COGNITIVE-INSPIRED APPROACHES TO NEURAL LOGIC NETWORK LEARNING

CHIA WAI KIT HENRY

(B.Sc.(Comp. Sci. & Info. Sys.)(Hons.), NUS;

M.Sc.(Research), NUS)

A THESIS SUBMITTED

FOR DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE

NATIONAL UNIVERSITY OF SINGAPORE

2010

To *my Family*

Acknowledgements

I would like to thank my advisor, **Professor Tan Chew Lim**. You have been a great motivator in times when I have felt completely lost. Your insightful advice and guidance have always fueled me to strive and persevere towards the final goal.

Many thanks to **Prof. Dong Jin Song**, **Dr. Colin Tan** and **Prof. Xiong Hui** who, as examiners of this thesis, has very kindly shared their ideas and opinions about the research in the course of my study.

My thanks also goes out to the Department of Computer Science which has allowed me to continue to pursue my doctoral degree while offering me an Instructorship to teach in the department.

Henry Chia

November 2010

Contents

Summary

The comprehensibility aspect of rule discovery is of much interest in the literature of knowledge discovery in databases. Many KDD systems are concerned primarily with the predictive, rather than the explanation capability of the systems. The discovery of comprehensible and human amenable knowledge requires the initial understanding of the cognitive processes that humans employ during decision making, particularly within the realm of cognitive psychology and behavioural decision science. This thesis identifies two such concepts for integration into existing data mining techniques: the language bias of human-like logic used in everyday decision and rational decision making. Human amenable logic can be realized using neural logic networks (*neulonets*) which are compositions of *net rules* that represent different decision processes, and are akin to common decision strategies identified in the realm of behavioral decision research. Each *net rule* is based on an elementary decision strategy in accordance to Kleene's three-valued logic where the input and output of *net rules* are ordered-pairs comprising values representing "true", "false" and "unknown". Other than these three "crisp" values, *neulonets* can also be enhanced to account for decision making under uncertainty by turning to its probabilistic variant where each value of the ordered pair represents a degree of truth and falsity. The notion of "rationality" in making rational decisions transpires in two forms: bounded rationality and ecological rationality. Bounded rationality entails the need to make decisions within limited constraints of time

and explanation capacity, while ecological rationality requires that decisions be adapted to the structure of its learning environment. Inspired by evolutionary cognitive psychology, *neulonets* can be evolved using genetic programming to form complex, yet boundedly rational, decisions. Moreover, ecological rationality can be realized when *neulonet* learning is performed under the context of niched evolution. The work described in this thesis aims to pave the way for endeavours in realizing a "cognitive-inspired" knowledge discovery system that is not only a good classification system, but also generates comprehensible rules which are useful to the human end users of the system.

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Pattern and knowledge discovery in data

With the proliferation of digital data being accumulated and collected from electronic records and through electronic transactions, the field of Knowledge Discovery in Databases (KDD) has seen unprecedented challenges in terms of making sense of the massive volume of data. KDD has been described as an overall process that comprises several components for discovering useful knowledge from data [Fayyad et al., 1996] with the two central ones being that of *data mining* — an application of specific algorithms for extracting patterns from data, as well as the *interpretation* component which relates to the novelty, utility and understandability of the mined patterns or rules. These two components are crucial as knowledge discovered should generally meet two goals [Fayyad et al., 1996] — the goal of *prediction* that pertains to the ability of a system in finding patterns that facilitates the forecast of future or unseen behaviour of related entities, as well as the goal of *description* that relates to how these discovered patterns can be presented to the user in a human-comprehensible form. Much research in data mining rely heavily on known techniques from machine learning, pattern recognition and statistics;

however, unmindful application of these techniques can only lead to the discovery of meaningless and invalid patterns or rules. Data mining research needs to set itself apart from similar endeavours in machine learning and pattern recognition by emphasizing on the discovery of understandable *patterns*. Only when interpretable pattern discovery is in place, would we turn to qualifying these patterns as useful or interesting *knowledge*.

To provide a formal definition of a *pattern*, we adopt the conceptual model outlined in [Frawley et al., 1992].

> Given a set of facts (data) F, a language L, and some measure of certainty C, we define a *pattern* as a statement S in L that describes relationships among a subset F_S of F with a certainty C, such that S is simpler (in some sense) than the enumeration of all facts in F_S.

