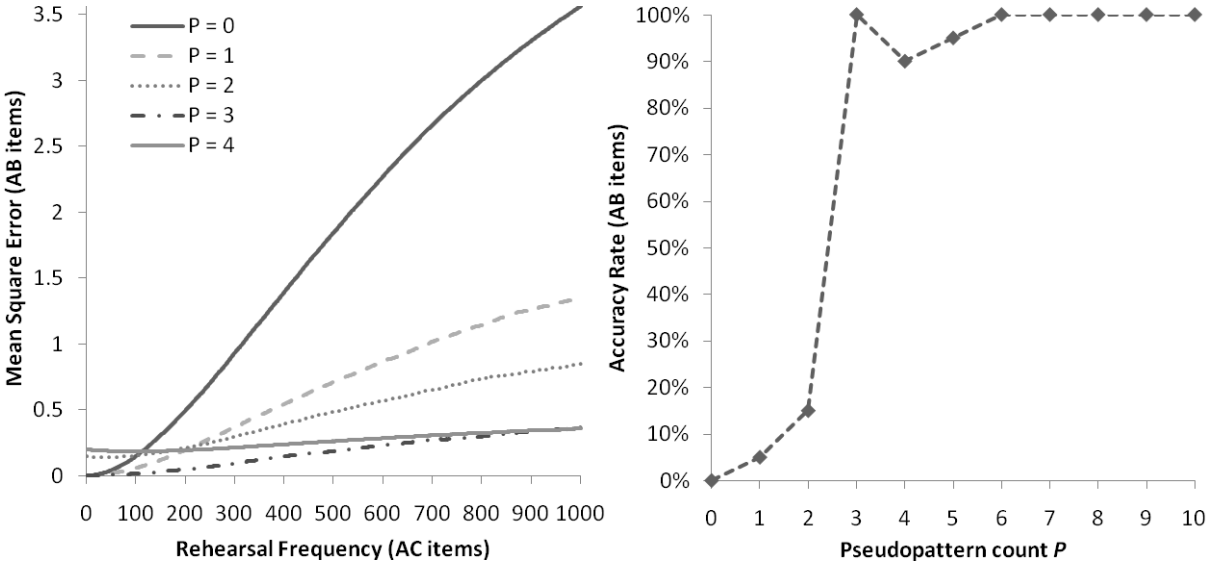


Figure 5.5: Workflow of the DCN consolidation procedure for the AB-AC list task

5.4 Pedagogical Example

To illustrate the workings of the DCN consolidation procedure (and by extension the INCA consolidation cycle) and how it can resolve catastrophic interference, a pedagogical example of sequential learning task, termed *AB-AC* list task, is provided that is a fundamental cognitive skill task classically used to demonstrate catastrophic forgetting in connectionist models [164]. The task is made up of two episodes, each comprising 20 distinct stimulus-response pairs to be learned. The same stimuli $A^{(p)}$ (where $p = 1, \ldots, 20$) are received in both episodes, but with different response patterns $B^{(p)}$ and $C^{(p)}$, respectively. Items $A^{(p)}$, $B^{(p)}$, $C^{(p)}$ are represented by binary-valued vectors, each comprising 5 bits with values 0 or 1 being equally likely. The DCN is tasked to sequentially learn the set of *A-B* associations in episode 1, and then that of *A-C* associations in episode 2.

The DCN workflow for this task is shown in Figure 5.5. Both the SLM and FLM are initially empty. In episode 1, the FLM first performs rapid online learning to capture each external pattern $(A^{(p)}, B^{(p)})$, as per the transient learning steps of the DCN consolidation procedure. No attention focus step is assumed here, and the maximum FLM capacity



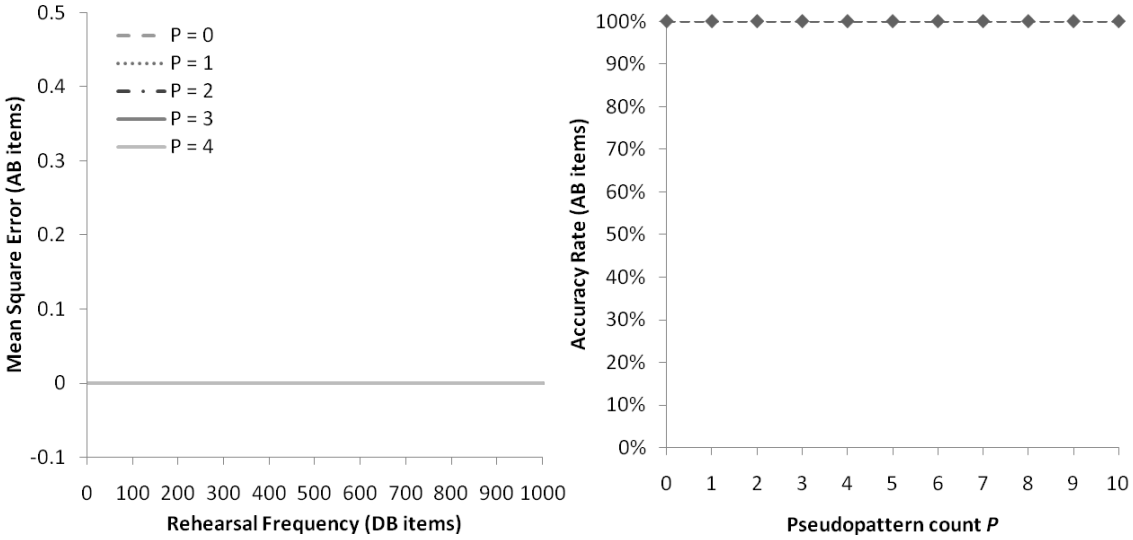
(a) Recall mean square error

(b) Recall accuracy rate

Figure 5.6: Recall of A-B items after learning A-C items

C is set to the length of an episode N_e (i.e., $C = N_e = 20$), which emerges naturally from the consideration that the FLM (ideally) learns one episode by one episode. In step 2, the FLM repeatedly generates C pseudopatterns $(\psi_A^{(p)}, \psi_B^{(p)})$ from the centroid coordinates of randomly selected rules and uses them to train the SLM in an offline fashion. For simplicity, no external pattern occurs concurrently in this step. Based on the pseudopatterns from the FLM, the SLM initializes its structure (also forming 20 rules) and subsequently performs F epochs of parameter tuning, as described in section 5.3.2.2, to gradually grasp the FLM representation of the A-B associations. For this experiment, the rehearsal frequency F is set between 1 to 1000.

During episode 2 (step 3), a set of P pseudopatterns $(\psi_A^{(p)}, \psi_B^{(p)})$ (where $P \ll N_e$) are generated from the SLM in the same way as that from FLM in episode 1, and subsequently fed into FLM together with each new external pattern $(A^{(p)}, C^{(p)})$. Whenever a new rule needs to be added in the FLM to capture an incoming pattern and its current size is at maximum capacity, the least recently used rule is deleted. In step 4, N pseudopatterns are generated from the FLM, which now contains $(\psi_A^{(p)}, \psi_B^{(p)})$ and $(\psi_A^{(p)}, \psi_C^{(p)})$ patterns reflecting the A-B and A-C associations, in order to tune the structure and parameters of the SLM. These correspond to the structural reorganization and interleaved learning steps of the DCN consolidation procedure, respectively.

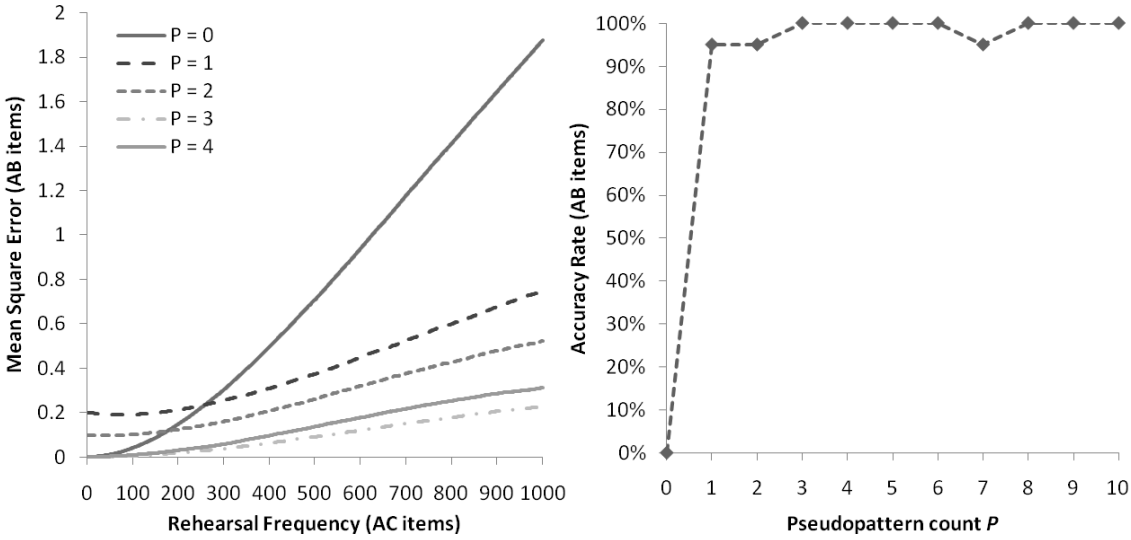


(a) Recall mean square error (b) Recall accuracy rate

Figure 5.7: Recall of A-B items after learning A-B and D-B items

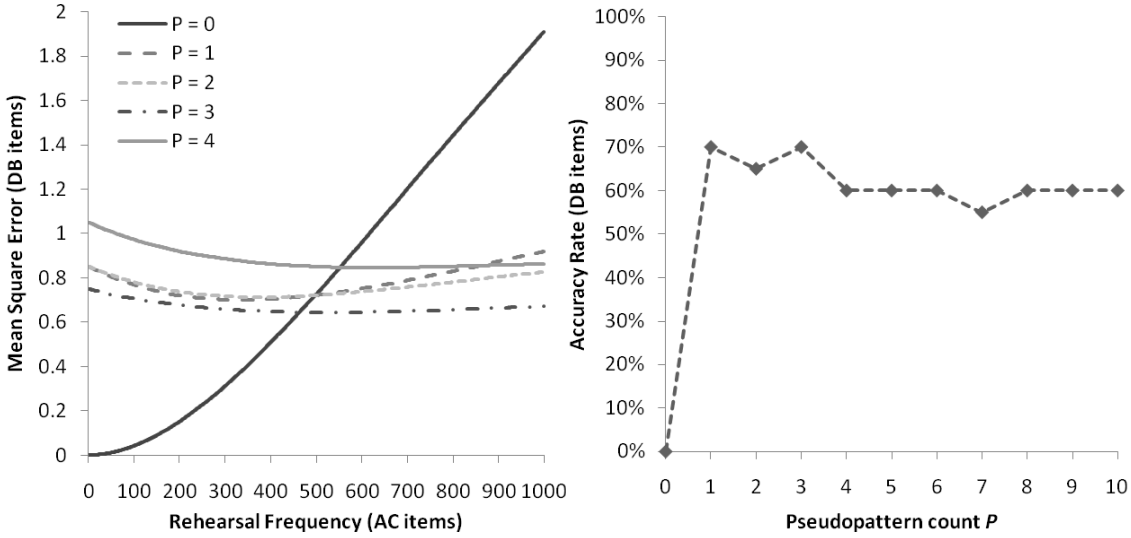
Simulations were conducted to study the effects of the number of pseudopatterns P (transmitted from SLM to FLM) on the recall performance of the old A-B patterns while learning A-C patterns in episode 2. Figure 5.6(a) shows the recall trace across different rehearsal frequencies measured in terms of mean square error (MSE), with P varied from 0 to 4. The MSE criterion can be viewed as (the square of) the distance between the target and predicted outputs for the B items. Catastrophic interference is evident when $P = 0$, which prohibits interleaving to occur, similar to that of traditional connectionist models with strict sequential learning. The advantage of interleaving is already clear with $P = 1$, yielding much more gradual forgetting. Further increasing P from 2 to 4 produces even lower interference levels, with the A-B association kept nearly intact. The same conclusion is obtained based on the recall accuracy of the A-B patterns after rehearsal in Figure 5.6(b), for $P = 0$ to 10. These results demonstrate the complementary roles of the SLM and FLM, and more generally the DCN consolidation procedure, in suppressing catastrophic interference, which contrasts with the sequential learning process in classical connectionist models. It is also worth noting that this procedure can be naturally applied to more than 2 episodes, possibly involving multiple, distinct experiences.

For further verifications, additional tests were conducted by introducing a new D-B association, also containing 20 stimulus-response pairs. Out of the 20 items $D^{(p)}$ in list D,



(a) Recall mean square error (b) Recall accuracy rate

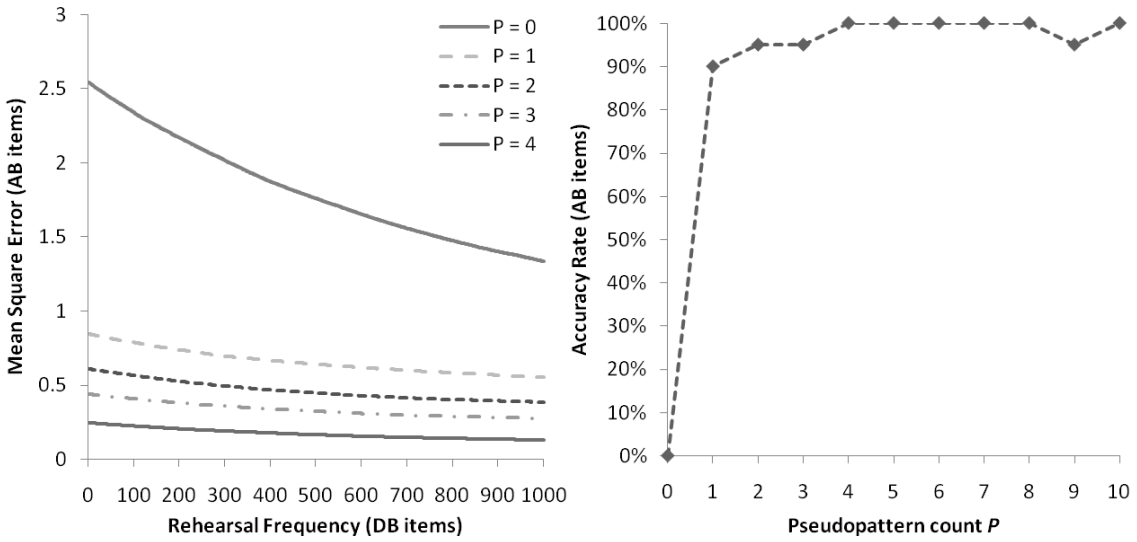
Figure 5.8: Recall of A-B items after learning A-B, D-B and A-C items



(a) Recall mean square error (b) Recall accuracy rate

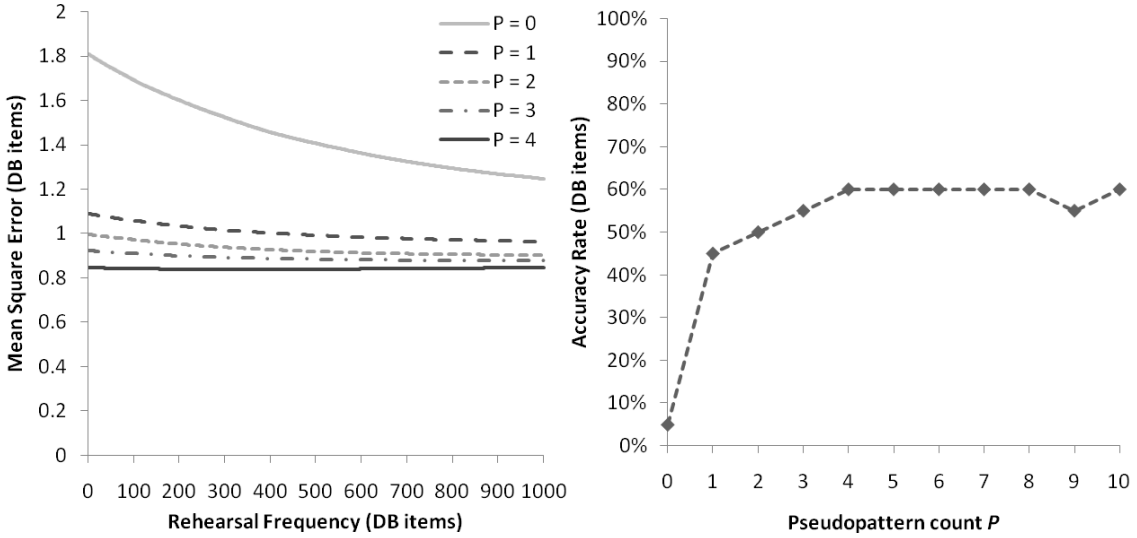
Figure 5.9: Recall of D-B items after learning A-B, D-B and A-C items

12 are taken from the 5-bit number combinations not found in list A, while the remaining 8 stem from a randomly chosen subset of list A. Consequently, the D-B association can be regarded as being compatible with the A-B association, in contrast to the A-C pairs. The incorporation of D-B patterns subsequently leads to three extended experimental scenarios. The first scenario involves learning of A-B association in episode 1, followed by that of D-B association in episode 2. The results using the same system configuration as in the previous experiment are shown in Figure 5.7. As observed, the presentation of D-B



(a) Recall mean square error (b) Recall accuracy rate

Figure 5.10: Recall of A-B items after learning A-B, A-C and D-B items



(a) Recall mean square error (b) Recall accuracy rate

Figure 5.11: Recall of D-B items after learning A-B, A-C and D-B items

patterns does not disrupt the existing SLM representation of A-B patterns regardless of the P setting, which can be attributed to the compatibility between the two associations. It must nevertheless be noted that this interference-free paradigm could not be immediately achieved in classical connectionist models whose network structures/topologies are fixed throughout the learning processes. This illustrates in turn the key advantage of the structural construction and reduction processes, which in the case of DCN consolidation procedure occur during the transient learning and structural reorganization steps.

The remaining two scenarios involve three-episode consolidation processes that constitute natural extensions of the two-episode cases previously discussed. Specifically, the second scenario is concerned with learning A-B, D-B and A-C patterns in sequence. The results are given in Figures 5.8 and 5.9, depicting the impacts of learning A-C patterns on the existing representations of A-B and D-B patterns, respectively. As with Figure 5.6, Figure 5.8 shows that pseudopattern transfer from the SLM to FLM (i.e., $P > 0$) plays an important role in mitigating catastrophic interference with the representation of previously learned patterns. It can also be seen that the presence of D-B patterns before learning A-C patterns imposes less interference with the representation of A-B patterns, which is expected given the mutual compatibility between the A-B and D-B patterns. With respect to the D-B patterns, Figure 5.9 again illustrates the benefit of $P > 0$ in maintaining the existing representation, although the recall accuracy of the association is not perfect (i.e., 55-70%). The latter is due to the limited FLM capacity (i.e., $C = 20$ only), in which case the FLM cannot completely capture both A-B and D-B associations (or their pseudopatterns) at the same time. Regardless, the importance of $P > 0$ is still clear, and increasing the FLM capacity should help improve the recall accuracy.

The last scenario considered in this study involves presenting the A-B, A-C and D-B associations in order. The recall performances of the A-B and D-B patterns are shown in Figures 5.10 and 5.11 respectively. From Figure 5.10(a), it can be observed that the recall MSE for the A-B association is gradually reduced as the rehearsal frequency for the D-B patterns is increased, indicating a restored recall of old association due to the presence of compatible patterns. It is also evident here that further improvements can be made by increasing P, allowing the representation of A-B patterns to be recovered much more quickly. Similarly, as illustrated in Figure 5.11(a), the learning speed and accuracy of D-B patterns are enhanced as P is increased. This result highlights the crucial role of interleaving in the context of *representational knowledge transfer* [26] (cf. section 5.1), whereby the existing representation can be exploited for the learning of a related task in the future. As with Figure 5.9(b), note again that the imperfect recall of D-B patterns in Figure 5.11(b) can be attributed to the limitation of the current FLM capacity.

Table 5.2: Parameter configurations of DCN for the experiments

Parameter	Translation Initiation Sites	Face Recognition
Maximum inputs D	5	3
Split threshold δ	0.6	0.4
Pseudopattern count P	5	5
Rehearsal frequency F	1-50	1-10

5.5 Experimental Studies

5.5.1 Simulation Setup

Simulation studies have been performed to evaluate the practical applicability of the DCN system in large-scale real-world tasks, the two most illustrative of which are reported here: the *translation initiation sites* (TIS) *prediction* [206] and *face image detection* [106]. The two tasks are particularly challenging due to the high-dimensional nature and large number of data patterns involved, which provide a suitable testbed to evaluate the DCN system as the key component in the INCA framework. For all experiments, the maximum FLM capacity C is set to be equal to the episode length N_e for simplicity, and the FLM membership threshold λ is fixed as 0.95 to foster sparse activation. The remaining parameters are chosen empirically for each case study, as per Table 5.2. Detailed results and analysis of the two case studies are presented in sections 5.5.2 and 5.5.3. Meanwhile, estimates of consolidation time of the proposed system can be found in Appendix C.