The lithe concept of a *language* above is useful as it does not limit itself to mere worded patterns, but supports a myriad of data mining models ranging from simple decision trees to the more complex neural network models. Among the different data mining models that can be used to describe relationships in data as *patterns* [Fayyad et al., 1996], *classification rules* are by far one of the most interpretable and intuitive representations. Indeed, the famous work on the General Problem Solver [Newell and Simon, 1972] demonstrated that human problem solving could be effectively expressed using *if... then* production rules. These are prediction rules where the rule antecedent (or *if* part) consists of a combination of predicting attribute values conditioned on some propositional logic (usually a conjunct), and the rule consequent (or *then* part) consists of a predicted value for the goal (or class) attribute. These rules are commonly adopted in rule-based expert systems, and rule discovery systems that induce conjunctive rules either directly from data (e.g. the AQ family of algorithms [Michalski et al., 1986]) or

indirectly from knowledge models generated from data (e.g. a decision tree model in C4.5 [Quinlan, 1993]). In contrast to the definition of a *pattern* above, Frawley *et al.* also defined *knowledge* as follows [Frawley et al., 1992].

> A pattern S that is interesting (according to a user-imposed interest measure) and certain enough (again according to the user's criteria) is called *knowledge*. The output of a program that monitors the set of facts in a database and produces patterns in this sense is *discovered knowledge*.

It is clear that *pattern/rule* discovery precedes *knowledge* discovery. Elevating the status of patterns to become practical knowledge involves a more subtle requirement for patterns to be novel and potentially useful to the user of the system. Interestingness [Piatetsky-Shapiro and Matheus, 1994, Silberschatz and Tuzhilin, 1995] is usually taken as an overall measure of pattern value, combining validity, novelty, usefulness, and simplicity. The literature on measures of interestingness is extensive (see [Freitas, 1999] for a survey and [Tan et al., 2002] for some examples of interestingness measures), ranging from objective to subjective measures. Subjective measures are less favourable to their objective counterparts as the former rely on subjective human judgement as the sole evaluation measure. As for objective measures based on statistical significance, three general unbiased considerations are usually taken into account: the coverage, completeness and predictive accuracy of the rule. Briefly, the *coverage* of a rule is the number of data examples satisfied by the rule antecedent regardless of its consequent target class; the *completeness* of a rule is the proportion of data examples of the target class covered by the rule; while *predictive accuracy* is the proportion of covered rules correctly predicted by the target class.

Comprehensibility (or interpretability) is another important ingredient when considering the practicality and usefulness of knowledge. Comprehensibility can be estimated by simplicity of a pattern (or rule) such as the number of bits to describe a pattern. However, it should be noted that simplicity can only be considered as a proxy for comprehensibility [Domingos, 1999]. Frawley's definition of a *pattern* also indicates that rule interpretability could be enhanced further by adopting a richer *language* using concepts that are more human amenable so as to facilitate the formulation of much more expressive rules. The enhancement of rule interpretability by injecting human amenable concepts into a richer language base for rule description facilitates the formulation of more expressive rules. This would then lead to improved *knowledge* discovery as subsequent application of prevailing research in quantifying rule interestingness can be applied seamlessly to the set of richer patterns.

1.2 Human amenable concepts

As highlighted at the beginning, *if... then* classification rules based on conjuncts are extensively used in rule-based production systems and, more recently, in association-rule based mining [Agrawal et al., 1993] and association-rule based classification methods [Freitas, 2000]. Without loss of generality, we represent these classification rules in the form $A \Rightarrow B$, where A is some antecedent evaluation on a set of predicting attribute values and B is the predicted class. Other than purely conjunctive rules where A is typically of the form $(a_1 \wedge a_2 \wedge \cdots \wedge a_k)$ for k predicting factors or attributes, other rule induction systems have also been adapted to use the boolean logic operators (AND/OR/NOT) as evident in decision lists [Rivest, 1987] and linear machine decision trees [Utgoff and Brodley, 1991].

Other than standard boolean logic, other "human amenable" logic can be used instead. In his article to *IEEE Intelligent Systems*, [Pazzani, 2000] stressed on the need to account for the cognitive processes that humans employ during decision making. He suggested that KDD should amalgamate cognitive psychology with artificial intelligence (machine learning), databases and statistics to create models that people would find intelligible as well as insightful. Of course, this does not necessarily mean that KDD systems should emulate the exact behavior of how people learn from data since inherent cognitive limitations prevent most people from finding subtle patterns in a reasonable amount of data. Instead, the key consideration for emulating human decision making should account for the biases of human learners which play a pertinent role toward the acceptance of knowledge acquired through data mining. A particular form of such bias (or more specifically, *language* bias) is the myriad of ways in which composite predicting attributes can be conditioned and constructed, *viz.* the different ways in which attributes in a rule antecedent can be combined with rudimentary concepts (other than pure conjuncts, disjuncts and negations) which would suit the language bias of human learning.

A motivating example comes from the *m-of-n* concept [Fisher and McKusick, 1989] which can be viewed as a generalization of the conjunctive concept where m and n are equal. Although *m-of-n* concepts can be expressed in terms of conjunctions and disjunctions, they are not easily interpretable even as decision trees; for example, a single 4-*of*-8 concept constitutes 70 conjunctive terms. [Murphy and Pazzani, 1991] showed that a bias towards this alternative concept is useful for decision tree learning, and its successful application is apparent even in rule extraction from neural networks [Towell and Shavlik, 1994, Towell and Shavlik, 1993]. This *m-of-n* concept provides a strong impetus into the use of similar concepts that are equally

amenable towards human understanding. To illustrate, we could extend the *m-of-n* concept to an *at-least-m-of-n* concept which provides a more extensive enumeration but nonetheless remains comprehensible. Using the same example above, a single *at-least*-4-*of*-8 concept is equivalent to 162 conjunctive terms. Following up on Pazzani's proposal on injecting insights from other disciplines into knowledge discovery, we turn to current trends of human decision making in cognitive science research and consider how these concepts can be modeled. Human decision-making has been extensively studied in the areas of psychology and economics. In particular, behavioral decision research draws insights from both arena (and even other disciplines including statistics and biology) to determine how people make judgments and choices, as well as the processes of decision making [Payne et al., 1998]. A wide variety of decision strategies have since been identified across various fields, a majority of which caters to preferential decision making, i.e. choosing the best of several alternatives available to the decision maker. Here, we provide a glimpse of some of these heuristics which are collectively described in [Payne et al., 1993].

- The *weighted additive heuristic* establishes the value of each alternative as the sum of *all* attributes scaled with their corresponding weights. The alternative with the highest value among all alternatives is chosen.

- The *equal weight heuristic* is a special case of the *weighted additive heuristic* where the weights of the attributes are similar, thereby ignoring information about the relative importance or probability of each attribute.

- The *satisficing heuristic* considers alternatives one at a time, stopping at and selecting the first alternative with all its attribute values above their corresponding pre-defined (or pre-computed) cutoff levels.

- The *lexicographic heuristic* determines the most important attribute and examines values of all alternatives on that attribute. The alternative with the

best value is chosen. In the case of a tie, the procedure repeats by examining the next important attribute among the tied alternatives, until a single alternative with an outright optimal value has been identified.

- The *elimination-by-aspects heuristic* begins with determination of the most important attribute and its cutoff value retrieved or computed. All alternatives with values for that attribute below the cutoff are eliminated and the process is repeated with the next important attribute until only one alternative remains.

- The *majority of confirming dimensions* heuristic processes pairs of alternatives. The values for each of two alternatives are compared on each attribute, and the one with a majority of winning attribute values is retained and compared with the next alternative. The final winning alternative is chosen.

- The *frequency of good and bad features* heuristic counts the number of good and bad features (based on some cutoff values) in each alternative. Depending upon whether good, bad or both features are considered, different variants of the heuristic would arise. This heuristic was inspired by the voting rule where attributes can be viewed as the voters.

- The *combined strategies*. Individuals sometimes use combinations of strategies. For example, one might choose an elimination heuristic first to remove poor alternatives, followed by a heuristic that examines and compares the attributes of the remaining alternative more closely.

- *Other heuristics* that are far simpler have also been proposed. For example, the strategy of the habitual heuristic is simply to choose what one chose last time.