5.5.2 Translation Initiation Sites Prediction

This section presents simulation results on the difficult problem of identifying the *translation initiation sites* (TIS) in vertebrate genomic sequences i.e., the location at which the translation process from messenger RNA into protein is initiated. Accurate TIS identification is crucial for better understanding of the translation process, gene structure, and protein coding, and for reliable amino acid prediction [206]. This study serves to show the scalability and generalization traits of the DCN consolidation and inference procedures in dealing with large task domains, where most of the traditional computational methods cannot be directly applied due to the lack of system scalability.

The dataset used here contains $3,312$ true TIS (functional ATG) and $10,063$ false TIS

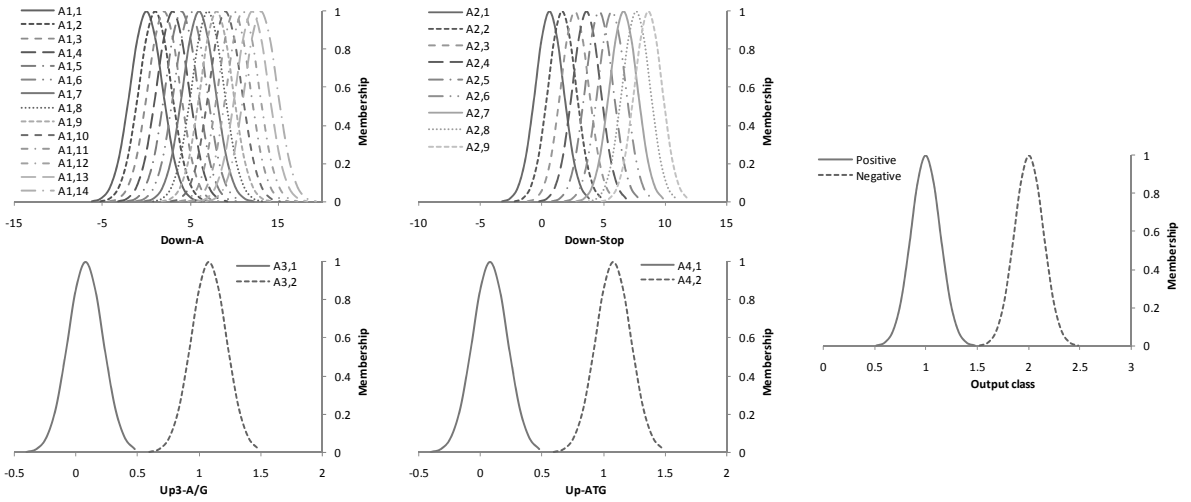


Figure 5.12: Fuzzy membership functions crafted in FLM for the TIS data (CV3)

Table 5.3: Sample fuzzy rules in FLM for the TIS task (CV3)

Rule	Down-A	Down-Stop	Up3-A/G	Up-ATG	Prediction
R_1	$A_{1,1}$	$A_{2,1}$	$A_{3,2}$	$A_{4,2}$	Negative
R_2	$A_{1,2}$	$A_{2,1}$	$A_{3,2}$	$A_{4,1}$	Positive
R_3	$A_{1,2}$	$A_{2,1}$	$A_{3,1}$	$A_{4,2}$	Negative
R_4	$A_{1,1}$	$A_{2,1}$	$A_{3,2}$	$A_{4,2}$	Negative
R_5	$A_{1,2}$	$A_{2,3}$	$A_{3,1}$	$A_{4,1}$	Negative
R_6	$A_{1,1}$	$A_{2,2}$	$A_{3,1}$	$A_{4,1}$	Negative
R_7	$A_{1,6}$	$A_{2,2}$	$A_{3,1}$	$A_{4,2}$	Negative
R_8	$A_{1,1}$	$A_{2,1}$	$A_{3,2}$	$A_{4,1}$	Positive
R_9	$A_{1,3}$	$A_{2,1}$	$A_{3,1}$	$A_{4,2}$	Negative
R_{10}	$A_{1,1}$	$A_{2,1}$	$A_{3,2}$	$A_{4,2}$	Negative
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

(non-functional ATG) [206]. A data point is generated for each potential start codon and captured by a sequence window of 200 nucleotides centered at the corresponding ATG triplet. The feature space for prediction is then built by matching 3 nucleotides to 1 amino acid and counting the frequency of the latter. Each amino acid is also classified as upstream or downstream based on its relative position to the centered ATG, giving 2×21 input features. The frequency of a pair of amino acids is counted as well, yielding 2×21^2 more features. Finally, 3 Boolean features are added: Down4-G (if G occurs at position +4), Up3-A/G (if A or G occurs at position -3), and Up-ATG (if ATG occurs in the upstream). In total, these give 927 features. Evaluation is then done via stratified 3-fold *cross-validation* (CV) i.e., partitioning the data into 3 groups of mutually exclusive training and test sets (CV1-CV3), each retaining the original TIS class proportion.

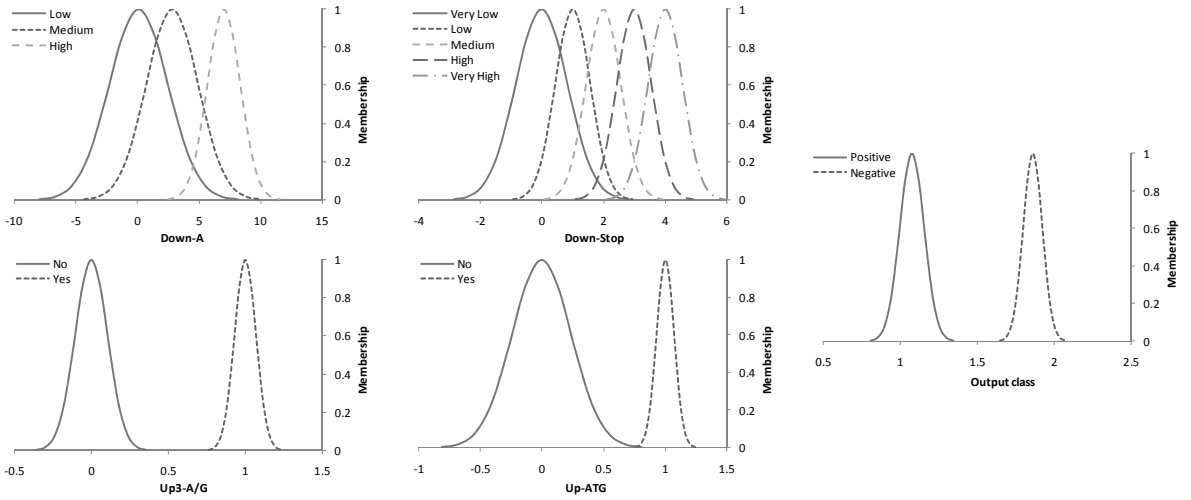


Figure 5.13: Fuzzy membership functions crafted in SLM for the TIS data (CV3)

Table 5.4: Initial fuzzy rules in SLM for the TIS task (CV3)

Rule	Down-A	Down-Stop	Up3-A/G	Up-ATG	Prediction
R_1	Medium	Very Low	Yes	Yes	Negative
R_2	Low	Very Low	Yes	No	Positive
R_3	Low	Very Low	No	Yes	Negative
R_4	Medium	Medium	No	No	Negative
R_5	Medium	Low	No	No	Negative
⋮	⋮	⋮	⋮	⋮	⋮
R_{20}	Medium	Very Low	No	No	Positive
R_{21}	Medium	Very High	No	Yes	Negative
R_{22}	Medium	Very Low	Yes	No	Positive
R_{23}	Low	Low	Yes	Yes	Negative
R_{24}	High	Very Low	Yes	No	Positive
⋮	⋮	⋮	⋮	⋮	⋮
R_{52}	High	Medium	No	Yes	Negative
R_{53}	High	Medium	Yes	Yes	Negative
R_{54}	High	Low	Yes	No	Negative
R_{55}	High	Medium	Yes	No	Negative
R_{56}	High	High	No	Yes	Negative

The first set of experiments conducted on the TIS data consists of carrying out the DCN consolidation procedure in a single episode scenario. In this, consolidation begins with the *attention focus* process building the attention layer, capturing the most informative input features for the TIS prediction. Four features were chosen: Down-A and Down-Stop (frequencies of A and stop codon in the downstream, respectively), Up3-A/G and Up-ATG, that are consistent with biological findings [206]. Next, FLM performs *transient learning* atop the attended feature space to encode the TIS data, with its max-

Table 5.5: Reorganized fuzzy rules in SLM for the TIS task (CV3)

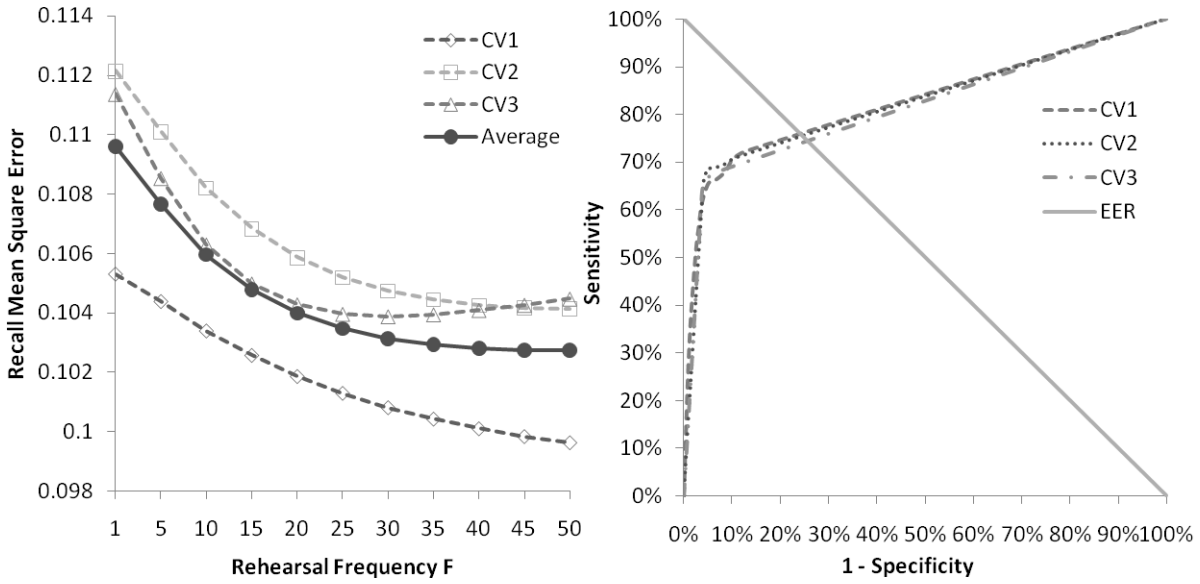
Rule	Down-A	Down-Stop	Up3-A/G	Up-ATG	Prediction
R_1	-	-	-	Yes	Negative
R_2	-	Very Low	Yes	No	Positive
R_3	-	Low	-	-	Negative
R_4	-	Medium	-	-	Negative
R_5	Low	-	No	-	Negative
R_6	Medium	Very Low	-	No	Positive
R_7	High	-	No	-	Negative
R_8	-	High	-	-	Negative
R_9	-	Very High	-	-	Negative

- = don't care

imum capacity set based on the data size as $C = N = 3,312 + 10,063 = 13,375$ rules. The set of antecedent and consequent labels constructed in the FLM for CV3 is depicted in Figure 5.12, and some samples of (episodic) rules crafted in the FLM are given in Table 5.3. Subsequently, *structural reorganization* step takes place based on the pseudopatterns reinstated from the FLM, which begins with the *label construction* and *reduction* steps to construct the antecedent and consequent labels in the SLM. The set of SLM antecedent and consequent labels identified for CV3 is shown in Figure 5.13.

Rule construction is then carried out to form the initial set of rules linking the antecedent and consequent labels. An example of the initial SLM rule base created in CV3 is shown in Table 5.4. Notably, as FLM captures similar patterns only once, the pseudopatterns generated randomly (with equal likelihood) from its rules would have a balancing effect when rehearsed into the SLM, which helps deal with the skewed TIS class distribution. One may observe that the 56 rules in Table 5.4 have many inconsequential antecedent links. This redundancy is examined next via the *rule reduction* step. For each input feature, consistent rules are identified which, when their antecedent links to the current feature are omitted, will have the same remaining antecedent and consequent labels. These rules are then removed to reduce the rule base, thereby improving system interpretability and generalization. Table 5.5 shows the resultant 9 rules only.

Next, the *interleaved learning* step is performed by transferring the pseudopatterns generated from the FLM into SLM. To measure the effects of the pseudorehearsal process on the DCN recall performance, experiments were conducted with different rehearsal



(a) Recall mean square error (b) Receiver operating characteristic plot

Figure 5.14: Performances of DCN for the TIS task in a single episode scenario

frequencies F, from 1 to 50. The results, summarized in Figure 5.14(a), indicate that increasing F generally leads to better recall of the actual TIS data, though with (linearly) increased learning time. Note that higher F may yield higher error (e.g. in CV3), because the pseudopatterns reinstated from the FLM may differ from the actual patterns.

Exploitation of the consolidated knowledge about TIS domain begins by presenting a test data point, comprising the complete 927 features, to the DCN input layer. Attention focus then takes place in the attention layer (built via the DCN consolidation procedure) to filter the inputs down to 4 features (i.e., Down-A, Down-Stop, Up3-A/G and Up-ATG). Next is the activation of those rules in SLM and FLM whose antecedents cover the 4 attended features, via the inference procedures described in (Eq. 5.11) and (Eq. 5.12). The outputs inferred by the two modules are then aggregated in the combination layer via (Eq. 5.10) to compute the final consensus about TIS prediction.

The generalization performances produced by the DCN inference procedures is shown in the *receiver operating characteristic* (ROC) chart [66] of Figure 5.14(b), which exemplifies the system's robustness and discriminative power on all test cases. The plot was obtained by varying a classification/decision threshold and measuring for each threshold value the *sensitivity* and *specificity* rates i.e., proportions of true and false TIS cases that

Table 5.6: Comparative results of various methods for the TIS data

System	#Features	#Rules	Accuracy	Sensitivity	Specificity	Precision	MCC
Naïve Bayes	All	-	86.12%	79.65%	88.24%	69.04%	0.6487
SVM (Linear kernel)	All	-	90.65%	80.56%	93.98%	81.49%	0.7482
Voted Perceptron	All	-	86.00%	68.93%	91.61%	73.01%	0.6174
C4.5 tree	40	517	87.58%	72.55%	92.53%	76.16%	0.6617
SLM (RFCMAC)	4.67	14	87.89%	64.64%	95.54%	82.66%	0.6574
DCN ($F = 50$)	4.67	19	88.22%	66.88%	95.24%	82.22%	0.6685

- = not applicable

are correctly classified, respectively. They are defined in (Eq. 5.15)-(Eq. 5.16)

$$Sensitivity = \frac{TP}{TP + FN} \quad \text{(Eq. 5.15)}$$

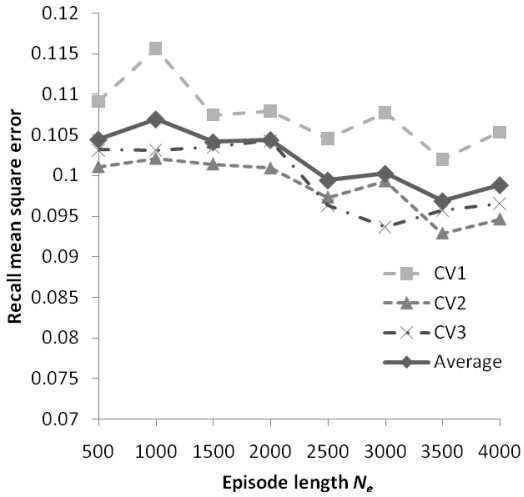
$$Specificity = \frac{TN}{FP + TN} \quad \text{(Eq. 5.16)}$$

where TP (FP) is the number of true (false) TIS cases, and TN (FN) the number of true (false) non-TIS cases. Larger area under the ROC curve implies better generalization, and EER is the *equal error rate* when specificity is equal to sensitivity. Two other related criteria not shown in the ROC chart but later used for comparison are the *precision* rate, i.e., proportion of correctly predicted positive cases, and *Matthew's Correlation Coefficient* (MCC) $\in [-1, 1]$ [161], as respectively defined in (Eq. 5.17)-(Eq. 5.18)

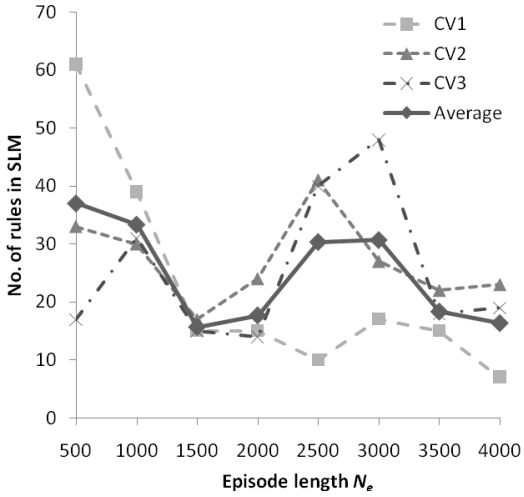
$$Precision = \frac{TP}{TP + FP} \quad \text{(Eq. 5.17)}$$

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad \text{(Eq. 5.18)}$$

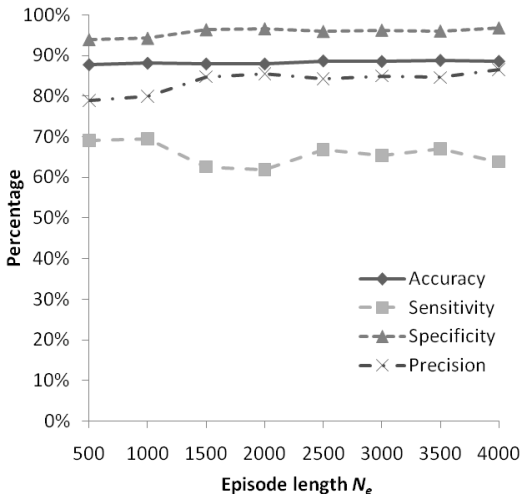
To provide a quality check for the DCN system, comparisons were made with several established and scalable techniques: Naïve Bayes [52], Support Vector Machine (SVM) with fast linear kernel implementation [65], Voted Perceptron [76], and C4.5 decision tree [216], all trained on a single-episode basis and using the 3-fold CV method. Comparison is also made between DCN and the previous standalone SLM implementation (i.e., RFCMAC), which is trained directly on the TIS data without going through the FLM. Table 5.6 shows the results averaged across 3 CVs in terms of accuracy, sensitivity, specificity, precision, MCC and the numbers of rules crafted and features selected. As seen, DCN exhibits good overall prediction performances, albeit having rather low sensitivity. While SVM performs well in terms of accuracy, sensitivity and MCC, it produces a black-box



(a) Recall mean square error



(b) Number of rules in SLM



(c) Generalization performances

Figure 5.15: Performances of DCN for the TIS task in multi-episode scenario

structure that cannot be qualitatively evaluated and interpreted, in contrast to DCN and RFCMAC. Meanwhile, compared to C4.5 tree, DCN yields a much smaller rule base

and thus superior interpretability. Its accuracy, sensitivity and MCC are also shown to improve those of the RFCMAC alone, thanks to the pattern balancing effect produced by the pseudopatterns from the FLM, though giving slightly larger rule base.

The second set of experiments conducted on the TIS data involves evaluating the robustness of the DCN system under multi-episode consolidation scenarios. The results obtained for different episode lengths $N_e = \{500, 1000, 1500, 2000, 2500, 3000, 3500, 4000\}$ are summarized in Figure 5.15. Figure 5.15(a) shows the recall performances of the DCN, measured using MSE with respect to the training patterns in each CV. As can be seen, the overall recall MSE tends to decrease as the episode length is increased. This suggests that better recall can be obtained by longer learning episode, at the expense of longer computational time and higher memory requirement, particularly those associated with the FLM size complexity that grows with the episode length (since $C = N_e$).

On the other hand, it is shown in Figure 5.15(b) that fluctuation occurs in the number of rules crafted in the SLM for different N_e. Nevertheless, the overall rule base complexity is relatively small (i.e., less than 40 rules on average). Experiments also indicate that the rule variation does not seem to greatly affect the system's generalization performances, as portrayed by the fairly stable (average) accuracy, specificity, sensitivity and precision rates in Figure 5.15(c). It can thus be concluded that the DCN recall and generalization performances are relatively insensitive to the episode length (and so the FLM capacity), which may again be attributed to the pseudopattern-based interaction between the SLM and FLM. Exploiting this feature, one can set the episode length according to the available computational budget, without significantly impacting the system's performances. Altogether, these results demonstrate the computational efficacy of the DCN system, in terms of scalability, interpretability, and generalization traits.