Although instinctive, the above seemingly innocuous heuristics give rise to a major

problem when attempting to realize them using a computational machine learning algorithm. Notice that many of these heuristics require the explicit specification of "weight" or "value" for each choice to be evaluated, prior to the application of the strategies. Moreover, the strategies are stipulated using linguistics terms such as "similar", "good", "bad" which are verbal expressions of uncertainty. The general way to deal with these verbal probability terms and phrases is to model them via numerical translations using probabilistic models or fuzzy set membership functions. The main argument in doing so is that the vagueness and flexibility in such linguistic terms and phrases, when translated to numerical connotations, produces a range of values or probabilities that might be considered as acceptable referents by users. As an example the quantifying term "several" may refer to integers from say, about 4 to 10. This idea have been adopted fairly widely by researchers because vagueness (or ambiguity) elicits responses in people that have decisional consequences. In the context of rule-based knowledge discovery in particular, it was observed that numerical representations of uncertainty-related information are more likely to facilitate the invocation of rule-based processing [Windschitl and Wells, 1996].

Moreover, knowledge within the real world is often incomplete; yet decisions must still be made from them. Behavioural studies in cognitive psychology have consistently found that uncertainty has a large influence on behaviour, with assessments of uncertainty being made in different forms throughout the literature of cognitive psychology [Kahneman and Tversky, 1982]. These include focusing uncertainty on statistical frequencies, subjective propensity or disposition, confidence associated with judgments, and strength of arguments. More recently, the conceptualization of uncertainty has also been attempted [Lipshitz and Strauss, 1997] due to the myriad definitions of uncertainty, and synonymous terms such as risk, ambiguity,

turbulence, equivocality, conflict, etc. Attempts to inject cognitive learning with uncertainty into Artificial Intelligence has seen many approaches in managing uncertainty. These include the classical mathematical theory of probability, as well as other numerical calculi for the explicit representation of uncertainty such as certainty factors and fuzzy measures [Shafer and Pearl, 1990].

There is also strong evidence of an intrinsic cognitive propensity of three-valued logic with the possibility of leaving some propositions undecided, thereby implying the independence of truth and falsity. This is in contrast to the the traditional probabilistic view that the truth value is one minus its falsity, and vice versa. This gives strong impetus to move towards (strong Kleene) three-valued logic [Kleene, 1952] which comprises three values: *truth*, *falsity* and *unknown*. Uncertainty with respect to Kleene's three-valued logic can be realized with an ordered-pair system with values represented by (t, f) with the constraint $t + f \leq 1$ [Stenning and van Lambalgen, 2008]

The first value t represents the degree of truth, while the second value f represents the degree of falsity. In this way, complete truth is denoted by $(1, 0)$, complete falsity is denoted by $(0, 1)$, while total ignorance is accorded $(0, 0)$. Note that the value $(1, 1)$ may sometimes be used to represent a contradiction or be undefined. Moreover, a situation in which we have an equal amount of arguments in favor of and against a proposition can also be represented, such as $(0.3, 0.3)$.

1.3 Rationality in decision making

Another significant contribution from Payne and colleagues' work on heuristics and strategies in decision making is attributed to the idea that an individual is *adaptive* by nature when making a decision [Payne et al., 1993]. This stems from

the observation that one chooses a strategy based on an effort-accuracy tradeoff within some usage context or environment. Their proposed theoretical framework for understanding how people decide which decision strategy to use in solving a particular judgment problem is based on the following five major assumptions.

1. People have available a repertoire of strategies or heuristics for solving decision problems of any complexity.

2. The available strategies are assumed to have differing costs (e.g. cognitive effort) and benefits (e.g. predictive accuracy) with respect to the individuals' goals and the constraints associated with the structure and content of any specific decision problem.

3. Different tasks are assumed to have properties that affect the relative costs and benefits of the various strategies. The second and third assumptions together contributes to the *no free lunch* notion, i.e. a given strategy may look relatively more attractive than other strategies in some environments and relatively less attractive than those same strategies in other environments.

4. An individual selects the strategy that one anticipates as the best for the task.

5. People use both a top-down and/or bottom-up view of strategy selection. In the top-down view, a priori information or perceptions of the task and strategies determine the subsequent strategy selection. But according to bottom-up (or data driven) approaches, subsequent actions are more influenced by the data encountered during the process of decision making than by a priori notions.

Strategy selection is a result of the compromise between the desire to make the most correct decision and the desire to minimize effort. Choosing a strategy is then

based upon an effort-accuracy tradeoff that depends on the relative weight a decision maker places on the goal of making an accurate decision versus saving cognitive effort. Payne and colleagues based their study of preference choice on hypothetical situations and randomly generated gambles, with results of predictive accuracies of various strategies measured against the traditional gold standard given by the weighted additive rule. A similar endeavor of *fast and frugal heuristics* by Gigerenzer, Todd and the ABC (Centre for Adaptive Behavior and Cognition) research group [Gigerenzer and Todd, 1999], on the other hand, focused on inferences with real world instances using measurements obtained from external real-world criteria. This latter work is imbued in the principle of *bounded rationality* — a school-of-thought in decision-making made famous by Herbert A. Simon [Simon, 1955], and has been effectively used to argue against *demonic* rational inference whereby the mind is viewed as "if it were a supernatural being possessing demonic powers of reason, boundless knowledge, and all of eternity with which to make decisions". Such an unrealistic and impractical view disregards any consideration of limited time, knowledge, or computational capacities. The major weakness of unbounded rationality is that it does not describe the way real people think since the search for the optimal solution can go on indefinitely.

Instead of devoting to demonic visions of rationality, the idea of bounded rationality replaces the omniscient mind of computing intricate probabilities and utilities with an adaptive toolbox filled with fast and frugal heuristics [Gigerenzer, 2001]. The premise is that much of human reasoning and decision making should be modeled with considerations of limited time, knowledge and computation tractability. One form of bounded rationality is Simon's concept of *satisficing* which we earlier discussed as one of the heuristics of behavioral decision making. To reiterate, a satisficing heuristic is a method for making a choice from a set of alternatives

encountered sequentially when one does not know much about the possibilities in advance. In such situations, there may be no optimal method for stopping search for further alternatives. Satisficing takes the short cut of setting an aspiration level and ending the search for alternatives as soon as one is found that exceeds the aspiration level. However, the setting of an appropriate aspiration level might entail a large amount of deliberation on the part of the decision maker. In contrast, *fast and frugal heuristics* represents the "purest" form of bounded rationality that employ a minimum of time, knowledge and computation to make adaptive choices in real environments. They limit their search of objects or information using easily computable stopping rules, and make their choices with easily computable decision rules. One such heuristic, Take the Best (TTB), chooses one of two alternatives by looking for cues and stopping as soon as one is found that discriminates between the two options being considered. The search for cues is in the order of their validity, i.e. how often the cue has indicated the correct versus the incorrect options. A similar heuristic, Take the Last, is even more frugal since cues are searched in the order determined by their past success in stopping search.

Other than the aspect of limited cognitive capacities, there is yet another equally important aspect of bounded rationality that has unfortunately been neglected in mainstream cognitive science. Specifically, we turn to Herbert Simon's original vision of bounded rationality as etched in the following analogy [Simon, 1990].

> Human rational behavior (and the rational behavior of all physical symbol systems) is shaped by a scissors whose two blades are the structure of task environments and the computational capabilities of the actor.

Clearly, Simon's scissors consists of two interlocking components: the limitations of the human mind, and the structure of the environments in which the mind operates. The first customary component of Simon's scissors indicates that models of

human judgment and decision making should be built on what we actually know about the mind's capacities rather than on fictitious competencies. Additionally, the second component of environment structure is of crucial importance because it can explain when and why heuristics perform well: if the structure of the heuristic is adapted to that of the environment. In this respect, the research program of fast and frugal heuristics tasks to "bring environmental structure back into bounded rationality" by supplementing bounded rationality with the concept of "ecological rationality".

Decision making mechanisms can be matched to particular kinds of task by exploiting the structure of information in their respective environments so as to arrive at more adaptively useful outcomes. A decision making mechanism is ecologically rational to the extent that it is adapted to the inherent structure of its environment. In other words, unlike the measure of cognitive effort in a heuristic, ecological rationality is not a feature of a heuristic *per se*, but a consequence of a match between heuristic and environment. In [Gigerenzer and Todd, 1999], simple heuristics has been shown to work well despite their frugal nature due to a trade-off in another dimension: that of generality versus specificity. To put the argument in perspective, we first assume a general-purpose computing machine, say a neural network. It is a general learning and inferencing strategy in the sense that the same neurologically-inspired architecture can be used across virtually any environment. Moreover, it can be highly focused by tuning its large number of free parameters to fit a specific environment. However, this immense capacity to fit can be a hindrance since overfitting can lead to poor generalization. On the other hand, simple heuristics are meant to apply to specific environments, but they do not contain enough detail to match any one environment precisely. A heuristic that works to make quick and accurate inferences in one domain may well not

work in another. Thus, different environments can have different specific heuristics that exploit their particular information structure to make adaptive decisions. But specificity can also be a danger: if a different heuristic was required for every slightly different decision-making environment, we would need an unthinkable multitude of heuristics to reason with, and we would not be able to generalize to previously unencountered environments. Simple heuristics avoid this trap by their very simplicity, which allows them to be robust when confronted by environmental change and enables them to generalize well to new situations.