5.5.3 Face Image Detection

In this section, face detection experiment is conducted employing the database developed at the Center for Biological and Computational Learning (CBCL), Massachusetts Institute of Technology [256, 106]. The main objective is to investigate the ability of the DCN system to scale up against, and incrementally consolidate knowledge acquired from,

Table 5.7: The original and extended CBCL datasets used for the experiments

	Training set			Testing set		
	Face	Non-face	Total	Face	Non-face	Total
Original	2,429	4,548	6,977	472	23,573	24,045
Extended	367,416	471,914	839,330	472	23,573	24,045

large object recognition task domain involving both high-dimensional feature space and huge number of data patterns. Face detection is a basic cognitive task which occurs in our everyday life and has been extensively studied in computer vision research. It poses a variety of interesting applications, for instance, as part of a face recognition system, a video surveillance system, or a human-computer interface. Faces also provide a class of visually similar objects, allowing to simplify the difficult task of object detection.

The original dataset in [256] consists of $6,977$ grayscale training images (with $2,429$ faces and $4,548$ non-faces) and $24,045$ test images (472 faces and $23,573$ non-faces). The size of each image is 19×19 pixels. In [256], three preprocessing steps were performed to extract the image features in a way that minimizes the variations between objects of the same class (i.e., face or non-face). First, pixels located near to the boundary of the images were omitted to exclude parts associated with the background. A best-fit intensity plane was then subtracted from the gray values to compensate for cast shadows. Lastly, histogram equalization was applied to remove variations in the brightness and contrast. Examples of histogram-equalized face and non-face images (randomly taken from the dataset) are depicted in Figure 5.16. In [106], extended experiments were made by computing the gray value gradients and Haar wavelets. The gray, gray gradient and Haar wavelet features were subsequently compared by feeding them as inputs to an SVM classifier. The results had shown that the histogram-equalized gray values offer the most discriminative features for face detection. This accordingly provides the motivation for (directly) employing the gray values in the current experiment with the DCN system.

In order to examine the scaling properties of the DCN system, the original training dataset was enlarged by generating artificial samples, following the procedure in [106, 273]. That is, additional non-faces were extracted from images that do not contain faces, such as landscapes, trees and buildings, while new faces were obtained by applying various

(a) Face (b) Non-face

Figure 5.16: Examples of face and non-face images in the CBCL database

image transformations to the original faces. Specifically, 477, 366 non-face images that originated from 100 photos collected from the web were added, yielding 481, 914 non-faces. Subsequently, 242 faces and 10, 000 non-faces were taken out to be a validation set (for model selection purposes), leaving 2, 187 faces and 471, 914 non-faces. Next, image transformations were carried out on each face by first blurring via arithmetic mean filter, employing window sizes of 2×2, 3×3, and 4×4, respectively. The results were in turn flipped laterally, producing $2,187 \times 4 \times 2 = 17,496$ faces. Finally, each face was rotated between $-20°$ to $20°$, with increment of $2°$. This gives a total of $17,496 \times 21 = 367,416$ faces. Together with the non-faces, the overall extended training dataset contains 839, 330 images. A summary of the datasets used in this study is presented in Table 5.7.

Experiments were first conducted on the original datasets to provide an initial quality check for DCN in a single episode scenario. The membership functions of the fuzzy labels crafted in the SLM are depicted in Figure 5.17, and their corresponding fuzzy rules are enumerated in Table 5.8. It was found that certain pixel coordinates (x, y) are more informative than the others. In particular, the DCN system selected three coordinates $(8, 10)$, $(9, 4)$ and $(9, 17)$ to discriminate between the faces and non-faces, as per Figure 5.17. The first coordinate corresponds to the region around the tip of the nose in the face, while the last two around the centers of the left and right cheeks, respectively. The mapping between the SLM rules in Table 5.8 and the image data is illustrated in Figure

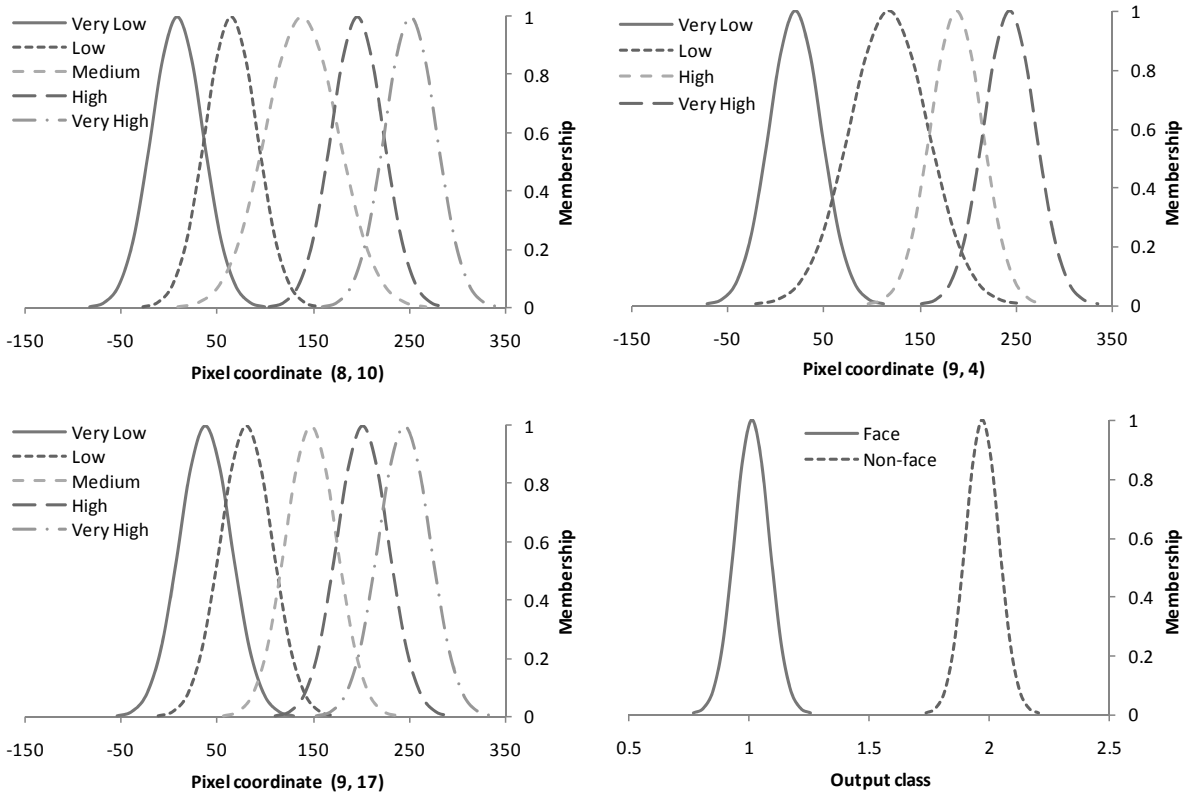


Figure 5.17: Fuzzy membership functions crafted in SLM for the original CBCL data

5.18. As seen, the rules in Table 5.8 are fairly compact and highly intuitive, suggesting that the bright pixels (around the nose and cheeks) generally contain more information about the faces, whereas the dark pixels correspond more closely to the non-faces.

The effects of the pseudorehearsal process in DCN on its recall performance, employing different rehearsal frequencies F from 1 to 10, are illustrated in Figure 5.19(a). As in the TIS experiment, the plotted result indicates that increasing F generally leads to better recall of the actual TIS data, although it imposes a longer consolidation time. The generalization performance of the system on the (original) test data, on the other hand, is summarized by the ROC chart in Figure 5.19(b). In essence, the relatively symmetric curve (with reference to the EER line) exemplifies a balanced prediction of the face and non-face classes, hence ratifying the system's robustness and discriminative power.

For performance comparisons, experiments were conducted using the classification techniques mentioned previously (in the TIS experiment) i.e., the Naïve Bayes [52], fast linear kernel Support Vector Machine (SVM) [65], Voted Perceptron [76], C4.5 decision tree [216], and the standalone SLM implementation (RFCMAC). It must be noted that,

Table 5.8: Fuzzy rules in SLM for the original CBCL data

Rule	Pixel $(8, 10)$	Pixel $(9, 4)$	Pixel $(9, 17)$	Prediction
R_1	Low	-	-	Non-face
R_2	High	-	Medium	Non-face
R_3	-	-	Very Low	Non-face
R_4	Very High	Very High	Very High	Face
R_5	Medium	Low	-	Non-face
R_6	-	Very Low	-	Non-face
R_7	High	High	Very High	Face
R_8	-	Low	High	Non-face
R_9	Very High	High	Very High	Face
R_{10}	High	Very High	Very High	Face
R_{11}	-	-	Low	Non-face
R_{12}	Very High	Very High	Medium	Face
R_{13}	Medium	High	-	Non-face
R_{14}	Very Low	-	-	Non-face
R_{15}	High	Low	-	Non-face
R_{16}	Medium	-	Medium	Non-face
R_{17}	Very High	High	High	Face
R_{18}	High	High	High	Non-face
R_{19}	Very High	Very High	High	Face
R_{20}	Very High	Low	Very High	Face
R_{21}	High	Very High	High	Face
R_{22}	Very High	High	Medium	Face
R_{23}	-	Low	Medium	Non-face
R_{24}	Medium	Very High	Very High	Face
R_{25}	Medium	Very High	High	Face

- = don't care

given the highly imbalanced nature of the test set (i.e., the face cases constitute only $\frac{472}{23{,}573} \approx 2\%$ of the entire set), the use of some evaluation criteria such as accuracy, specificity and precision is not appropriate. To better reflect the generalization performance, the *balanced accuracy* and *area under the curve* (AUC) [66] metrics popularly used for face detectors are adopted. The balanced accuracy Acc_{bal} is defined in (Eq. 5.19)

$$Acc_{bal} = \frac{1}{2}\left(\frac{TP}{TP+FN} + \frac{TN}{FP+TN}\right) \qquad \text{(Eq. 5.19)}$$

while the AUC is computed by integrating over the horizontal axis of the ROC curve. In this, the face and non-face cases are treated as positives and negatives, respectively. The AUC is defined within a unit interval $[0, 1]$, whereby a perfect predictor will have a unit AUC whereas random guessing will give an AUC of 0.5.

The consolidated generalization performances of the methods are given in Table 5.9.

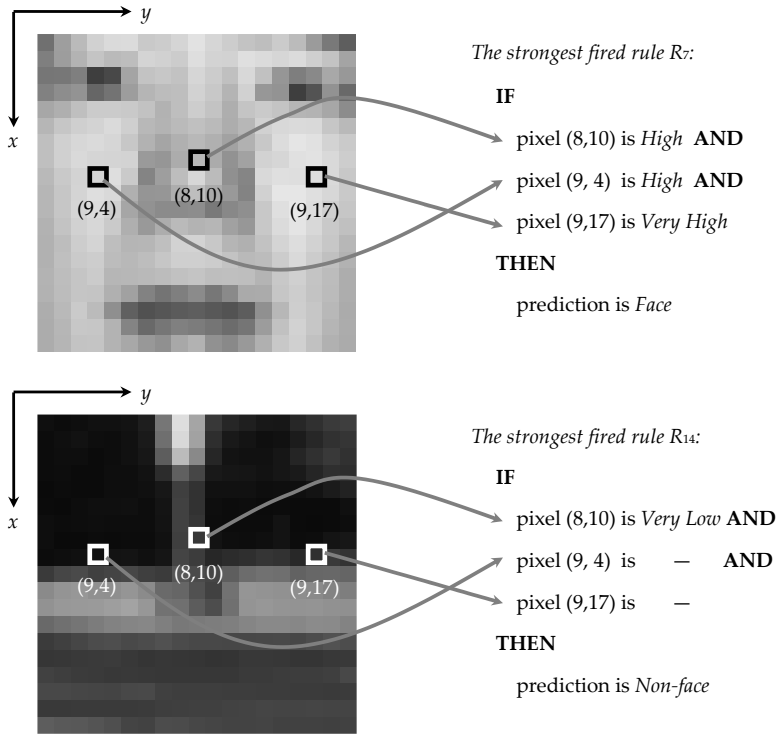
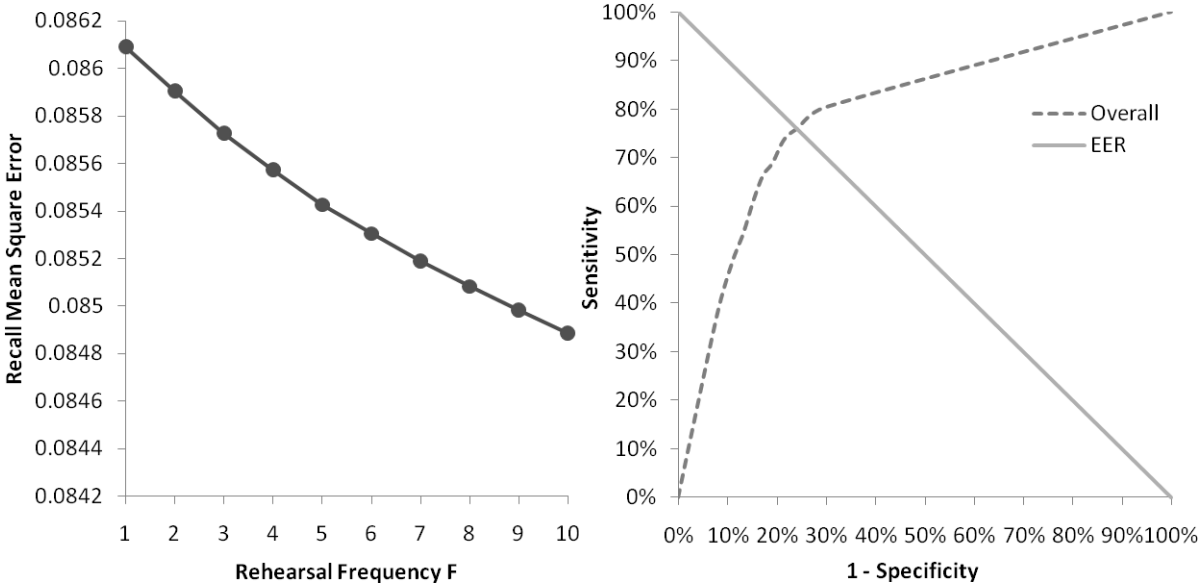


Figure 5.18: Interpretation of fuzzy rules in SLM with respect to the image data



(a) Recall mean square error (b) Receiver operating characteristic plot

Figure 5.19: Performances of DCN for the original CBCL data

It can be noticed that in general the DCN and SLM systems outperforms the remaining methods. This may be largely attributed to the rule base reduction processes (i.e., the attention focus and structural reorganization steps in DCN), which allow to automatically identify compact yet accurate set of features and rules. It is also evident that the two

Table 5.9: Comparative results of various methods for the original CBCL data

System	#Features	#Rules	Acc_{bal}	Sensitivity	MCC	AUC
Naïve Bayes	All	-	69.15%	45.13%	0.201	0.775
SVM (Linear kernel)	All	-	56.84%	15.68%	0.127	0.568
Voted Perceptron	All	-	60.21%	22.67%	0.176	0.634
C4.5 tree	18	121	60.94%	27.54%	0.127	0.577
SLM (RFCMAC)	3	25	73.69%	61.86%	0.182	0.794
DCN ($F = 10$)	3	25	74.55%	66.31%	0.177	0.791

- = not applicable

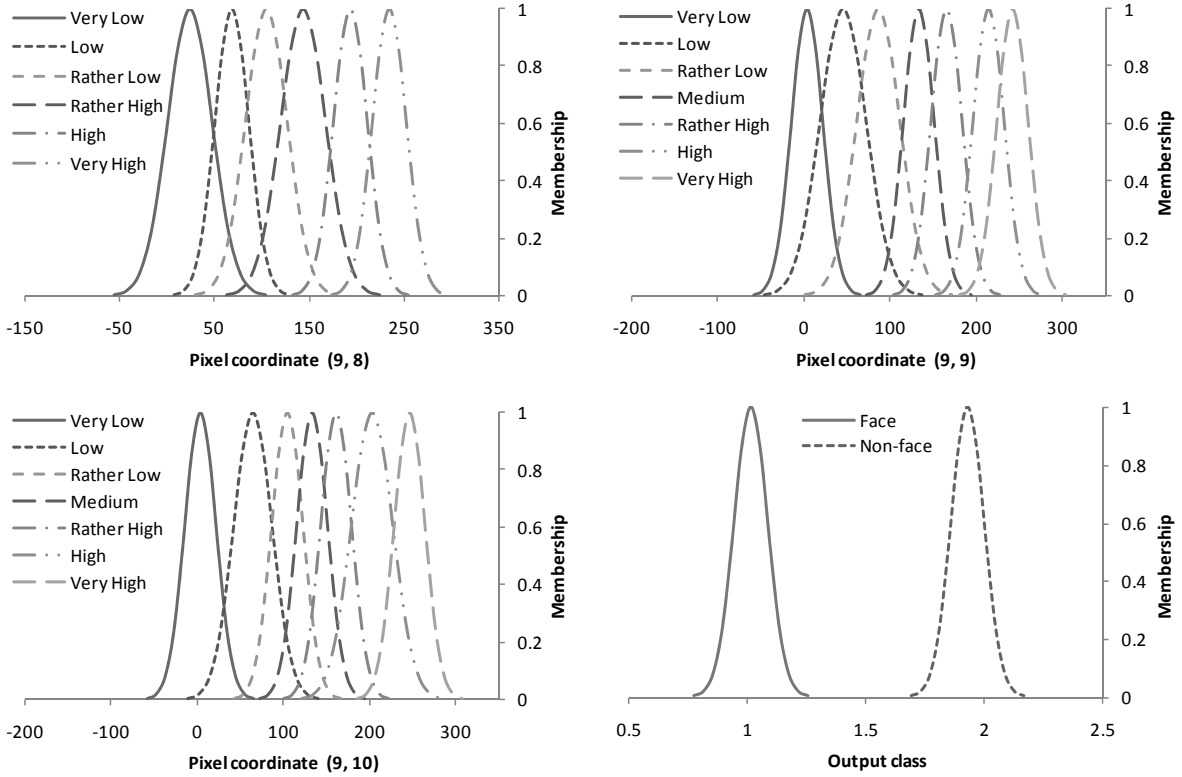


Figure 5.20: Fuzzy membership functions crafted in SLM for the extended CBCL data

systems yield a smaller rule base, and hence better interpretability, than the C4.5 tree. Comparing DCN and SLM, the former is shown to produce better balanced accuracy and sensitivity (thanks to the pattern balancing effect produced by the FLM rehearsal), while their MCC and AUC are comparable. Note here that the low MCC values obtained by all methods are due to the strong bias induced by the majority (non-face) class[2].

The scaling behavior of the DCN system is subsequently evaluated using the extended training set. Clearly, given the ultra-large number of patterns in this dataset, performing

[2]From (Eq. 5.18), it can be deduced that when TN is very high, MCC $\approx \frac{TP}{\sqrt{(TP+FP)(TP+FN)}} = \sqrt{\text{Precision} \times \text{Sensitivity}}$, which tends to be a low value.