The need to act quickly when making rational (both bounded and ecological) decisions is no doubt desirable. However, if simple rules do not lead to appropriate actions within their environments, then they are not adaptive; in other words, they are not ecologically valid. According to [Forster, 1999], the requirement of ecological validity goes beyond the pragmatic requirements of speed and frugality. Fast and frugal heuristics emerge from an underlying complexity in a process of evolution that is anything but fast and frugal, and these complexities are necessary in order to satisfy the requirement of ecological rationality which would otherwise be implausible. There could even be some conservation law for complexity that says that simplicity achieved at the higher level is at the expense of complexity at a lower level of implementation. Gigerenzer later also concurred that heuristics are adaptations that have evolved by natural selection to perform important mental computations efficiently and effectively [Gigerenzer, 2002]. This idea of evolving heuristics has already been investigated extensively in the realm of evolutionary psychology, and is in line with the notion of ecological rationality.

1.4 Evolutionary psychology

Evolutionary psychology uses concepts from biology to study and understand human behavior and decision making. In this respect, the mind is viewed as a set of information-processing machines that were designed by Darwinian natural selection to solve adaptive problems [Cosmides and Tooby, 1997]. Psychologists have long known that the human mind contains neural circuits that are specialized for different modes of perception, such as vision and hearing. But cognitive functions of learning, reasoning and decision making were thought to be accomplished by circuits that are very general purpose. In actual fact, the flexibility of human reasoning, in terms of our ability to solve many different kinds of problems, was thought to be evidence for the generality of such circuits.

Evolutionary psychology, however, suggests otherwise. According to the groundbreaking work of [Tooby and Cosmides, 1992], biological machines are calibrated to the environments in which they evolved, and these evolved problem solvers are equipped with "crib sheets", so they come to a problem already "knowing" a lot about it. For example, a newborn's brain has response systems that "expect" faces to be present in the environment: babies less than 10 minutes old turn their eyes and head in response to face-like patterns. But where does this "hypothesis" about faces come about? Without a crib sheet about faces, a developing child could not proceed to learn much about its environment. Different problems require different crib sheets. Indeed, having two machines is better than one when the crib sheet that helps solves problems in one domain is misleading in another. This suggests that many evolved computational mechanisms will be domain-specific, i.e. they will be activated in some domains but not others; this is in stark contrast to the domain-general view of general-purpose cognitive systems that applies to problems in any domain. The more crib sheets a system has, the more problems it can solve,

and a brain equipped with a multiplicity of specialized inference engines will be able to generate sophisticated behavior that is sensitively tuned to its environment.

One of the principles of evolutionary psychology states that "our neural circuits were designed by natural selection to solve problems that our ancestor faced during our species' evolutionary history" [Cosmides and Tooby, 1997]. So the function of our brain is to generate behavior that is appropriate to our environmental circumstances. Appropriateness has different meanings for different organisms. Citing the example of dung, for us a pile of dung is disgusting, but for a female dung fly looking for a good environment to raise her children, the same pile of dung is a beautiful vision. Our neural circuits were designed by the evolutionary process, and natural selection is the only evolutionary force that is capable of creating complexly organized machines. Natural selection does not work "for the good of the species", as many people think. It is a process in which a phenotypic design feature causes its own spread through a population which might either lead to permanence or extinction. Using the same example, an ancestral human who had neural circuits that make dung smell sweet would get sick more easily and even die. In contrast, a person with neural circuits that made him avoid dung would get sick less often, and have a longer life. As a result, the dung-eater will have fewer children than the dung-avoider. Since the neural circuitry of children tends to resemble that of their parents, there will be fewer dung-eaters in the next generation, and more dung-avoiders. As this process continues through the generations, the dung-eaters will eventually disappear leaving us — the descendents of dung-avoiders. No one is left who has neural circuits that make dung delicious. In essence, the reason we have one set of circuits rather than another is that the circuits that we have were better at solving problems that our ancestors faced during our species' evolutionary history than alternative circuits were.