Table 5.10: Fuzzy rules in SLM for the extended CBCL data

Rule	Pixel $(9,8)$	Pixel $(9,9)$	Pixel $(9,10)$	Prediction
R_1	-	Medium	-	Non-face
R_2	-	Low	-	Non-face
R_3	Very High	Very High	Very High	Face
R_4	-	Rather High	-	Non-face
R_5	-	Rather Low	-	Non-face
R_6	High	High	Very High	Face
R_7	Rather High	High	Very High	Face
R_8	High	High	High	Face
R_9	Very High	Very High	High	Face
R_{10}	High	Very High	Very High	Face
R_{11}	-	Very Low	-	Non-face
R_{12}	Rather High	-	Rather High	Non-face
R_{13}	Very High	High	Rather High	Face
R_{14}	High	High	Rather High	Face
R_{15}	Very Low	-	-	Non-face
R_{16}	Rather Low	-	-	Non-face
R_{17}	High	-	Medium	Non-face
R_{18}	Very High	High	Medium	Face
R_{19}	High	-	Rather Low	Non-face
R_{20}	Very High	High	High	Face
R_{21}	-	-	Very Low	Non-face
R_{22}	-	-	Low	Non-face
R_{23}	Rather High	Very High	-	Non-face
R_{24}	Very High	Very High	Rather High	Face
R_{25}	Very High	High	Rather Low	Face
R_{26}	High	Very High	High	Face
R_{27}	Low	-	-	Non-face
R_{28}	Rather High	-	Rather Low	Non-face
R_{29}	High	Very High	Rather High	Non-face
R_{30}	Rather High	-	Medium	Non-face
R_{31}	Very High	High	Very High	Face
R_{32}	-	Very High	Rather Low	Non-face
R_{33}	Very High	Very High	Medium	Face
R_{34}	Rather High	-	High	Non-face

- = don't care

a single-episode consolidation is not feasible, which would involve high space and time requirements. Accordingly, multi-episode consolidation is carried out, where the episode length and rehearsal frequency are respectively set as $N_e = 10000$ and $F = 10$. The fuzzy labels and rules constructed in the SLM for this experiment are shown in Figure 5.20 and Table 5.10, respectively. In this case, the DCN system selected a slightly different set of pixel coordinates $(9,8)$, $(9,9)$, $(9,10)$ to detect the faces, which essentially correspond to

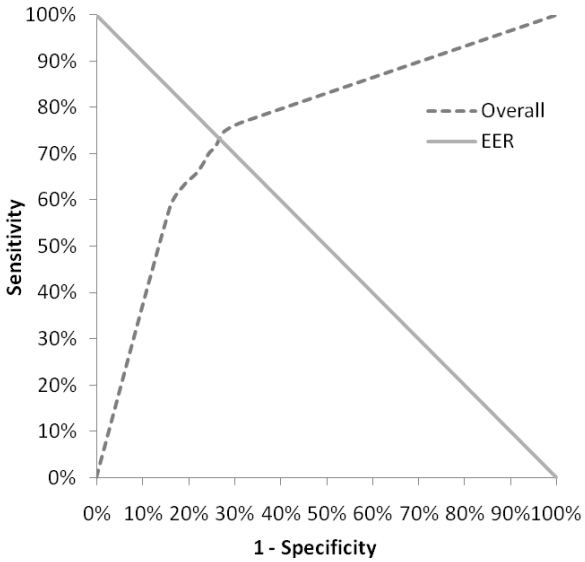


Figure 5.21: Generalization performance of DCN for the extended CBCL data

Table 5.11: Consolidated results of the DCN system for the extended CBCL data

System	#Features	#Rules	Acc_{bal}	Sensitivity	MCC	AUC
DCN ($F = 10$)	3	34	72.31%	67.80%	0.145	0.759

the center image region (i.e., around the tip of nose). It is also shown that the size of the rule base crafted for the extended training set is slightly more complex than that for the original set. Regardless, the two rule bases are similar, providing the rational explanation that the region around the nose tip is informative, and that bright intensities in some selected pixels can generally be associated with the face category (and vice versa).

The generalization capacity of the DCN system on the test set is showcased by the ROC plot in Figure 5.21. As observed, the fairly symmetric curve gained in this experiment is similar to the previous curve obtained for the original test set, indicating a good, balanced discrimination between the face and non-face cases. A summary of DCN's generalization performance is subsequently given in Table 5.11. Compared to the results in Table 5.9, a slight drop was observed in the overall performance that can be attributed to the introduction of artificial and transformed images in the extended training data. For this scenario, no benchmarking is performed with the other techniques listed in Table 5.9, either due to insufficient memory or because their training time is too long. It is also worth noting that to the best of our knowledge no neural network model, particularly that of memory consolidation [222, 75, 18, 200, 117], has been applied to problem of this

scale. In a nutshell, these results provide evidence for the scalability of the DCN system, and hence its computational efficacy to deal with large tasks.

5.6 Summary

A novel dual neuro-fuzzy system (NFS) termed *Dual Consolidation Network* (DCN) is put forward in this chapter, which attempts to model the complementary interactions between the hippocampal and cortical systems in the brain supporting robust consolidation of knowledge in humans. The effectiveness of the proposed system to efficiently cope with large-scale sequential tasks, while minimizing catastrophic interference, is encouraging. This justifies the possibility for applying the DCN system to model the acquisition of cognitive skills or concepts in more novel task domains, such as emulating the human driving expertise to create automation technologies for intelligent transportation systems [203, 202]. By extension, it is expected that employing the DCN as the core component in the INCA framework would open the way to build robust and scalable self-organizing mechanisms as well as higher-level cognitive abilities required for general intelligence.

Chapter 6

Concluding Remarks

6.1 Major Accomplishments

The ever-increasing complexity of today's systems and devices highlights the need for natural communication and human-like learning and cognitive abilities, hence for a more general machine intelligence. Research on general intelligence is now growing and gaining significant attention from the AI and cognitive sciences communities, with promising supporting computational models and architectures being the object of intense investigations. It is clear here that cognitive architectures will play a central role in providing a blueprint for constructing general intelligent systems that support a broad range of capabilities matching those of humans. With respect to this goal, the major accomplishments made by the current research are summarized in Figure 6.1 and reiterated as follows:

- The first contribution of this thesis comprises a survey of contemporary cognitive architectures in chapter 2, which provides a systematic categorization of various design approaches (i.e., symbolic, emergent and hybrid). Critical evaluations on several representative cognitive architectures have been presented, highlighting their salient merits and limitations. Pointers for future development of cognitive architectures were subsequently provided, which consist of several key milestones and directions crucial for the roadmap to general machine intelligence. In this, particular emphasis was given on the importance of three aspects: *knowledge consolidation*, *scalability trait*, and *metacognitive functions*, in addressing the shortcomings of the existing approaches and in creating better cognitive architectures or systems.

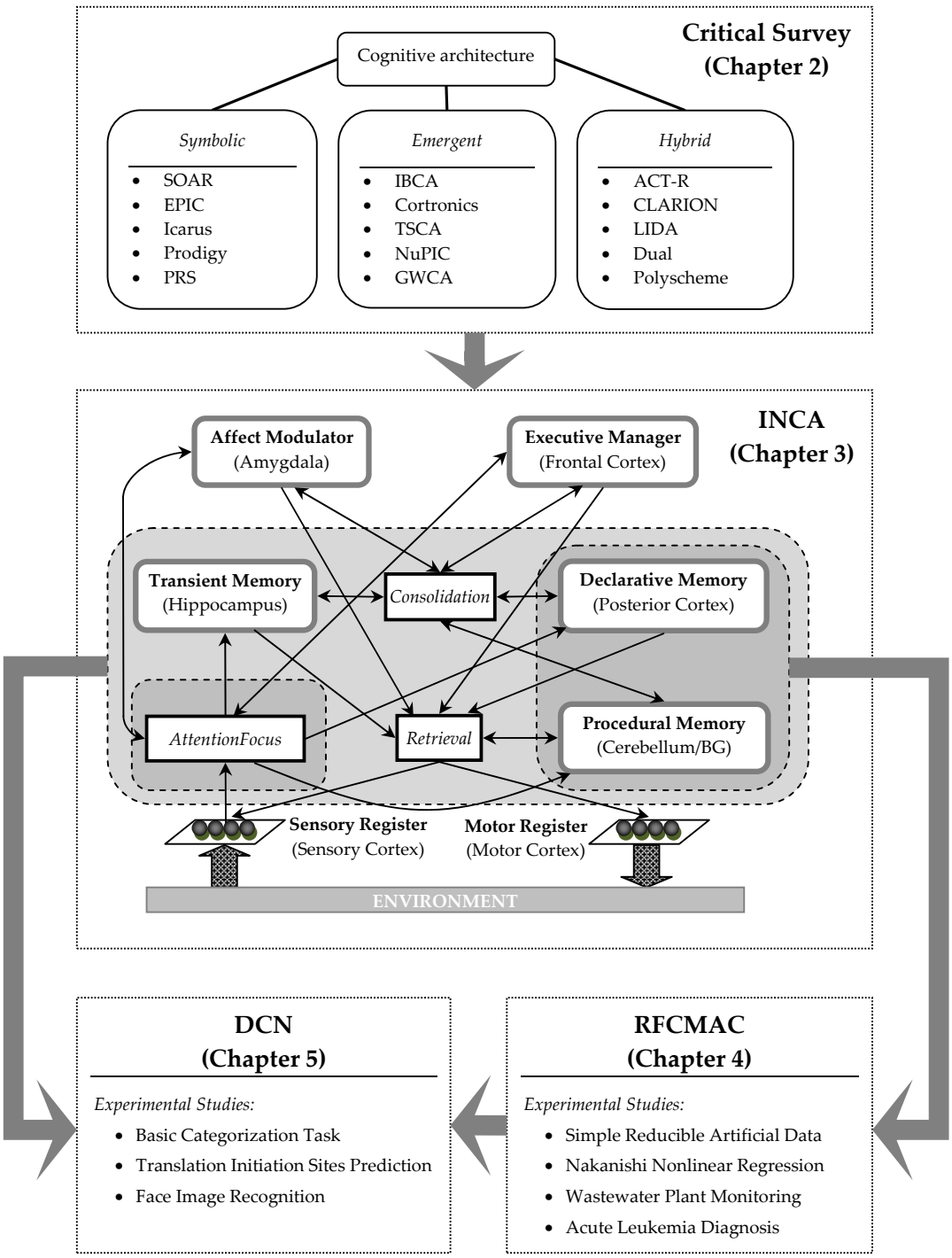


Figure 6.1: Major accomplishments of the current research

- Aiming at developing a comprehensive framework for general machine intelligence and in particular for building support for knowledge consolidation, scalability, and metacognition, a novel *Integrated Neuro-Cognitive Architecture* (INCA) has been formulated in chapter 3, which draws inspirations from the established functional aspects of the prominent brain circuitries and their interactions. Investigation on

the chief neuroscientific and psychological findings about the aspects of learning and memory in the brain as well as review on related computational models that capture (some of) these aspects have been made. Building upon these ideas, an outline of the INCA framework was proposed that employs *neuro-fuzzy system* (NFS) as a primary modeling approach to realize the constituent architectural modules. To systematically guide and regulate the interaction among different modules, two interaction procedures were developed, namely *consolidation* and *inference cycles*, which give a unique flavor to the INCA framework with respect to the state-of-the-art cognitive architectures. Evaluation on the cognitive plausibility of INCA has also been performed, and its key design features and capabilities have been compared with other prominent cognitive architectures.

- As an initial step in the development of INCA, particularly for realizing its long-term (declarative, procedural, executive) memory modules, a reduced rule-based localized NFS termed the *Reduced Fuzzy Cerebellar Model Articulation Controller* (RFCMAC) has been proposed in chapter 4. The model emulates the two-stage neural development of cortical memories in the brain to construct and reduce memory representation, respectively. The idea is realized in both label generation and rule generation phases of the RFCMAC learning process to derive a compact and representative rule base structure, prior to an iterative parameter tuning phase. The incorporation of reduction mechanisms provides RFCMAC with several benefits, including discovery of highly concise and intuitive rules, satisfactory generalization performances, and enhanced system scalability. A series of experiments, from using simple reducible data and nonlinear regression benchmark to complex water plant monitoring and high-dimensional leukemia diagnosis tasks, have demonstrated the efficacy of the model as a novel knowledge extraction tool.

- In extension, to provide a basic infrastructure supporting the consolidation and inference cycles in INCA, a novel *Dual Consolidation Network* (DCN) has been formulated in chapter 5 that models the complementary interactions between the hippocampal and cortical systems in the brain to consolidate and exploit knowl-

edge effectively. The proposed DCN consists of a slow-learning module (SLM) and a fast-learning module (FLM), modeling the cortical and hippocampal circuits in the brain respectively. The SLM is implemented using the RFCMAC model described in chapter 4, while the FLM comprises a transient NFS model that encodes distinct events using a sparse, non-overlapping representation. Knowledge consolidation is subsequently accomplished via a multi-episode complementary learning between SLM and FLM, using pseudopatterns as primary means for inter-module transfer of knowledge. This mechanism enables the system to robustly learn in a fully sequential manner (without recourse to the original patterns) while minimizing interference due to the dynamics of the environment. Experiments on basic categorization task and two large-scale problem domains have affirmed the knowledge consolidation, scalability, and generalization features of the proposed model.

6.2 Future Research

The INCA project is ongoing, and specific milestones would be needed to guide its development. Accordingly, several recommendations for its future research are given below:

- With reference to the DCN model, there are a number of possible improvements that can be explored to further extend its capabilities. For instance, the *attention focus* step in the DCN consolidation procedure currently yields a set of (input) features that remains fixed during the course of consolidation. Employing a dynamic, online feature selection or combination method to produce an evolving set of features across episodes, and investigating how this can be handled by the system without causing catastrophic interference, would be an interesting research direction. Another potential extension is the *parallelization* of the consolidation procedure, thereby allowing to assess the capacity of the DCN system in the face of non-ideal situations, e.g. when an episode is followed immediately by another episode, or when the system operates under limited computational budget.

- In relation to the INCA consolidation cycle, the role of the *affect modulator* (amygdala analogue) in providing goal or intention signals to modulate the consolidation

process has not been directly established in the DCN model. This can be realized by considering the role of the amygdala as a critic system that receives inputs from the major sensory systems and higher-order association areas of the cortex, and projects its outputs to various brain systems, particularly the frontal cortex and brainstem region [19, 199]. The transient NFS model described in chapter 5 can be used to model such function, ultimately to critique the potential reward of the actions taken by the other INCA modules, with the resulting internal modulation shaping future actions to be more rewarding. This would allow INCA to control its internal operations in a more strategic, task-appropriate manner.

- Building upon the above proposal, another promising milestone in the development of INCA is to realize the *reflective learning* and *control* mechanisms involving the interaction between executive manager and affect modulator, thereby providing INCA with powerful metacognitive capacities. These are particularly important for deploying, optimizing and comparing different (sub)modules in INCA with different inductive biases for a given task, thus enabling the system to monitor/intervene its internal operations and bootstrap its performance autonomously. Such mechanism would also help address the "no free lunch" issue in model optimization [290] and potentially lead to simpler models that exhibit good performances. Our initial effort in this direction consisted of building a modular meta-optimization framework [154, 131] that employs genetic algorithm (GA) [109] to fine-tune the free parameters (hyperparameters) of an arbitrary NFS model. However, the use of GA is not cognitively justified, and incorporating a more biologically plausible approach, such as the meta-learning method in [53, 231], would offer a prospective venue.

- Future development of INCA would also involve an incremental implementation and experimental validation of the *consolidation* and *inference cycles*, thus realizing more detailed interactions among different architectural modules. The general procedure would be to first start with a simple configuration comprising minimal number of modules and a validation test is accordingly performed using this configuration. In an automated vehicle system application, for instance, a minimal

configuration for enabling basic driving skills (e.g. lane following) would involve a transient memory and a procedural memory module, whose interaction can be implemented via a single DCN model. A successful configuration can then be extended to larger-scale organization by including new modules, such as a declarative memory for capturing general car navigation skills, an executive manager for automatic optimization of the car agent's configuration, and so on. Ultimately, such systematic development of module interactions would lead to a more comprehensive model of human cognition that can provide deeper understanding of the functional roles of and the information flow among various parts in the brain.

- One promising venue in which the INCA framework can be thoroughly tested is in the area of *virtual embodiment*, which involves controlling virtual agents living in virtual worlds [87]. The intelligent car simulation environment that has been developed in [203, 202] (and described partly in chapter 3, section 3.4) is an excellent example of such domain, requiring integration of different modules at various levels of abstraction, from operational to tactical/strategic level. The ultimate goal is to develop an automated driving system that is capable of reducing the burdens of the human driver while providing safe and smooth vehicle operations. Another prospective domain example is the development of *artificial creatures* in social network games [87], which requires integrative teaching/learning methodologies for the acquisition of linguistic and non-linguistic (perception-motor) knowledge. The consolidation and inference mechanisms provided by INCA would nicely serve this purpose, making it possible to systematically evaluate and guide progress toward building general machine intelligence in digital world.

- The idea of knowledge consolidation and metacognition in INCA also tallies closely with educational research, especially with regard to building *intelligent tutoring* systems [160, 88, 228]. Such application involves not only simple memorization of facts, but also higher cognitive elements such as reasoning and planning, knowledge construction and transfer, self-monitoring and -reflection, hypothesis testing, etc. This points in turn to a prospective application of INCA in the field of *natural*

language comprehension. Of particular practical interest in this area is a paradigm in which the INCA agent is exposed to massive (electronic) text resources and tasked to automatically learn and maintain large-scale semantic knowledge base, so as to facilitate comprehensive language understanding and commonsense reasoning [260, 58]. Successfully addressing this issue would allow to build intelligent avatars with human-like minds, that can utilize multiple information repositories and interact with the human operators in a natural way.

All in all, it is expected that, while advancing further in the development and experimental validations of the INCA framework, a new generation of cognitive architectures and models would emerge with the ability to acquire, consolidate and exploit knowledge effectively, and to monitor, regulate and improve themselves at even-higher cognitive levels. It would then be possible to create large-scale, multi-component intelligent systems that are substantially more effective than they are today. In turn, this would open a door for novel applications of the systems in various challenging tasks requiring robust learning and social cognition, thereby eventually realizing a true general machine intelligence.

Appendix A

Yager Inference Scheme

This appendix elaborates the Yager inference scheme employed by the RFCMAC system in chapter 4. Section A.1 begins by introducing the fundamentals of the Yager inference procedure. An example illustrating the procedure is subsequently given in section A.2.

A.1 Basic Concept

The inference procedure in fuzzy logic system aims at deducing valid conclusions for the observed inputs from a set of IF-THEN fuzzy rules. The most commonly used approach to infer the conclusion at the output is *Generalized Modus Ponens* (GMP) [151], as formally defined in (Eq. A.1) for all inputs $x \in \mathbf{X}$ and outputs $y \in \mathbf{Y}$

$$\begin{aligned} \text{Fuzzy rule: IF } x \text{ is } A \text{ THEN } y \text{ is } B \qquad & (\equiv A \rightarrow B) \\ \text{Observed input: } x \text{ is } \tilde{A} \qquad & (\equiv \tilde{A}) \\ \text{Conclusion: } y \text{ is } \tilde{B} \qquad & (\equiv \tilde{B}) \end{aligned} \qquad \text{(Eq. A.1)}$$

The fuzzy rule in (Eq. A.1) essentially corresponds to fuzzy relation R on the Cartesian product space $\mathbf{X} \times \mathbf{Y}$ derived from fuzzy labels A and B using fuzzy implication operation $I : [0,1]^2 \rightarrow [0,1]$, as per (Eq. A.2)

$$\mu_R(x,y) = I\left(\mu_A(x), \mu_B(y)\right) \qquad \text{(Eq. A.2)}$$

There are two approaches to interpret (Eq. A.2): one is based on the *conjunctive model* of fuzzy relation and another is based on the *disjunctive model* of fuzzy relation [54]. The *Compositional Rule of Inference* (CRI) scheme [294] is a well-established example that

adopts the former approach, in which the implication operator $I(.)$ translates into fuzzy conjunction operator, T-norm or $T(.)$, as per (Eq. A.3)

$$\mu_R(x, y) = T\left(\mu_A(x), \mu_B(y)\right) \tag{Eq. A.3}$$

Despite its wide use in the approximate reasoning literature [151], the CRI scheme poses many difficulties in the choice of appropriate implication (or conjunction) operator [277]. Furthermore, the scheme involves composition operations that are conceptually unclear, and it produces an undesirable output: $\tilde{B} = \tilde{A} \circ (A \rightarrow B) \neq B$ when $\tilde{A} = A$ and the implication $I(.)$ is based on conjunctive model of fuzzy relation [277].

The Yager inference scheme [124] employed by the RFCMAC system resolves the aforementioned problems by employing the disjunctive model of fuzzy relation, which is realized using a fuzzy disjunction operator, S-norm or $S(.)$, and a fuzzy negation operator, $N(.)$. The formula (Eq. A.4) summarizes the procedure

$$\mu_R(x, y) = S\left(N\left(\mu_A(x)\right), \mu_B(y)\right) \tag{Eq. A.4}$$

It can be seen that (Eq. A.4) maps closely to the statement '$\neg A \cup B$' and correspondingly the material implication $A \rightarrow B$ in crisp logic, which is conceptually simple and clear. The other key feature is that the output will track precisely the rule consequence when the input perfectly matches the rule antecedent, i.e., if $\tilde{A} = A$ then $\tilde{B} = \tilde{A} \circ (A \rightarrow B) = B$. These features provide in turn the primary motivation to map the Yager inference scheme into the RFCMAC framework, yielding the RFCMAC-Yager system.

A.2 Illustrative Example

An exemplary problem illustrating the Yager inference process in the RFCMAC-Yager system is presented in Figure A.1. The problem involves a car brake control system which comprises two inputs, velocity v_1 and distance to a front obstacle d_2, and a single output, brake force b_1. The two fuzzy rules used are given in (Eq. A.5) and (Eq. A.6)

$$R_1 : \text{IF } v_1 \text{ is } \mathit{Fast} \text{ and } d_2 \text{ is } \mathit{Near} \text{ THEN } b_1 \text{ is } \mathit{Strong} \tag{Eq. A.5}$$

$$R_2 : \text{IF } v_1 \text{ is } \mathit{Slow} \text{ and } d_2 \text{ is } \mathit{Far} \text{ THEN } b_1 \text{ is } \mathit{Weak} \tag{Eq. A.6}$$

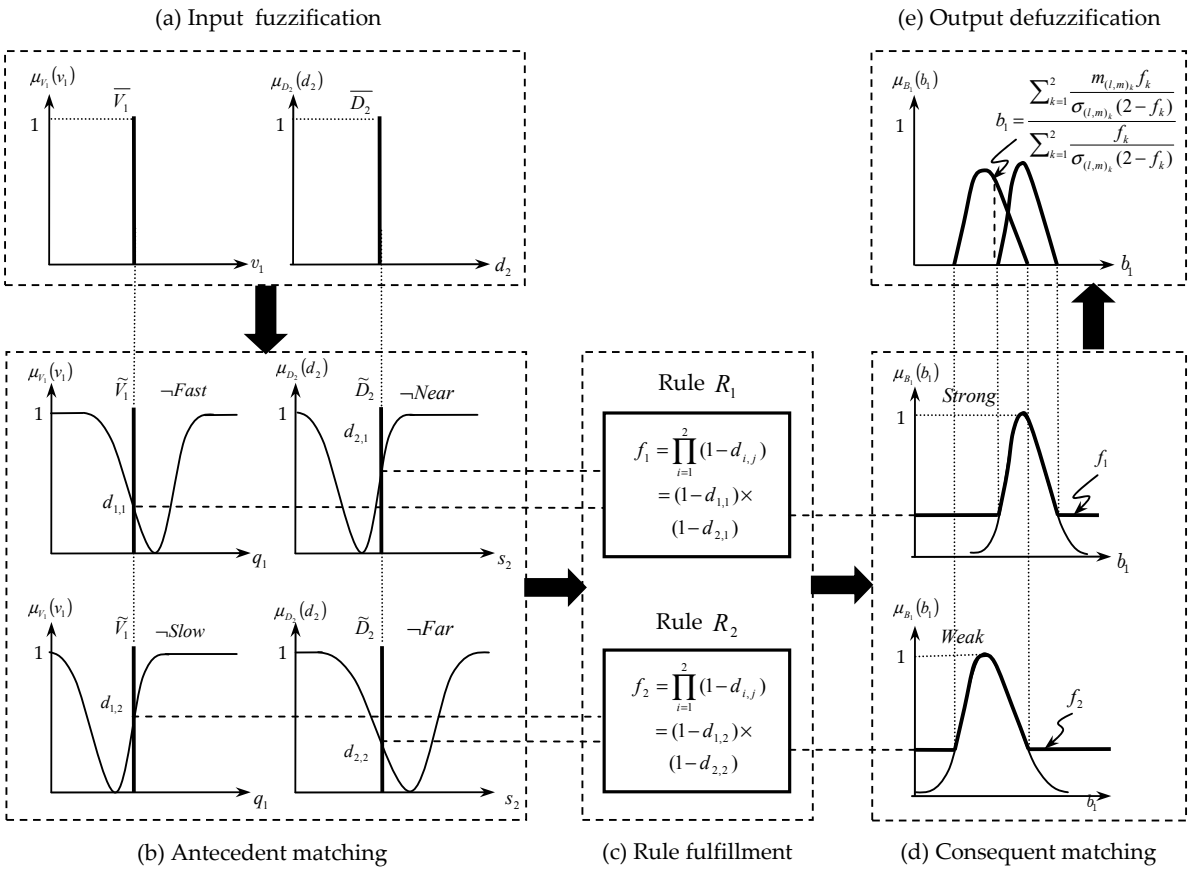


Figure A.1: Example of Yager inference in the RFCMAC-Yager system

In this example, it is assumed that the antecedent labels $(Slow, Fast)$ and $(Near, Far)$ (for inputs v_1 and d_2, respectively) are located adjacent to one another.

Firstly, the inputs v_1 and d_2 are fuzzified into fuzzy singletons $\tilde{V}_1$ and $\tilde{D}_2$, respectively, as shown in Figure A.1(a). Subsequently, the dissimilarity measures $d_{1,1}$ and $d_{1,2}$ between input v_1 and (neighboring) antecedent labels $Fast$ and $Slow$ are computed respectively, and similarly $d_{2,1}$ and $d_{2,2}$ for labels $Near$ and Far associated with d_2. Figure A.1(b) summarizes this process. Next, the physical rule nodes in RFCMAC-Yager that correspond to R_1 and R_2 are invoked to derive the firing strengths f_1 and f_2 based on $d_{1,1}, d_{2,1}$ and $d_{1,2}, d_{2,2}$, respectively, as per Figure A.1(c). In turn, a consequent matching process takes place to invoke the parameters of consequent labels $Strong$ and $Weak$, with which R_1 and R_2 are linked respectively, as illustrated in Figure A.1(d). Lastly, defuzzification procedure is carried out to derive the final, crisp output b_1, as in Figure A.1(e).

Appendix B

Parameter Tuning Derivation

This appendix describes the mathematical derivation of the RFCMAC parameter tuning phase, which is concerned with procedures for updating the kernel parameters of the fuzzy labels in the formulated rule base. Section B.1 and B.2 present the updating rule derivations for the consequent labels and antecedent labels, respectively.

B.1 Consequent Part

With reference to (Eq. 4.24), the updating of kernel parameter $\theta_{l,m}$ (i.e., centroid $m_{l,m}$ or width $\sigma_{l,m}$ for Gaussian MF) in a consequent label $C_{l,m}$ depends on the derivative term $\frac{\partial E}{\partial \theta_{l,m}}$. The term can be resolved using the standard *chain rule*, as per (Eq. B.1)

$$\begin{aligned}\frac{\partial E}{\partial \theta_{l,m}} &= \frac{\partial E}{\partial y_m}\frac{\partial y_m}{\partial \theta_{l,m}} \\ &= -\left(t_m - y_m\right)\frac{\partial y_m}{\partial \theta_{l,m}}\end{aligned} \qquad \text{(Eq. B.1)}$$

To derive $\frac{\partial y_m}{\partial \theta_{l,m}}$, the output defuzzification formula in (Eq. 4.2) first needs to be rewritten in terms of the individual parameter $\theta_{l,m}$ of consequent label $C_{l,m}$, as given in (Eq. B.2)

$$\begin{aligned}y_m &= \frac{\sum_{k\in\mathbf{S}} \frac{m_{(l,m)_k} f_k}{\sigma_{(l,m)_k}(2-f_k)}}{\sum_{k\in\mathbf{S}} \frac{f_k}{\sigma_{(l,m)_k}(2-f_k)}} \\ &= \frac{\sum_{l=1}^{L_m}\left(\frac{m_{l,m}}{\sigma_{l,m}}\sum_{k\in\mathbf{S}_{l,m}}\frac{f_k}{2-f_k}\right)}{\sum_{l=1}^{L_m}\left(\frac{1}{\sigma_{(l,m)_k}}\sum_{k\in\mathbf{S}_{l,m}}\frac{f_k}{2-f_k}\right)}\end{aligned} \qquad \text{(Eq. B.2)}$$

where $\mathbf{S}_{l,m}$ is the set of selected rule indices that point to $C_{l,m}$.

For simplicity, let N_m and D_m denote the numerator and denominator of (Eq. B.2)

respectively. Using this notation and (Eq. B.2), $\frac{\partial y_m}{\partial \theta_{l,m}}$ can be resolved as in (Eq. B.3)

$$\begin{aligned}\frac{\partial y_m}{\partial \theta_{l,m}} &= \frac{D_m \frac{\partial \sum_{l'=1}^{L_m}\left(\frac{m_{l',m}}{\sigma_{l',m}}\sum_{k\in\mathbf{S}_{l',m}}\frac{f_k}{2-f_k}\right)}{\partial \theta_{l,m}} - N_m \frac{\partial \sum_{l'=1}^{L_m}\left(\frac{1}{\sigma_{l',m}}\sum_{k\in\mathbf{S}_{l',m}}\frac{f_k}{2-f_k}\right)}{\partial \theta_{l,m}}}{D_m^2} \\ &= \frac{1}{D_m}\frac{\partial \sum_{l'=1}^{L_m}\left(\frac{m_{l',m}}{\sigma_{l',m}}\sum_{k\in\mathbf{S}_{l',m}}\frac{f_k}{2-f_k}\right)}{\partial \theta_{l,m}} - \frac{y_m}{D_m}\frac{\partial \sum_{l'=1}^{L_m}\left(\frac{1}{\sigma_{l',m}}\sum_{k\in\mathbf{S}_{l',m}}\frac{f_k}{2-f_k}\right)}{\partial \theta_{l,m}}\end{aligned} \tag{Eq. B.3}$$

In the context of $m_{l,m}$, (Eq. B.3) subsequently evaluates to (Eq. B.4)

$$\begin{aligned}\frac{\partial y_m}{\partial m_{l,m}} &= \frac{1}{D_m}\frac{1}{\sigma_{l,m}}\sum_{k\in\mathbf{S}_{l,m}}\frac{f_k}{2-f_k} - \frac{y_m}{D_m}(0) \\ &= \frac{1}{D_m\sigma_{l,m}}\sum_{k\in\mathbf{S}_{l,m}}\frac{f_k}{2-f_k}\end{aligned} \tag{Eq. B.4}$$

while the corresponding formula for $\sigma_{l,m}$ is presented in (Eq. B.5)

$$\begin{aligned}\frac{\partial y_m}{\partial \sigma_{l,m}} &= \frac{1}{D_m}\left(-\frac{m_{l,m}}{\sigma_{l,m}^2}\right)\sum_{k\in\mathbf{S}_{l,m}}\frac{f_k}{2-f_k} - \frac{y_m}{D_m}\left(-\frac{1}{\sigma_{l,m}^2}\right)\sum_{k\in\mathbf{S}_{l,m}}\frac{f_k}{2-f_k} \\ &= \left(\frac{y_m - m_{l,m}}{D_m\sigma_{l,m}^2}\right)\sum_{k\in\mathbf{S}_{l,m}}\frac{f_k}{2-f_k}\end{aligned} \tag{Eq. B.5}$$

Substituting back (Eq. B.4) and (Eq. B.5) into (Eq. B.1) accordingly yields (Eq. B.6) and (Eq. B.7) respectively

$$\frac{\partial E}{\partial m_{l,m}} = -\left(\frac{t_m - y_m}{D_m\sigma_{l,m}}\right)\sum_{k\in\mathbf{S}_{l,m}}\frac{f_k}{2-f_k} \tag{Eq. B.6}$$

$$\frac{\partial E}{\partial \sigma_{l,m}} = -\left(\frac{(t_m - y_m)(y_m - m_{l,m})}{D_m\sigma_{l,m}^2}\right)\sum_{k\in\mathbf{S}_{l,m}}\frac{f_k}{2-f_k} \tag{Eq. B.7}$$

As such, the (full) updating formulae of the parameters $m_{l,m}$ and $\sigma_{l,m}$ for each p^{th} data point can be written as in (Eq. B.8) and (Eq. B.9), respectively

$$\Delta m_{l,m}^{(p)} = \eta\left(\frac{t_m^{(p)} - y_m^{(p)}}{D_m^{(p)}\sigma_{l,m}^{(p)}}\right)\sum_{k\in\mathbf{S}_{l,m}^{(p)}}\frac{f_k^{(p)}}{2-f_k^{(p)}} \tag{Eq. B.8}$$

$$\Delta \sigma_{l,m}^{(p)} = \eta\left(\frac{\left(t_m^{(p)} - y_m^{(p)}\right)\left(y_m^{(p)} - m_{l,m}^{(p)}\right)}{D_m^{(p)}(\sigma_{l,m}^{(p)})^2}\right)\sum_{k\in\mathbf{S}_{l,m}^{(p)}}\frac{f_k^{(p)}}{2-f_k^{(p)}} \tag{Eq. B.9}$$

where $\Delta m_{l,m}^{(p)} = m_{l,m}^{(p+1)} - m_{l,m}^{(p)}$ and $\Delta \sigma_{l,m}^{(p)} = \sigma_{l,m}^{(p+1)} - \sigma_{l,m}^{(p)}$.

It should be noted, however, that each consequent label $C_{l,m}$ may be linked to many rules R_k. As a result, the amount of update computed in (Eq. B.8)-(Eq. B.9) may be overly large. To avoid excessive changes and upsetting the learning stability, the amount of update is scaled down by accounting for the number of rules being selected $|\mathbf{S}_{l,m}^{(p)}|$ which point to $C_{l,m}$. The resultant revised updating formulae are subsequently given in (Eq. B.10)-(Eq. B.11)

$$\Delta m_{l,m}^{(p)} = \frac{\eta}{|\mathbf{S}_{l,m}^{(p)}|}\left(\frac{t_m^{(p)} - y_m^{(p)}}{D_m^{(p)}\sigma_{l,m}^{(p)}}\right)\sum_{k\in\mathbf{S}_{l,m}^{(p)}}\frac{f_k^{(p)}}{2-f_k^{(p)}} \tag{Eq. B.10}$$

$$\Delta \sigma_{l,m}^{(p)} = \frac{\eta}{|\mathbf{S}_{l,m}^{(p)}|}\left(\frac{\left(t_m^{(p)} - y_m^{(p)}\right)\left(y_m^{(p)} - m_{l,m}^{(p)}\right)}{D_m^{(p)}(\sigma_{l,m}^{(p)})^2}\right)\sum_{k\in\mathbf{S}_{l,m}^{(p)}}\frac{f_k^{(p)}}{2-f_k^{(p)}} \tag{Eq. B.11}$$

B.2 Antecedent Part

Similar to the consequent section, the updating of each antecedent label $A_{i,j}$ is proportional to the derivative term $\frac{\partial E}{\partial \theta_{i,j}}$, which subsequently evaluates to (Eq. B.12)

$$\begin{aligned}\frac{\partial E}{\partial \theta_{i,j}} &= \frac{\partial \frac{1}{2}\sum_{m=1}^{M}(t_m - y_m)^2}{\partial \theta_{i,j}} \\ &= -\sum_{m=1}^{M}(t_m - y_m)\frac{\partial y_m}{\partial \theta_{i,j}}\end{aligned} \tag{Eq. B.12}$$

Using the notations N_m and D_m described before, the term $\frac{\partial y_m}{\partial \theta_{i,j}}$ translates to (Eq. B.13)

$$\begin{aligned}\frac{\partial y_m}{\partial \theta_{i,j}} &= \frac{D_m\frac{\partial\sum_{k\in\mathbf{S}}\frac{m_{(l,m)_k}f_k}{\sigma_{(l,m)_k}(2-f_k)}}{\partial\theta_{i,j}} - N_m\frac{\partial\sum_{k\in\mathbf{S}}\frac{f_k}{\sigma_{(l,m)_k}(2-f_k)}}{\partial\theta_{i,j}}}{D_m^2} \\ &= \frac{1}{D_m}\sum_{k\in\mathbf{S}}\frac{m_{(l,m)_k}}{\sigma_{(l,m)_k}}\frac{\partial\frac{f_k}{(2-f_k)}}{\partial\theta_{i,j}} - \frac{y_m}{D_m}\sum_{k\in\mathbf{S}}\frac{1}{\sigma_{(l,m)_k}}\frac{\partial\frac{f_k}{(2-f_k)}}{\partial\theta_{i,j}} \\ &= \frac{1}{D_m}\sum_{k\in\mathbf{S}}\left(\frac{m_{(l,m)_k} - y_m}{\sigma_{(l,m)_k}}\right)\frac{\partial\frac{f_k}{(2-f_k)}}{\partial\theta_{i,j}} \\ &= \frac{1}{D_m}\sum_{k\in\mathbf{S}}\left(\frac{m_{(l,m)_k} - y_m}{\sigma_{(l,m)_k}}\right)\left(\frac{(2-f_k)\frac{\partial f_k}{\partial\theta_{i,j}} - f_k\frac{\partial(2-f_k)}{\partial\theta_{i,j}}}{(2-f_k)^2}\right) \\ &= \frac{1}{D_m}\sum_{k\in\mathbf{S}}\left(\frac{m_{(l,m)_k} - y_m}{\sigma_{(l,m)_k}}\right)\left(\frac{2}{(2-f_k)^2}\right)\frac{\partial f_k}{\partial\theta_{i,j}}\end{aligned} \tag{Eq. B.13}$$

where $\theta_{i,j}$ denotes either the centroid $m_{i,j}$ or width $\sigma_{i,j}$ in Gaussian MF. In the case of $m_{i,j}$, the term $\frac{\partial f_k}{\partial \theta_{i,j}}$ can be evaluated as per (Eq. B.14)

$$\begin{aligned}
\frac{\partial f_k}{\partial m_{i,j}} &= \frac{\partial \prod_{i'=1}^{I} \left(1 - d_{(i',j)_k}\right)}{\partial m_{i,j}} \\
&= \frac{\partial \prod_{i'=1}^{I} \exp\left(-\frac{(m_{i',j}-x_i)^2}{\sigma_{i',j}^2}\right)}{\partial m_{i,j}} \\
&= \left[\prod_{i'=1, i'\neq i}^{I} \exp\left(-\frac{(m_{i',j}-x_{i'})^2}{\sigma_{i',j}^2}\right)\right]\left[\frac{\partial \exp\left(-\frac{(m_{i,j}-x_i)^2}{\sigma_{i,j}^2}\right)}{\partial m_{i,j}}\right] \\
&= \left[\prod_{i'=1, i'\neq i}^{I} \exp\left(-\frac{(m_{i',j}-x_{i'})^2}{\sigma_{i',j}^2}\right)\right]\left[\frac{-2(m_{i,j}-x_i)}{\sigma_{i,j}^2}\exp\left(-\frac{(m_{i,j}-x_i)^2}{\sigma_{i,j}^2}\right)\right] \\
&= \frac{-2(m_{i,j}-x_i)}{\sigma_{i,j}^2}\prod_{i'=1}^{I}\exp\left(-\frac{(m_{i',j}-x_{i'})^2}{\sigma_{i',j}^2}\right) \\
&= \frac{2(x_i - m_{i,j})}{\sigma_{i,j}^2} f_k
\end{aligned} \tag{Eq. B.14}$$

while, for $\sigma_{i,j}$, the term translates to (Eq. B.15)

$$\begin{aligned}
\frac{\partial f_k}{\partial \sigma_{i,j}} &= \left[\prod_{i'=1, i'\neq i}^{I} \exp\left(-\frac{(m_{i',j}-x_{i'})^2}{\sigma_{i',j}^2}\right)\right]\left(\frac{\partial \exp\left(-\frac{(m_{i,j}-x_i)^2}{\sigma_{i,j}^2}\right)}{\partial \sigma_{i,j}}\right] \\
&= \left[\prod_{i'=1, i'\neq i}^{I} \exp\left(-\frac{(m_{i',j}-x_{i'})^2}{\sigma_{i',j}^2}\right)\right]\left[\frac{2(m_{i,j}-x_i)^2}{\sigma_{i,j}^3}\exp\left(-\frac{(m_{i,j}-x_i)^2}{\sigma_{i,j}^2}\right)\right] \\
&= \frac{2(m_{i,j}-x_i)^2}{\sigma_{i,j}^3}\prod_{i'=1}^{I}\exp\left(-\frac{(m_{i',j}-x_{i'})^2}{\sigma_{i',j}^2}\right) \\
&= \frac{2(x_i - m_{i,j})^2}{\sigma_{i,j}^3} f_k
\end{aligned} \tag{Eq. B.15}$$

Substituting (Eq. B.14)-(Eq. B.15) into (Eq. B.13), and then into (Eq. B.12), yield (Eq. B.16)-(Eq. B.17) respectively

$$\frac{\partial E}{\partial m_{i,j}} = -\left(\frac{x_i - m_{i,j}}{\sigma_{i,j}^2}\right)\sum_{m=1}^{M}\left(\frac{t_m - y_m}{D_m}\right)\sum_{k\in\mathbf{S}}\left(\frac{m_{(l,m)_k} - y_m}{\sigma_{(l,m)_k}}\right)\left(\frac{4f_k}{(2-f_k)^2}\right) \tag{Eq. B.16}$$

$$\frac{\partial E}{\partial \sigma_{i,j}} = -\left(\frac{(x_i - m_{i,j})^2}{\sigma_{i,j}^3}\right)\sum_{m=1}^{M}\left(\frac{t_m - y_m}{D_m}\right)\sum_{k\in\mathbf{S}}\left(\frac{m_{(l,m)_k} - y_m}{\sigma_{(l,m)_k}}\right)\left(\frac{4f_k}{(2-f_k)^2}\right) \tag{Eq. B.17}$$

which in turn lead to the updating formulae in (Eq. B.18)-(Eq. B.19)

$$\Delta m_{i,j}^{(p)} = \eta \left(\frac{x_i^{(p)} - m_{i,j}^{(p)}}{(\sigma_{i,j}^{(p)})^2} \right) \sum_{m=1}^{M} \left(\frac{t_m^{(p)} - y_m^{(p)}}{D_m^{(p)}} \right) \sum_{k \in \mathbf{S}} \left(\frac{m_{(l,m)_k}^{(p)} - y_m^{(p)}}{\sigma_{(l,m)_k}^{(p)}} \right) \left(\frac{4 f_k^{(p)}}{(2 - f_k^{(p)})^2} \right) \tag{Eq. B.18}$$

$$\Delta \sigma_{i,j}^{(p)} = \eta \left(\frac{(x_i^{(p)} - m_{i,j}^{(p)})^2}{(\sigma_{i,j}^{(p)})^3} \right) \sum_{m=1}^{M} \left(\frac{t_m^{(p)} - y_m^{(p)}}{D_m^{(p)}} \right) \sum_{k \in \mathbf{S}} \left(\frac{m_{(l,m)_k}^{(p)} - y_m^{(p)}}{\sigma_{(l,m)_k}^{(p)}} \right) \left(\frac{4 f_k^{(p)}}{(2 - f_k^{(p)})^2} \right) \tag{Eq. B.19}$$

where $\Delta m_{i,j}^{(p)} = m_{i,j}^{(p+1)} - m_{i,j}^{(p)}$ and $\Delta \sigma_{i,j}^{(p+1)} = \sigma_{i,j}^{(p+1)} - \Delta \sigma_{i,j}^{(p)}$.

As with (Eq. B.10)-(Eq. B.11), downscaling is required to prevent large updates in the antecedent label parameters. To achieve this, both (Eq. B.18) and (Eq. B.19) are multiplied by a factor of $\frac{1}{4M|\mathbf{S}|}$, where $|\mathbf{S}|$ and M are the number of rules being selected and number of output dimensions respectively. The modified updating equations are given in (Eq. B.20)-(Eq. B.21)

$$\Delta m_{i,j}^{(p)} = \frac{\eta}{M|\mathbf{S}|} \left(\frac{x_i^{(p)} - m_{i,j}^{(p)}}{(\sigma_{i,j}^{(p)})^2} \right) \sum_{m=1}^{M} \left(\frac{t_m^{(p)} - y_m^{(p)}}{D_m^{(p)}} \right) \sum_{k \in \mathbf{S}} \left(\frac{m_{(l,m)_k}^{(p)} - y_m^{(p)}}{\sigma_{(l,m)_k}^{(p)}} \right) \left(\frac{f_k^{(p)}}{(2 - f_k^{(p)})^2} \right) \tag{Eq. B.20}$$

$$\Delta \sigma_{i,j}^{(p)} = \frac{\eta}{M|\mathbf{S}|} \left(\frac{(x_i^{(p)} - m_{i,j}^{(p)})^2}{(\sigma_{i,j}^{(p)})^3} \right) \sum_{m=1}^{M} \left(\frac{t_m^{(p)} - y_m^{(p)}}{D_m^{(p)}} \right) \sum_{k \in \mathbf{S}} \left(\frac{m_{(l,m)_k}^{(p)} - y_m^{(p)}}{\sigma_{(l,m)_k}^{(p)}} \right) \left(\frac{f_k^{(p)}}{(2 - f_k^{(p)})^2} \right) \tag{Eq. B.21}$$

Appendix C

Time Performance

This appendix provides estimates of learning (training) time required by the proposed RFCMAC and DCN systems for various tasks presented in section 4.5 and 5.5, respectively. All experiments were conducted using a personal computer with Core 2 Duo 2.13 GHz processor, 4 MB L2 cache, and 2 GB random access memory. The time performances of the RFCMAC and DCN systems are provided in section C.1 and C.2, respectively.

C.1 RFCMAC Time Performance

Table C.1: Learning time of RFCMAC-Yager for Nakanishi datasets

Data	Learning time (sec)
Nonlinear system	0.890
Chemical plant	2.563
Stock market	0.031

Table C.2: Learning time of RFCMAC-Yager for water plant dataset

Data	Learning time (sec)			
	CV1	CV2	CV3	Average
2-class set	0.204	0.219	0.766	0.396 ± 0.320
3-class set	0.860	0.531	0.515	0.635 ± 0.195

Table C.3: Learning time of RFCMAC-Yager for leukemia dataset

Data	Learning time (sec)
Independent set	0.969
Leave-one-out set	2.189 ± 0.147

C.2 DCN Time Performance

Table C.4: Consolidation time of DCN for TIS dataset

Setting	Episode length (N_e)	Learning time (sec)			
		CV1	CV2	CV3	Average
Single-episode	8917 ($= 2/3 \times 13375$)	12.891	12.391	12.875	12.719 ± 0.284
Multi-episode	500	16.672	15.812	15.391	15.958 ± 0.653
	1000	22.078	21.875	21.969	21.974 ± 0.102
	1500	18.313	27.328	28.828	24.823 ± 5.687
	2000	18.953	31.016	31.359	27.109 ± 7.066
	2500	17.610	30.578	34.344	27.511 ± 8.779
	3000	17.985	30.156	32.235	26.792 ± 7.698
	3500	18.187	20.297	22.062	20.182 ± 1.950
	4000	17.421	21.656	21.891	20.323 ± 2.516

Table C.5: Learning time of DCN for MIT face dataset

Data	Episode length (N_e)	Learning time (sec)
Original set	6977	7.234
Extended set	10000	4409.440

Appendix D

Publication List

D.1 Journal Paper

- R. J. Oentaryo, M. Pasquier and C. Quek, "RFCMAC: A novel reduced localized neuro-fuzzy system approach to knowledge extraction," *Expert Systems with Applications*, to be published. [**2009 ISI impact factor: 2.908**].

- R. J. Oentaryo and M. Pasquier, "Knowledge consolidation and inference in the integrated neuro-cognitive architecture," *IEEE Intelligent Systems*, to be published.[**2009 ISI impact factor: 3.144**].

- R. J. Oentaryo, M. Pasquier, and C. Quek, "GenSoFNN-Yager: A novel brain-inspired neuro-fuzzy decision support system realizing Yager inference," *Expert Systems with Applications*, vol. 35, no. 4, pp. 1825-1840, 2008. [**2009 ISI impact factor: 2.908**].

- M. Pasquier and R. J. Oentaryo, "Learning to drive the human way: A step towards intelligent vehicles," *International Journal of Vehicle Autonomous Systems, Special Issue on Advances in Autonomous Vehicle Technologies for Urban Environment*, vol. 6, no. 1-2, pp. 24-47, 2008.

- R. J. Oentaryo, M. Pasquier, and C. Quek, "Dual consolidation network," *Computational Intelligence*, submitted.

- R. J. Oentaryo, E. Lumanpauw, and M. Pasquier, "Generic hyper-parameter evolutionary optimization of pattern recognition system," *IEEE Transactions on Sys-*

tems, Man, and Cybernetics, Part B, submitted.

- L. San, M. J. Er, R. J. Oentaryo, B. S. Lim, X. Li, and L. -Y. Zhai, "Growing and pruning-based parsimonious fuzzy neural network: An effective approach to data-driven modeling," *IEEE Transactions on Neural Networks*, submitted.

- M. J. Er and R. J. Oentaryo, "Book review - Computational intelligence: Methods and techniques," *IEEE Computational Intelligence Magazine*, submitted.

- R. J. Oentaryo, D. Partouche, M. Pasquier, and C. Quek, "Anticipatory driving system based on the generic self-organizing fuzzy neural network," in preparation.

D.2 Conference Paper

- R. J. Oentaryo and M. Pasquier, "A novel dual neuro-fuzzy system approach for large-scale knowledge consolidation," in *Proceedings of the IEEE International Conference on Fuzzy Systems*, Jeju, South Korea, 2009, pp. 53-58. [**IEEE CIS Student Travel Grant**]

- D. Ko, R. J. Oentaryo, and M. Pasquier, "An adaptive history network method to improve the genetic optimization of pattern recognition systems," in *Proceedings of the IEEE International Conference on Fuzzy Systems*, Jeju, South Korea, 2009, pp. 23-28. [**IEEE CIS Student Travel Grant**]

- R. J. Oentaryo and M. Pasquier, "Towards a novel integrated neuro-cognitive architecture (INCA)," in *Proceedings of the IEEE International Joint Conference on Neural Networks*, Hong Kong, China, 2008, pp. 1902-1909.

- R. J. Oentaryo and M. Pasquier, "A reduced rule-based localist network for data comprehension," in *Proceedings of the IEEE International Joint Conference on Neural Networks*, Hong Kong, China, 2008, pp. 2660-2667.

- W. Duch, R. J. Oentaryo, and M. Pasquier, "Cognitive architectures: Where do we go from here?", in *Frontiers in Artificial Intelligence and Applications*, P. Wang, B. Goertzel, and S. Franklin, Eds.: IOS Press, 2008, vol. 171, pp. 122-136.

- E. Lumanpauw, M. Pasquier, and R. J. Oentaryo, "Generic GA-based meta-level parameter optimization for pattern recognition systems," in *Proceedings of the IEEE Congress on Evolutionary Computation*, Singapore, 2007, pp. 1593-1600.

- R. J. Oentaryo and M. Pasquier, "GenSoFNN-Yager: A novel hippocampus-like learning memory system realizing Yager inference," in *Proceedings of the IEEE International Joint Conference on Neural Networks*. Vancouver, BC, Canada, 2006, pp. 705-712. [**IEEE CIS Student Travel Grant, and Best Session Presentation Award**]

- R. J. Oentaryo and M. Pasquier, "Self-trained automated parking system," in *Proceedings of the IEEE International Conference on Control, Automation, Robotics and Vision*, Kunming, China, 2004, vol. 2, pp. 1005-1010.

- R. J. Oentaryo, M. J. Er, L. San, L. -Y Zhai, and X. Li, "Bayesian ART-based fuzzy inference system: A new approach to prognosis of machining processes," *IEEE International Conference on Prognostics and Health Management*, submitted.

- M. Pratama, M. J. Er, X. Li, R. J. Oentaryo, L. San, L. -Y. Zhai, A. T. Jahromi, and I. Arifin, "Tool wear prediction using evolutionary dynamic fuzzy neural network," *IEEE International Conference on Prognostics and Health Management*, submitted.

D.3 Book Chapter

- R. J. Oentaryo, M. Pasquier, and W. Duch, "From cognitive architectures to general human-like intelligence," in *Developing a Roadmap to Human Level Intelligence*, W. Duch and J. R. Taylor, Eds.: Springer, submitted.

D.4 Technical Report

- R. J. Oentaryo, "Automated driving based on self organizing GenSoYager neuro-fuzzy system," Centre for Computational Intelligence, Nanyang Technological University, Singapore, Technical Report C2i-TR-002/05, 2005.

Bibliography

[1] A. Agah and G. A. Bekey. Cognitive architecture for robust adaptive control of robots in a team. *Journal of Intelligent and Robotic Systems*, 20:251–273, 1997.

[2] A. Agah and G. A. Bekey. Phylogenetic and ontogenetic learning in a colony of interacting robots. *Autonomous Robots*, 4:85–100, 1997.

[3] J. S. Albus. A theory of cerebellar function. *Mathematical Biosciences*, 10:2561, 1971.

[4] J. S. Albus. A new approach to manipulator control: The cerebellar model articulation controller. *Journal of Dynamic Systems, Measurement, and Control*, 97(220-227), 1975.

[5] J. S. Albus and A. J. Barbera. RCS: A cognitive architecture for intelligent multiagent systems. *Annual Reviews in Control*, 29(1):87–99, 2005.

[6] I. Aleksander. Neural systems engineering: Towards a unified design discipline? *Computing & Control Engineering Journal*, 1(6):259–265, 1990.

[7] P. Alvarez and L. R. Squire. Memory consolidation and the medial temporal lobe. *Proceedings of the National Academic Scicences of USA*, 91:7041–7045, 1994.

[8] J. A. Anderson, P. Allopenna, G. S. Guralnik, D. Scheinberg, J. A. Santini, S. Dimitriadis, B. B. Machta, and B. T. Merritt. Programming a parallel computer: The Ersatz brain project. In W. Duch, J. Mandziuk, and J. M. Zurada, editors, *Challenges to Computational Intelligence*. Springer, Berlin, 2006.

[9] J. R. Anderson. *The Architecture of Cognition*. Harvard University Press, Cambridge, MA, 1983.

[10] J. R. Anderson. *Learning and Memory.* Wiley, New York, 2nd edition, 2000.

[11] J. R. Anderson. Human symbol manipulation within an integrated cognitive architecture. *Cognitive Science*, 29(3):313–341, 2005.

[12] J. R. Anderson, S. A. Betts, J. L. Ferris, and J. M. Fincham. Neural imaging to track mental states while using an intelligent tutoring system. *Proceedings of the National Academy of Science*, 107(15):7018–7023, 2010.

[13] J. R. Anderson, D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere, and Y. Qin. An integrated theory of the mind. *Psychological Review*, 111(4):1036–1060, 2004.

[14] J. R. Anderson and C. Lebiere. The Newell test for a theory of cognition. *Behavioral and Brain Science*, 26:587–637, 2003.

[15] K. K. Ang and C. Quek. RSPOP: Rough set-based pseudo outer-product fuzzy rule identification algorithm. *Neural Computation*, 17:205–243, 2005.

[16] P. P. Angelov and X. Zhou. Evolving fuzzy-rule-based classifiers from data streams. *IEEE Transactions on Fuzzy Systems*, 16(6):1462–1475, 2008.

[17] B. Ans, S. Carbonnel, and S. Valdois. A connectionist model multiple-trace memory model for polysyllabic word reading. *Psychological Review*, 105:678–723, 1998.

[18] B. Ans and S. Rousset. Avoiding catastrophic forgetting by coupling two reverberating neural networks. *Academie des Sciences, Sciences de la vie*, 320:989–997, 1997.

[19] J. L. Armony, D. Servan-Schreiber, J. D. Cohen, and J. E. LeDoux. Computational modeling of emotion: Explorations through the anatomy and physiology of fear conditioning. *Trends in Cognitive Sciences*, 1:28–34, 1997.

[20] A. Asuncion and D.J. Newman. UCI Machine Learning Repository. *School of Information and Computer Science, University of California. Available: http://www.ics.uci.edu/ mlearn/MLRepository.html*, 2007.

[21] B. J. Baars and S. Franklin. How conscious experience and working memory interact. *Trends in Cognitive Sciences*, 7(4):166–172, 2003.

[22] A. Baddeley. *Working Memory*. Oxford University Press, New York, 1986.

[23] A. Baddeley. The episodic buffer: A new component of working memory? *Trends in Cognitive Sciences*, 4(11):417–423, 2000.

[24] A. G. Barto, R. S. Sutton, and P. Brouwer. Associative search network: A reinforcement learning associative memory. *Biological Cybernetics*, 40:201–211, 1981.

[25] A. G. Barto, R. S. Sutton, and C. J. C. H. Watkins. Learning and sequential decision making. In M. Gabriel and J. W. Moore, editors, *Learning and Computational Neuroscience: Foundations of Adaptive Networks*, pages 539–602. MIT Pres, 1990.

[26] J. Baxter. Learning internal representations. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8. MIT Press, Cambridge, MA, 1996.

[27] J. M. Bernardo and A. F. M. Smith. *Bayesian Theory*. Wiley, New York, 1996.

[28] R. Bogacz and C. Giraud-Carrier. A novel modular neural architecture for rule-based and similarity-based reasoning. In S. Wermter and R. Sun, editors, *Hybrid Neural Systems*. Springer-Verlag, Heidelberg, 2000.

[29] R. P. Bonasso, R. J. Firby, E. Gat, D. Kortenkamp, D. Miller, and M. Slack. Experiences with an architecture for intelligent, reactive agents. *Journal of Experimental and Theoretical Artificial Intelligence*, 9:237–256, 1997.

[30] E. S. Boyden, A. Katoh, and J. L. Raymond. Cerebellum-dependent learning: the role of multiple plasticity mechanisms. *Annual Review of Neuroscience*, 27:581–609, 2004.

[31] M. E. Bratman. *Intentions, Plans, and Practical Reason*. Harvard University Press, Cambridge, MA, 1987.

[32] R. Brooks. Elephants don't play chess. *Robotics and Autonomous Systems*, 6:3–15, 1986.

[33] R. Brooks and L. A. Stein. Building brains for bodies. *Autonomous Robotics*, 1:7–25, 1994.

[34] J. P. Brunet, P. Tamayo, T. R. Golub, and J. P. Mesirov. Metagenes and molecular pattern discovery using matrix factorization. *Proceedings of the National Academic Sciences of USA*, 101:4164–4169, 2004.

[35] J. G. Carbonell, O. Etzioni, Y. Gil., R. Joseph, C. A. Knoblock, S. Minton, and M. M. Veloso. Prodigy: An integrated architecture for planning and learning. *SIGART Bulletin*, 2(4):51–55, 1991.

[36] R. Carpenter and J. . Freeman. Computing machinery and the individual: The personal Turing test. *Available at: http://www.jabberwacky.com/*, 2005.

[37] N. L. Cassimatis. *Polyscheme: A cognitive architecture for integrating multiple representation and inference schemes*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 2002.

[38] N. L. Cassimatis. A cognitive substrate for human-level intelligence. *AI Magazine*, 27(2):45–56, 2006.

[39] N. L. Cassimatis, J. Trafton, M. Bugajska, and A. Schultz. Integrating cognition, perception and action through mental simulations in robotocs. *Robotics and Autonomous Systems*, 49(1-2):13–23, 2004.

[40] G. C. Cawley and N. L. C. Talbot. Gene selection in cancer classification using sparse logistic regression with Bayesian regularisation. *Bioinformatics*, 22(19):2348–2355, 2006.

[41] D. Chakraborty and N. R. Pal. A neuro-fuzzy scheme for simultaneous feature selection and fuzzy rule-based classification. *IEEE Transactions on Neural Networks*, 15(1):110–123, 2004.

[42] D. Choi, T. Konik, N. Nejati, C. Park, and P. Langley. A believable agent for first-person shooter games. In *Proceedings of the Annual Artificial Intelligence and Interactive Digital Entertainment Conference*, pages 71–73, Stanford, CA, 2007.

[43] M. L. Chow, E. J. Moler, and I. S. Mian. Identifying marker genes in transcription profiling data using a mixture of feature relevance experts. *Physiological Genomics*, 5:99–111, 2001.

[44] T. M. Cover and P. E. Hart. Nearest pattern classification. *IEEE Transactions on Information Theory*, IT-13(1):21–27, 1967.

[45] A. Crawford and M. Beckerle. Purification and characterization of zyxin, an 82,000-dalton component of adherens junctions. *Journal of Biological Chemistry*, 266(9):5847–5853, 1991.

[46] T. R. Davies and S. J. Russel. A logical approach to reasoning by analogy. In *Proceedings of the International Joint Conference of Artificial Intelligence*, volume 1, pages 264–270. Morgan Kauffmann, Milan, 1987.

[47] M. Davis, D. L. Walker, and Y. Lee. Neurophysiology and neuropharmacology of startle and its affective modulation. In M. E. Dawson, A. M. Schell, and A. H. Bohmelt, editors, *Startle Modification: Implications for Neuroscience, Cognitive Science, and Clinical Science*, pages 95–113. Cambridge University Press, 1999.

[48] H. de Garis, M. Korkin, F. Gers, E. Nawa, and M. Hough. Building an artificial brain using an FPGA based CAM-brain machine. *Applied Mathematics and Computation*, 111(2-3):163–192, 2000.

[49] T. Dean. A computational model of the cerebral cortex. In *Proceedings of the National Conference on Artificial Intelligence*, pages 938–943, Cambridge, MA, 2005. MIT Press.

[50] G. DeJong and R. Mooney. Explanation-based learning: An alternative view. *Machine Learning*, 1:145–176, 1986.

[51] M. d'Inverno, M. Luck, M. Georgeff, D. Kinny, and M. Wooldridge. The dMARS Architechure: A specification of the distributed multi-agent reasoning system. *Autonomous Agents and Multi-Agent Systems*, 9(1-2):5–53, 2004.

[52] P. Domingos and M. Pazzani. On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29:103130, 1997.

[53] K. Doya. Metalearning and neuromodulation. *Neural Networks*, 15:495–506, 2002.

[54] D. Dubois and H. Prade. What are fuzzy rules and how to use them? *Fuzzy Sets and Systems*, 84(2):169–185, 1996.

[55] W. Duch. Filter methods. In I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, editors, *Feature Extraction, Foundations and Applications*, Series Studies in Fuzziness and Soft Computing, pages 89–118. Physica-Verlag, Springer, 2006.

[56] W. Duch. Towards comprehensive foundations of computational intelligence. In W. Duch and J. Mandziuk, editors, *Challenges for Computational Intelligence*, Springer Studies in Computational Intelligence, pages 261–316. Springer, 2007.

[57] W. Duch and K. Grudzinski. Meta-learning via search combined with parameter optimization. In *Inteligent Information Systems*, Advances in Soft Computing, pages 13–22. Springer, 2002.

[58] W. Duch, P. Matykiewicz, and J. Pestian. Neurolinguistic approach to natural language processing with applications to medical text analysis. *Neural Networks*, 21(10):1500–1510, 2008.

[59] W. Duch, R. J. Oentaryo, and M. Pasquier. Cognitive architectures: Where do we go from here? In P. Wang, B. Goertzel, and S. Franklin, editors, *Frontiers in Artificial Intelligence and Applications*, volume 171, pages 122–136. IOS Press, 2008.

[60] W. Duch, R. Setiono, and J.M. Zurada. Computational intelligence methods for rule-based data understanding. *Proceedings of the IEEE*, 92(5):771–805, 2004.

[61] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, New York, NY, 2nd edition, 2000.

[62] H. Eichenbaum. *The Cognitive Neuroscience of Memory: An Introduction*. Oxford University Press, New York, 2002.

[63] M. Estomih and G. Gregory. *Clinical Neuroanatomy and Neuroscience*. Saunders, Philadelphia, 2006.

[64] M. W. Eysenck and M. Keane. *Cognitive Pscyhology: A Student's Handbook*. Psychology Press, 5th edition, 2005.

[65] R. E. Fan, K. W. Chang, C. J. Hsieh, X. R. Wang, and C. J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[66] T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27:861–874, 2006.

[67] U. Fayyad and K. B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1022–1029. Morgan Kauffman, Chambry, France, 1993.

[68] E. A. Feigenbaum. Some challenges and grand challenges for computational intelligence. *Journal of the ACM*, 50(1):32–40, 2003.

[69] M. H Ferkin, A. A. Pierce, and S. Franklin. Self-referencing and recognition in meadow voles, microtus pennsylvanicus. *Animal Cognition*, 114:863–874, 2008.

[70] B. Ferry, B. Roozendaal, and J. McGaugh. Role of norepinephrine in mediating stress hormone regulation of long-term memory storage: A critical involvement of the amygdala. *Biological Psychiatry*, 46(9):1140–1152, 1999.

[71] R. J. Firby. *Adaptive execution in complex dynamic worlds*. PhD thesis, Ph.D. Thesis, Yale University, 1989.

[72] S. Franklin. IDA: A conscious artifact? *Journal of Consciousness Studies*, 10:47–66, 2003.

[73] S. Franklin. The LIDA architecture: Adding new modes of learning to an intelligent, autonomous, software agent. In *Proceedings of the International Conference on Integrated Design and Process Technology*. Society for Design & Process Science, San Diego, 2006.

[74] S. Franklin, A. Kelemen, and L. McCauley. IDA: A cognitive agent architecture. In *Proceedings of the IEEE Conference on Systems, Man and Cybernetics*, volume 3, pages 2646–2651. San Diego, CA, 1998.

[75] M. French. Pseudo-recurrent connectionist networks: An approach to the "sensitivity-stability"dilemma. *Connection Science*, 9(4):353–379, 1997.

[76] Y. Freund and R. E. Schapire. Large margin classification using the perceptron algorithm. In *Annual Conference on Computational Learning Theory*, pages 209–217. New York, NY, 1998.

[77] D. Friedlander and Franklin. LIDA and a theory of mind. In *Proceedings of the First Artificial General Intelligence Conference*, Memphis, TN, 2008. IOS Press.

[78] W. T. Fu, D. Bothell, S. Douglass, C. Haimson, M. H. Sohn, and J. A. Anderson. Toward a real-time model-based training system. *Interacting with Computers*, 18(6):1216–1230, 2006.

[79] J. M. Fuster. Prefrontal cortex and the bridging of temporal gaps in the perception-action cycle. In A. Diamond, editor, *The Development and Neural Bases of Higher Cognitive Functions*, volume 608, pages 318–329. New York Academy of Sciences, New York, 1990.

[80] J. M. Fuster. *Cortex and Mind: Unifying Cognition*. Oxford University Press, 2003.

[81] H. Gardner. *Multiple Intelligences: The Theory in Practice*. Basic Books, New York, 1993.

[82] D. Geman, C. d'Avignon, D. Naiman, and R. L. Winslow. Classifying Gene Expression Profiles from Pairwise mRNA Comparisons. *Statistical Applications in Genetics and Molecular Biology*, 3(1):19, 2004.

[83] M. Georgeff and F. F. Ingrand. Real-time reasoning: The monitoring and control of spacecraft systems. In *Proceedings of the IEEE Conference on Artificial Intelligence Applications*, volume 1, pages 198–204, Santa Barbara, CA, 1990.

[84] M. P. Georgeff and A. Lansky. Reactive reasoning and planning: An experiment with a mobile robot. In *Proceedings of the National Conference on Artificial Intelligence*, pages 677–682, 1987.

[85] K. A. Gluck and R. W. Pew. *Modeling Human Behavior with Integrated Cognitive Architectures: Comparison, Evaluation, and Validation.* Lawrence Erlbaum Associates, 2005.

[86] M. A. Gluck and C. Myers. Hippocampal mediation of stimulus representation: a computational theory. *Hippocampus*, 3:491516, 1993.

[87] B. Goertzel, C. Pennachin, N. Geissweiller, M. Looks, A. Senna, W. Silva, A. Heljakka, and C. Lopes. An integrative methodology for teaching embodied non-linguistic agents, applied to virtual animals in Second Life. In P. Wang, B. Goertzel, and S. Franklin, editors, *Frontiers in Artificial Intelligence and Applications*, volume 171, pages 161–175. IOS Press, 2008.

[88] C. M. Goh and C. Quek. EpiList: An intelligent tutoring system shell for implicit development of generic cognitive skills that support bottom-up knowledge construction. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 37(1):58–71, 2007.

[89] D. E. Goldberg. *Genetic Algorithms in Seach, Optimization, and Machine Learning.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 1989.

[90] R. N. Goldman and J. S. Weinberg. *Statistics: An Introduction.* Prentice Hall, Englewood Cliffs, NJ, 1985.

[91] P. S. Goldman-Rakic and H. R. Friedman. Topography of cognition: Parallel distributed networks in primary association cortex. *Annual Review of Neuroscience*, 11:137–156, 1988.

[92] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.

[93] A. M. Graybiel. Building action repertories: Memory and learning functions of the basal ganglia. *Current Opinion in Neurobiology*, 5:733–741, 1995.

[94] S. Grossberg. Adaptive pattern classification and universal recoding: I. Parallel development and coding of neural feature detectors. *Biological Cybernetics*, 23:121–134, 1976.

[95] S. Grossberg. Competitive learning: From interactive activation to adaptive resonance. *Cognitive Science*, 11:23–63, 1987.

[96] S. Grossberg. The imbalanced brain: From normal behavior to schizophrenia. *Biological Psychiatry*, 48:81–98, 2000.

[97] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[98] K. Haigh and M. M. Veloso. Interleaving planning and robot execution for asynchronous user requests. In *Proceedings of the International Conference on Intelligent Robots and Systems*, Osaka, Japan, 1996.

[99] M. A. Hall. *Correlation-based feature subset selection for machine learning.* PhD thesis, Department of Computer Science, University of Waikato, 1999.

[100] C. J. Harris, X. Hong, and J. Q. Gan. *Adaptive Modelling, Estimation and Fusion from Data: A Neurofuzzy Approach.* Springer, 2002.

[101] M. E. Hasselmo and B. P. Wyble. Free recall and recognition in a network model of the hippocampus: Simulating effects of scopolamon human memory function. *Behavioral Brain Research*, 89:1–34, 1997.

[102] J. Hawkins and S. Blakeslee. *On Intelligence: How a New Understanding of the Brain Will Lead to the Creation of Truly Intelligent Machines*. Times Books, 2004.

[103] D. O. Hebb. *The Organization of Behavior: A Neuropsychological Theory*. John Wiley & Sons, New York, 1949.

[104] R. Hecht-Nielsen. A cortronic neural network models of cortical function. In S. Usui and T. Omori, editors, *Proceedings of the International Conference on Neural Information Processing*. IOA Press, KitaKyushu, Japan, 1998.

[105] R. Hecht-Nielsen. *Confabulation Theory: The Mechanism of Thought*. Springer, 2007.

[106] B. Heisele, T. Poggio, and M. Pontil. Face detection in still gray images. Technical Report A.I. Memo 1687, Center for Biological and Computational Learning, Massachusetts Institute of Technology, Cambridge, MA, 2000.

[107] G. E. Hinton. Connectionist learning procedures. *Artificial Intelligence*, 40:185–234, 1989.

[108] J. A. Hobson and E. F. Pace-Schott. The cognitive neuroscience of sleep: Neuronal systems, consciousness and learning. *Nature Reviews Neuroscience*, 3:679–693, 2002.

[109] J. H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, 1975.

[110] K. S. Hwang and Y. B. Hsu. A self-improving fuzzy cerebellar model articulation controller with stochastic action generation. *Cybernetics and Systems*, 33(2):101–128, 2002.

[111] F. F. Ingrand, M. P. Georgeff, and A. S. Rao. An architecture for real-time reasoning and system control. *IEEE Expert*, 7(6):34–44, 1992.

[112] J. S. R. Jang. ANFIS: Adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man and Cybernetics*, 23(3):665–685, 1993.

[113] R. M. Jones, J. E. Laird, P. E. Nielsen, K. J. Coulter, P. Kenny, and F. V. Koss. Automated intelligent pilots for combar flight simulation. *AI Magazine*, 20(1):27–41, 1999.

[114] J. Jonides, E. E. Smith, R. A. Koeppe, E. Awh, S. Minoshima, and M. A. Mintun. Spatial working memory in humans as revealed by PET. *Nature*, 363:623–625, 1993.

[115] C. C. Jou. A fuzzy cerebellar model articulation controller. In *Proceedings of the IEEE International Conference on Fuzzy Systems*, pages 1171–1178. San Diego, 1992.

[116] M. A. Just, P. A. Carpenter, and S. Varma. Computational modeling of high-level cognition and brain function. *Human Brain Mapping*, 8:128–136, 1999.

[117] S. Káli and P. Dayan. Off-line replay maintains declarative memories in a model of hippocampal neocortical interactions. *Nature Neuroscience*, 7(3):286–294, 2004.

[118] G. Kalton. *Introduction to Survey Sampling*. SAGE University Paper series on Quantitative Applications in the Social Sciences. SAGE Publications, Inc., Beverly Hills and London, 1983.

[119] G. A. Kaminka and C. R. Burghart. Evaluating architectures for intelligence. Technical report, Technical Report WS-07-04, AAAI Press, Menlo Park, CA, 2007.

[120] E. Kandel. Small systems of neurons. *Scientific American*, 3:61–70, 1979.

[121] E. R. Kandel, J. H. Schwartz, and T. M. Jessel. *Principles of Neural Science*. McGraw-Hill, Health Professions Division, New York, 4th edition, 2000.

[122] P. Kanerva. *Sparse Distributed Memory*. MIT Press, Cambridge, MA, 1988.

[123] N. Kasabov. Evolving fuzzy neural networks for supervised/unsupervised online. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 31(6):902–918, 2001.

[124] J. M. Keller, R. R. Yager, and H. Tahani. Neural network implementation of fuzzy logic. *Fuzzy Sets and Systems*, 45(1):1–12, 1992.

[125] W. G. Kennedy, M. D. Bugajska, A. M. Harrison, and J. G. Trafton. Like-Me simulation as an effective and cognitively plausible basis for social robotics. *International Journal of Social Robotics*, 1(2):181–194, 2009.

[126] S. J. Kia and G. G. Coghill. Unsupervised clustering and centroid estimation using dynamic competitive learning. *Biological Cybernetics*, 67:433–443, 1991.

[127] D. Kieras and D. E. Meyer. An overview of the EPIC architecture for cognition and performance with application to human-computer interaction. *Human-Computer Interaction*, 12:391–438, 1997.

[128] S. Killcross, T. Robbins, and B. Everitt. Different types of fear-conditioned behaviour mediated by separate nuclei within amygdala. *Nature*, 388(6640):377380, 1997.

[129] K. Kiryazov, G. Petkov, M. Grinberg, B. Kokinov, and C. Balkenius. The interplay of analogy-making with active vision and motor control in anticipatory robots. In *Lecture Notes in Artificial Intelligence*, volume 4520, pages 233–253. Springer Verlag, Berlin, 2007.

[130] A. H. Klopf. A neuronal model of classical conditioning. *Psychobiology*, 16(2):85–123, 1988.

[131] D. Ko, R. J. Oentaryo, and M. Pasquier. An adaptive history network method to improve the genetic optimization of pattern recognition systems. In *Proceedings of the IEEE International Conference on Fuzzy Systems*, pages 23–28, Jeju, South Korea, 2009.

[132] T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69, 1982.

[133] B. Kokinov. The DUAL cognitive architecture: A hybrid multi-agent approach. In A. Chon, editor, *Proceedings of the European Conference on Artificial Intelligence*, pages 203–207. John Wiley & Sons, London, 1994.

[134] B. Kokinov and A. Petrov. Integration of memory and reasoning in analogy-making: The AMBR model. In D. Gentner, K. Holyoak, and B. Kokinov, editors, *The Analogical Mind: Perspectives from Cognitive Science*. MIT Press, Cambridge, MA, 2001.

[135] E. Korner and G. Matsumoto. Cortical architecture and self-referential control for brain-like computation. *IEEE Engineering in Medicine and Biology Magazine*, 21(5):121–133, 2002.

[136] J. L. Krichmar and G. M. Edelman. Brain-based devices for the study of nervous systems and the development of intelligent machines. *Artificial Life*, 11:63–77, 2005.

[137] J. K. Krushke. ALCOVE: An exemplar-based connectionist model of category learning. *Psychological Review*, 99:22–44, 1992.

[138] J. E. Laird. Extending the Soar cognitive architecture. In P. Wang, B. Goertzel, and S. Franklin, editors, *Frontiers in Artificial Intelligence and Applications*, volume 171, pages 224–235. IOS Press, 2008.

[139] J. E. Laird, P. S. Rosenbloom, and A. Newell. Soar: An architecture for general intelligence. *Artificial Intelligence*, 33:1–64, 1987.

[140] P. Langley. An adaptive architecture for physical agents. In *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pages 18–25. Compiegne, France, 2005.

[141] P. Langley. Cognitive architectures and general intelligent systems. *AI Magazine*, 27:33–44, 2006.

[142] P. Langley and D. Choi. Learning recursive control programs from problem solving. *Journal of Machine Learning Research*, 7:493–518, 2006.

[143] N. Larvac and S. Dzeroski. *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood, New York, 1994.

[144] A. Lazo and P. Rathie. On the entropy of continuous probability distributions. *IEEE Transactions on Information Theory*, 24(1):120–122, 1978.

[145] J. LeDoux. Brain mechanisms of emotion and emotional learning. *Current Opinion in Neurobiology*, 2:191–197, 1992.

[146] C. C. Lee. Fuzzy logic in control systems: Fuzzy logic controller Part II. *IEEE Transactions on Systems, Man and Cybernetics*, 20(2):419435, 1990.

[147] R. L. Lewis. An architecturally-based theory of sentence comprehension. In *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society*, pages 108–113, 1993.

[148] J. Li and L. Wong. Identifying good diagnostic gene groups from gene expression profiles using the concept of emerging patterns. *Bioinformatics*, 18(5):725–734, 2002.

[149] C. J. Lin and C. T. Lin. An ART-based fuzzy adaptive learning control network. *IEEE Transactions on Fuzzy Systems*, 5(4):477–496, 1997.

[150] C. T. Lin and C. S. G. Lee. Reinforcement structure/parameter learning for neural-network based fuzzy logic control systems. *IEEE Transactions on Fuzzy Systems*, 2:46–63, 1994.

[151] C. T. Lin and C. S. G. Lee. *Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems*. Prentice Hall, Upper Saddle River, NJ, 1996.

[152] F. Liu, C. Quek, and G. S. Ng. A novel generic Hebbian-ordering-based fuzzy rule base reduction approach to Mamdani neuro-fuzzy system. *Neural Computation*, 19:1656–1680, 2007.

[153] M. Looks, B. Goertzel, and C. Pennachin. Novamente: An integrative architecture for general intelligence. In DC Washington, editor, *Proceedings of the AAAI Symposium on Achieving Human-Level Intelligence via Integrated Systems and Research*, 2003.

[154] E. Lumanpauw, M. Pasquier, and R. J. Oentaryo. Generic GA-based meta-level parameter optimization for pattern recognition systems. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 1593–1600, Singapore, 2007.

[155] R. Maclin and J. Shavlik. Incorporating advice into agents that learn from reinforcements. In *Proceedings of the National Conference on Artificial Intelligence*, pages 694–699, San Meteo, CA, 1994. Morgan Kauffmann.

[156] E. H. Mamdani and S. Assilian. An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Human-Computer Studies*, 51(2):135–147, 1999.

[157] C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[158] H. Markram. The blue brain project. *Nature Reviews Neuroscience*, 7(2):153–160, 2006.

[159] D. Marr. A theory of cerebellar cortex. *Journal of Physiology*, 202:437–479, 1969.

[160] T. Matsui. Optimization method for selecting problems using the learner's model in intelligent adaptive instruction system. *IEICE Transactions on Information and Systems*, E80-D(2):196–205, 1997.

[161] B. W. Matthews. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta*, 405:442451, 1975.

[162] J. L. McClelland, B. L. McNaughton, and R. C. O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychological Review*, 102(3):419–457, 1995.

[163] J. L. McClelland, D. E. Rumelhart, and the PDP Research Group. *Parallel Distributed Processing: Explorations in the Microstructures of Cognition*. MIT Press, Cambridge, MA, 1986.

[164] M. McCloskey and N. J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *The Pscyhology of Learning and Motivation*, volume 24, pages 109–165. Academic Press, New York, 1989.

[165] J. L. McGaugh. Memory - A century of consolidation. *Science*, 287:248–251, 2000.

[166] T. McKenna. The development of cortical models to enable neural-based cognitive architectures. In R. Hecht-Nielsen and T. McKenna, editors, *Computational Models for Neuroscience: Human Cortical Information Processing*, pages 171–204. Springer-Verlag, London, 2003.

[167] M. Meeter, C. E. Myers, and M. A. Gluck. Integrating incremental learning and episodic memory models of the hippocampal region. *Psychological Review*, 112(3):560–58, 2005.

[168] D. E. Meyer and D. E. Kieras. A computational theory of executive cognitive processes and multiple-task performance: Part 1. Basic mechanisms. *Psychological Review*, 104(1):3–65, 1997.

[169] D. Michie. *On Machine Intelligence*. Ellis Horwood, Chichester, 2nd edition, 1986.

[170] W. T. Miller, F. H. Glanz, and L. G. Kraft. Cmac: An associative neural network alternative to backpropagation. *Proceedings of the IEEE*, 78(10):1561–1567, 1990.

[171] B. Milner. Visual recognition and recall after right temporal lobe excision in man. *Neuropsychologia*, 6:191, 1968.

[172] M. Minsky. A framework for representing knowledge. In P. Winston, editor, *The Psychology of Computer Vision*. McGraw-Hill, New York, 1975.

[173] S. N. Minton. Quantitative results concerning the utility of explanation-based learning. *Artificial Intelligence*, 42:363–391, 1990.

[174] T. M. Mitchell, J. Allen, P. Chalasani, J. Cheng, O. Etzioni, M. Ringuette, and J. C. Schlimmer. Theo: A framework for self-improving systems. In K. VanLehn, editor, *Architectures for Intelligence*, pages 323–355. Lawrence Erlbaum Associates, Hillsdale, NJ, 1991.

[175] T. M. Mitchell, R. Keller, and S. Kedar-Cabelli. Explanation-based generalization: A unifying view. *Machine Learning*, 1:47–80, 1986.

[176] S. H. Muggleton. *Inductive Logic Programming*. New York. Academic Press, 1992.

[177] N. Muscettola, P. P. Nayak, B. Pell, and B. C. Williams. Remote Agent: To boldly go where no AI system has gone before. *Artificial Intelligence*, 103(1-2):5–47, 1998.

[178] D. J. Musliner, E. H. Durfee, and K. G. Shin. CIRCA: A cooperative intelligent real-time control architecture. *IEEE Transactions on Systems, Man and Cybernetics*, 23(6):1561–1574, 1993.

[179] M. Mutafchieva and B. Kokinov. Can language be replaced? Physical representations of relations instead of language labels in relational mapping: Do they help young children? In *Proceedings of the 29th Annual Conference of the Cognitive Science Society*, Hillsdale, NJ, 2007.

[180] L. Nadel, A. Samsonovich, L. Ryan, and M. Moscovitch. Multiple trace theory of human memory: Computational, neuroimaging and neuropsychological results. *Hippocampus*, 10:352–368, 2000.

[181] K. Nader, G. E. Schafe, and J. E. LeDoux. Fear memories require protein synthesis in the amygdala for reconsolidation after retrieval. *Nature*, 406:722726, 2000.

[182] H. Nakanishi, I. B. Turksen, and M. Sugeno. A review and comparison of six reasoning methods. *Fuzzy Sets and Systems*, 57(3):257–294, 1993.

[183] D. Nauck, F. Klawonn, and R. Kurse. *Foundations of Neuro-Fuzzy Systems*. John Wiley & Sons, New York, NY, 1997.

[184] I. Naveh and R. Sun. A cognitively based simulation of academic science. *Computational and Mathematical Organization Theory*, 12:313–337, 2006.

[185] A. Nestor and B. Kokinov. Towards active vision in the DUAL cognitive architecture. *International Journal of Information Theories and Applications*, 11(1):9–15, 2004.

[186] A. Newell. *Unified Theories of Cognition*. Harvard University Press, 1990.

[187] A. Newell and H. A. Simon. GPS: A program that simulates human thought. In E. A. Feigenbaum and J. Feldman, editors, *Computers and Thought*. McGraw-Hill, New York, 1963.

[188] G. S. Ng, C. Quek, and H. Jiang. FCMAC-EWS: A bank failure early warning system based on a novel localized pattern learning and semantically associative fuzzy neural network. *Expert Systems with Applications*, 34(2):989–1003, 2008.

[189] D. V. Nguyen and D. M. Rocke. Tumor classification by partial least squares using microarray gene expression data. *Bioinformatics*, 18(1):39–50, 2002.

[190] M. N. Nguyen, D. Shi, and C. Quek. FCMAC-BYY: Fuzzy CMAC using Bayesian Ying-Yang learning. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 36(5):1180–1190, 2006.

[191] J. Nie and D.A. Linkens. FCMAC: A fuzzified cerebellar model articulation controller with self-organizing capacity. *Automatica*, 30(4):655–664, 1994.

[192] N. Nilsson. Human-level artificial intelligence? Be serious! *AI Magazine*, 26(4):68–75, 2005.

[193] A. Nuxoll and J. Laird. A cognitive model of episodic memory integrated with a general cognitive architecture. In *Proceedings of the International Conference on Cognitive Modeling*, pages 220–225, 2004.

[194] R. J. Oentaryo. Automated driving based on self-organizing GenSoYager neuro-fuzzy system. Technical Report C2i-TR-002/05, Centre for Computational Intelligence, Nanyang Technological University, 2005.

[195] R. J. Oentaryo and M. Pasquier. A reduced rule-based localist network for data comprehension. In *Proceedings of the International Joint Conference on Neural Networks*, pages 2660–2667. Hong Kong, 2008.

[196] R. J. Oentaryo and M. Pasquier. Towards a novel integrated neuro-cognitive architecture (INCA). In *Proceedings of the IEEE International Joint Conference on Neural Networks*, pages 1902–1909. Hong Kong, 2008.

[197] R. J. Oentaryo, M. Pasquier, and C. Quek. GenSoFNN-Yager: A novel brain-inspired generic self-organizing neuro-fuzzy system realizing Yager inference. *Expert Systems with Applications*, 35(4):1825–1840, 2008.

[198] R. C. O'Reilly, T. S. Braver, and J. D. Cohen. A biologically-based computational model of working memory. In A. Miyake and P. Shah, editors, *Models of Working Memory*, pages 375–411. Cambridge University Press, 1999.

[199] R. C. O'Reilly and M. J. Frank. Making working memory work: A computational model of learning in the prefrontal cortex and basal ganglia. *Neural Computation*, 18:283–328, 2006.

[200] R. C. O'Reilly and Y. Munakata. *Computational Explorations in Cognitive Neuroscience: Understanding of the Mind by Simulating the Brain*. MIT Press, Cambridge, MA, 2000.

[201] E. Parzen. On estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33:1065–1076, 1962.

[202] M. Pasquier and R. J. Oentaryo. Learning to drive the human way: A step towards intelligent vehicles. *International Journal of Vehicle Autonomous Systems, Special Issue on Advances in Autonomous Vehicle Technologies for Urban Environment*, 6(1-2):24–47, 2008.

[203] M. Pasquier, C. Quek, and M. Toh. Fuzzylot: A novel self-organising fuzzy-neural rule-based pilot system for automated vehicles. *Neural Networks*, 14(8):1099–1112, 2001.

[204] I. P. Pavlov. *Conditioned Reflexes: Investigation of the Physiological Activity of the Cerebral Cortex*. Oxford University Press, London, 1927.

[205] Z. Pawlak. Rough sets. *International Journal of Computer and Information Sciences*, 11:341–356, 1982.

[206] A. G. Pedersen and H. Nielsen. Neural network prediction of translation initiation sites in eukaryotes: Perspectives for EST and genome analysis. In *Proceedings of the International Conference on Intelligent Systems for Molecular Biology*, pages 226–233. 1997.

[207] F. J. Pelletier, G. Sutcliffe, and C. B. Suttner. The Development of CASC. *AI Communications*, 15(2-3):79–90, 2002.

[208] J. Peng, Y. Wang, and W. Sun. Trajectory-tracking control for mobile robot using recurrent fuzzy cerebellar model articulation controller. *Neural Information Processing - Letters and Reviews*, 11(1):15–23, 2007.

[209] G. Petkov and B. Kokinov. JUDGEMAP - Integration of analogy-making, judgment, and choice. In *Proceedings of the Annual Conference of the Cognitive Science Society*, Hillsdale, NJ, 2006. Erlbaum.

[210] G. Petkov, T. Naydenov, M. Grinberg, and B. Kokinov. Building robots with analogy-based anticipation. In C. Freksa, M. Kohlhase, and K. Schill, editors, *Lecture Notes in Artificial Intelligence*, volume 4314, pages 76–90. Springer Verlag, Berlin, 2007.

[211] J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schoelkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1998.

[212] J. L. Pollock. Rational cognition in OSCAR. In *Agent Theories, Architectures, and Languages*, pages 71–90. Springer, 1999.

[213] M. J. D. Powell. Radial basis functions for multivariable interpolation: A review. In J.C. Mason and M.G. Cox, editors, *Algorithims for Approximation*, pages 143–167. Oxford University Press, 1987.

[214] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C*. Cambridge University Press, Cambridge, England, 1986.

[215] C. Quek, M. Pasquier, and P. W. Ng. Structuring expert control using the integrated process supervision architecture. In *Expert Systems Techniques and Applications*. CRC Press, Newark, 1999.

[216] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.

[217] R. Quinlan. Inductive learning of decision trees. *Machine Learning*, 1:81–106, 1986.

[218] B. Rapp. *The Handbook of Cognitive Neuropsychology: What Deficits Reveal about the Human Mind*. Psychology Press, 2001.

[219] R. Ratcliff. Connectionist models of recognition memory: Constraints imposed by learning and forgetting functions. *Psychological Review*, 97:285–308, 1990.

[220] J. Rickel and W. Johnson. Task-oriented collaboration with embodied agents in virtual worlds. In J. Cassell, J. Sullivan, and S. Prevost, editors, *Embodied conversational agents*, pages 95–122. MIT Press, Boston, 2000.

[221] S. Ritter, J. R. Anderson, K. R. Koedinger, and A. Corbett. Cognitive tutor: Applied research in mathematics education. *Psychonomic Bulletin & Review*, 14:249–255, 2007.

[222] A. Robins. Catastrophic forgetting, rehearsal, and pseudorehearsal. *Connection Science*, 7:123–146, 1995.

[223] H. J. Rong, N. Sundararajan, G. B. Huang, and P. Saratchandran. Sequential adaptive fuzzy inference system (SAFIS) for nonlinear system identification and prediction. *Fuzzy Sets and Systems*, 157(9):1260–1275, 2006.

[224] J. Rosbe, R. S. Chong, and D. E. Kieras. Modeling with perceptual and memory constraints: An EPIC-Soar model of a simplified enroute air traffic control task. Technical report, SOAR Technology Inc., Ann Arbor, Michigan, 2001.

[225] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[226] R. Salgia, E. Pisick, M. Sattler, J. Li, N. Uemura, W. Wong, S. Burky, H. Hirai, L. Chen, and J. Griffin. p130CAS forms a signaling complex with the adapter protein crk1 in hematopoietic cells transformed by the BCR/ABL oncogene. *Journal of Biological Chemistry*, 271(41):25198–25203, 1996.

[227] A. V. Samsonovich and K. A. DeJong. Designing a self-aware neuromorphic hybrid. In K. R. Thorisson, H. Vilhjalmsson, and S. Marsela, editors, *Proceedings of the AAAI Workshop on Modular Construction of Human-Like Intelligence*, pages 71–78. Pittsburg, PA, 2005.

[228] A. V. Samsonovich, K. A. DeJong, A. Kitsantas, E. E. Peters, N. Dabbagh, and M. L. Kalbfleisch. Cognitive constructor: An intelligent tutoring system based on a biologically inspired cognitive architecture (BICA). In P. Wang, B. Goertzel, and S. Franklin, editors, *Frontiers in Artificial Intelligence and Applications*, volume 171, pages 311–325. IOS Press, 2008.

[229] D. L. Schacter. Implicit memory: History and current status. *Journal of Experimental Psychology: Learning, Memory and Cognition*, 13:501–518, 1987.

[230] N. Schmajuk and J. DiCarlo. Stimulus configuration, classical conditioning and hippocampal function. *Psychological Review*, 99:268–305, 1992.

[231] N. Schweighofer and K. Doya. Meta-learning in reinforcement learning. *Neural Networks*, 16:5–9, 2003.

[232] W. B. Scoville. Amnesia after bilateral mesial temporal-lobe excision: Introduction to case H.M. *Neuropsychologia*, 6:211–213, 1968.

[233] M. P. Shanahan. A cognitive architecture that combines internal simulation with a global workspace. *Consciousness and Cognition*, 15:433–449, 2006.

[234] M. P. Shanahan and B. J. Baars. Applying global workspace theory to the frame problem. *Cognition*, 98(2):157–176, 2005.

[235] S. C. Shapiro. The SNePS semantic network processing system. In N. V. Findler, editor, *Associative Networks: The Representation and Use of Knowledge by Computers*, pages 179–203. Academic Press, New York, 1979.

[236] L. Shastri. A computational model of episodic memory formation in the hippocampal system. *Neurocomputing*, 38-40:889–897, 2001.

[237] L. Shastri and V. Ajjanagadde. From simple associations to systematic reasoning: A connectionist encoding of rules, variables, and dynamic bindings using temporal synchrony. *Behavioral and Brain Sciences*, 16(3):417–494, 1993.

[238] J. Shavlik and T. Dietterich. *Readings in Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1990.

[239] Q. Shen and R. Jensen. Selecting informative features with fuzzy-rough sets and its application for complex systems monitoring. *Pattern Recognition*, 37(7):1351–1363, 2004.

[240] S. K. Shevade and S. S. Keerthi. A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics*, 19(17):2246–2253, 2003.

[241] J. Sim, W. L. Tung, and C. Quek. FCMAC-Yager: A novel Yager-inference-scheme-based fuzzy CMAC. *IEEE Transactions on Neural Networks*, 17(6):1394–1410, 2006.

[242] B. F. Skinner. *The Behavior of Organisms: An Experimental Analysis*. Appleton-Century-Crofts, New York, 1938.

[243] A. Sloman. What sort of architecture is required for a humanlike agent. In M. Wooldridge and A. Rao-Dordrecht, editors, *Foundations of Rational Agency*. Kluwer Academic Publishers, 1999.

[244] A. Smola and B. Schölkopf. A tutorial on support vector regression. Technical report, NeuroCOLT2 Technical Report Series, NC2-TR-1998-030, 1998.

[245] P. Smolensky. On the proper treatment of connectionism. *Behavioral and Brain Sciences*, 11:1–74, 1988.

[246] J. F. Sowa. Conceptual graphs for a database interface. *IBM Journal of Research and Development*, 20(4):336–357, 1976.

[247] L. R. Squire and N. J. Cohen. Human memory and amnesia. In G. Lynch, J. L. McGaugh, and N. M. Weinberger, editors, *Neurobiology of Learning and Memory*. Guilford Press, New York, 1984.

[248] L. R. Squire, C. E. L. Stark, and R. E. Clark. The medial temporal lobe. *Annual Review of Neuroscience*, 27:279–306, 2004.

[249] S. F. Su, Z. J. Lee, and Y. P. Wang. Robust and fast learning for fuzzy cerebellar model articulation controllers. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 36(1):203–208, 2006.

[250] R. Sun and F. Alexandre. *Connectionist Symbolic Integration*. Erlbaum, Hillsdale, NJ, 1997.

[251] R. Sun and L. Bookman. *Computational Architectures Integrating Neural and Symbolic Processes*. Kluwer Academic Publishers, Needham, MA, 1994.

[252] R. Sun, E. Merrill, and T. Peterson. From implicit skills to explicit knowledge: A bottom-up model of skill learning. *Cognitive Science*, 25(2):203–244, 2001.

[253] R. Sun and T. Peterson. Autonomous learning of sequential tasks: Experiments and analyses. *IEEE Transactions on Neural Networks*, 9(6):1217–1234, 1998.

[254] R. Sun and X. Zhang. Top-down versus bottom-up learning in cognitive skill acquisition. *Cognitive Systems Research*, 5(1):63–89, 2004.

[255] R. Sun and X. Zhang. Accounting for a variety of reasoning data within a cognitive architecture. *Journal of Experimental and Theoretical Artificial Intelligence*, 18(2):169–191, 2006.

[256] K. K. Sung. *Learning and example selection for object and pattern recognition*. PhD thesis, Artificial Intelligence Laboratory and Center for Biological and Computational Learning, Massachusetts Institute of Technology, Cambridge, MA, 1996.

[257] G. Sutcliffe and C. Suttner. The State of CASC. *AI Communications*, 19(1):35–48, 2006.

[258] R. S. Sutton and A. G. Barto. Time-derivative models of pavlovian reinforcement. In M. Gabriel and J. Moor, editors, *Learning and Computational Neuroscience: Foundations of Adaptive Networks*, volume 4, pages 497–537. MIT Press, 1990.

[259] W. Swartout, J. Gratch, R. E. H. Hill, S. Marsella, J. Rickel, and D. Traum. Toward virtual humans. *AI Magazine*, 27(2):96–108, 2006.

[260] J. Szymanski, T. Sarnatowicz, and Duch W. Towards avatars with artificial minds: Role of semantic memory. *Journal of Ubiquitous Computing and Intelligence*, 2:1–11, 2008.

[261] N. A. Taatgen, D. Huss, D. Dickison, and J. R. Anderson. The acquisition of robust and flexible cognitive skills. *Journal of Experimental Psychology: General*, 137(3):548–565, 2008.

[262] N. A. Taatgen and F. J. Lee. Production compilation: A simple mechanism to model complex skill acquisition. *Human Factors*, 45(1):61–76, 2003.

[263] T. Takagi and M. Sugeno. Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man and Cybernetics*, 15(1):116–132, 1985.

[264] M. Tambe, W. L. Johnson, R. M. Jones, F. Koss, J. E. Laird, P. S. Rosenbloom, and K. B. Schwamb. Intelligent agents for interactive simulation environments. *AI Magazine*, 16:15–39, 1995.

[265] A. C. Tan, D. Q. Naiman, L. Xu, R. L. Winslow, and D. Geman. Simple decision rules for classifying human cancers from gene expression profiles. *Bioinformatics*, 21(20):3896–3904, 2005.

[266] T. Z. Tan, C. Quek, and G. S. Ng. Ovarian cancer diagnosis by hippocampus and neocortex-inspired learning memory structures. *Neural Networks*, 18:818–825, 2005.

[267] R. F. Thompson, N. H. Donegan, and D. G. Lavond. The psychobiology of learning and memory. In R. C. Atkinson, R. J. Hermstein, G. Lindzey, and R. D. Luce, editors, *Steven's Handbook of Experimental Psychology*. Wiley, New York, 2nd edition, 1986.

[268] E. L. Thorndike. *Animal Intelligence: Experimental Studies*. Macmillan, New York, 1911.

[269] R. Tibshirani, T. Hastle, B. Narasimhan, and G. Chu. Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proceedings of the National Academic Sciences of USA*, 99(10):6567–6572, 2002.

[270] C. W. Ting and C. Quek. A novel blood glucose regulation using TSK0-FCMAC: A fuzzy CMAC based on the zero-ordered TSK fuzzy inference scheme. *IEEE Transactions on Neural Networks*, 20(5):563–582, 2009.

[271] M. E. Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001.

[272] "TOSCA". TOSCA: A comprehensive brain-based cognitive architecture. Technical report, DARPA-IPTO Biologically-Inspired Cognitive Architecture (BICA) Phase 1 Architecture Report, 2006.

[273] I. W. H. Tsang, J. T. Y. Kwok, and J. M. Zurada. Generalized core vector machines. *IEEE Transactions on Neural Network*, 15(5):1126–1140, 2006.

[274] E. Tulving. Episodic and semantic memory. In E. Tulving and W. Donaldson, editors, *Organization of Memory*. Academic Press, New York, 1972.

[275] W. L. Tung and C. Quek. GenSoFNN: A generic self-organizing fuzzy neural network. *IEEE Transactions on Neural Networks*, 13(5):1075–1086, 2002.

[276] A. Turing. Computing Machinery and Intelligence. *Mind*, 49:433–460, 1950.

[277] I. B. Turksen and Z. Zhong. An approximate analogical reasoning scheme based on similarity measures and interval valued fuzzy sets. *Fuzzy Sets and Systems*, 34:323–346, 1990.

[278] M. M. Veloso and J. G. Carbonell. Integrating analogy into a general problem-solving architecture. In M. Zemankova and Z. Ras, editors, *Intelligent Systems*, pages 29–51. Ellis Horwood, Chichester, UK, 1990.

[279] M. M. Veloso, J. G. Carbonell, A. Perez, D. Borrajo, and J. Blythe. Integrating Planning and Learning: The PRODIGY Architecture. *Journal of Experimental and Theoretical Artificial Intelligence*, 7:81–120, 1995.

[280] D. Vernon, G. Metta, and G. Sandini. A survey of artificial cognitive systems: Implications for the autonomous development of mental capabilities in computational agents. *IEEE Transactions on Evolutionary Computation*, 11(2):151–180, 2007.

[281] P. Wang. *Rigid Flexibility: The Logic of Intelligence*. Springer, 2006.

[282] Y. Wang, I. V. Tetko, M. A. Hall, E. Frank, A. Facius, K. F. X. Mayer, and H. W. Mewes. Gene selection from microarray data for cancer classification:A machine learning approach. *Computational Biology and Chemistry*, 29:37–46, 2005.

[283] Z. Q. Wang, J. L. Schiano, and M. Ginsberg. Hash-coding in CMAC neural networks. In *Proceedings of the IEEE International Conference on Neural Networks*, volume 3, pages 1698–1703, 1996.

[284] C. J. Watkins. *Learning with delayed rewards*. PhD thesis, Cambridge University, Psychology Department, Cambridge, UK, 1989.

[285] J. Weng. Developmental robotics: Theory and experiments. *International Journal of Humanoid Robotics*, 1(2):199–236, 2004.

[286] S. Wermter. *Hybrid connectionist natural language processing*. Thompson International, London, 1995.

[287] B. Widrow and S. D. Stearns. *Adaptive Signal Processing*. Prentice Hall, New Jersey, 1985.

[288] M. Wilson and B. McNaughton. Reactivation of hippocampal ensemble memories during sleep. *Science*, 265:676–679, 1994.

[289] G. Winocur. Anterograde and retrograde amnesia in rats with dorsal hippocampal or dorsomedial thalamic lesions. *Behavioural Brain Research*, 38:145–154, 1990.

[290] D. H. Wolpert and W. G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1:67–82, 1997.

[291] R. E. Wray, J. E. Laird, A. Nuxoll, D. Stokes, and A. Kerfoot. Synthetic adversaries for urban combat training. In *Proceedings of the Innovative Applications of Artificial Intelligence Conference*, San Jose, CA, 2004.

[292] T. Yagi, A. Morimoto, M. Eguchi, S. Hibi, M. Sako, E. Ishii, S. Mizutani, S. Imashuku, M. Ohki, and H. Ichikawa. Identification of a gene expression signature associated with pediatric AML prognosis. *Blood*, 102(5):1849–1856, 2003.

[293] J. Yi, S. Kloeker, C. Jensen, S. Bockholt, H. Honda, H. Hirai, and M. Beckerle. Members of the Zyxin family of LIM proteins interact with members of the p130Cas

family of signal transducers. *Journal of Biological Chemistry*, 277(11):9580–9589, 2002.

[294] L. A. Zadeh. Calculus of fuzzy restrictions. In L. A. Zadeh, K. S. Fu, K. Tanaka, and M. Shimura, editors, *Fuzzy Sets and Their Applications to Cognitive and Decision Processes*, pages 1–39. Academic Press, New York, 1975.

[295] S. M. Zola-Morgan and L. R. Squire. The primate hippocampal formation: Evidence for a time-limited role in memory storage. *Science*, 250:288–290, 1990.