As witnessed above, evolutionary psychology is grounded in ecological rationality, since it assumes that our minds were designed by natural selection to fit a myriad of domains (environments) using separate crib sheets (heuristics) to efficiently and effectively solve practical problems in each of them. While evolutionary psychology focuses specifically on ancestral environments and practical problems with fitness consequences, the concepts of bounded and ecological rationality can additionally encompass decision making in present environments. Despite that the evolutionary psychology program with its associated claim of massive mental modularity, and the *fast and frugal heuristics* movement with its claim of an adaptive toolbox of simple cognitive procedures have independently developed over the last couple of decades, these programs should generally be seen as mutual supporters rather than as competing programs [Carruthers, 2006].

1.5 Summary

We started off this chapter within the realm of data mining, deliberating over issues of rule comprehensibility in classification rule generation. In order to broaden our view, we crossed-over to the domain of psychology and cognitive science, mulling over heuristics and strategies of decision making as well as considerations of uncertainty handling. This inadvertently led us to consider their usefulness as well as adaptiveness in the context of bounded and ecological rationality. It is appropriate that we now retract back to our proverbial domain and consider the relevance of the two rationality notions in the context of learning machines.

It comes as no surprise that the same Darwinian underpinnings of evolutionary psychology is instilled in evolutionary approaches of machine learning such as genetic algorithms and genetic programming. Interestingly, there have been attempts to

model bounded rationality with evolutionary techniques. However, this concept of rationality is explored with respect to different parts of the system. For example, in the modeling of bounded rational economic agents [Edmonds, 1999], an agent is given limited information about the environment, limited computation power, and limited memory. In contrast, the work of [Manson, 2005] examines genetic programming parameters which includes population size, fitness measures, number of generations, etc. with respect to collaries of bounded rationality related to memory, bounding of computation resources, complexities that evolved programs can achieve, etc. Ecological rationality, on the other hand, is almost never mentioned in the literature of evolutionary computation as its associated aspect of adaptiveness to the environment is inherent in such approaches; the environment being generally the problem domain (or training data set in the case of supervised learning).

In the above discussion, we have generally identified two major aspects of cognitive science that can be integrated into a model of machine learning for rule-based knowledge discovery. Firstly, to model a repertoire of simple, bounded rational heuristics for decision making using Neural Logic Networks [Teh, 1995], including the use of its probabilistic variants for tackling issues of uncertainty; and secondly, the adaptation of these heuristics within the notion of ecological rationality and it's fit to the environment using the evolutionary learning paradigm of Genetic Programming [Koza, 1992]. The thesis is formally stated as follows:

> To realize a cognitive-inspired knowledge discovery system through the evolution of rudimentary decision operators on crisp and probabilistic variants of Neural Logic Networks under a genetic programming evolutionary platform.

Chapter 2 of this proposal will be set aside as a primer for neural logic networks

and their analogy to models of heuristics and strategies in decision making. Chapter 3 focuses on the rudiments of evolutionary learning with neural logic networks by describing the learning algorithm with emphasis on the genetic operations and the fitness measure. The aspect of rule extraction will also be discussed. Chapter 4 details continuing work on evolutionary neural network learning within an association-based classification platform as an initial exploration towards niched evolution. We show how the notions of support and confidence of association rule mining can be extended to the fitness function. Chapter 5 addresses the issue of niched evolution in greater detail with an alternative evolutionary paradigm. Moreover, we consider the probabilistic variant of neural logic network and how it can be used to in rule discovery on continuous valued data. Chapter 6 concludes with an overview of the cognitive issues involved in the research.

Chapter 2

The Neural Logic Network

Neural Logic Network (or *Neulonet*) learning is an amalgam of neural network and expert system concepts [Teh, 1995, Tan et al., 1996]. Its novelty lies in its ability to address the language bias of human learners by simulating human-like decision logic with rudimentary *net rules*. Some of the human amenable concepts emulated include the *priority* operation that depicts the notion of assigning varying degrees of bias to different decision factors, and the *majority* operation that involves some strategy of vote-counting. Within the realm of data mining and rule discovery, these richer logic operations supplement the standard boolean logic operations of conjunction, disjunction and negation, so as to allow for a neater and more human-like expression of complex logic in a given problem domain. This chapter is devoted as a primer on Neural Logic Networks (*Neulonet*) with detailed analysis of its characteristics, properties, semantics and basic operations. By composing rudimentary *net rules* together, the effectiveness of a *neulonet* as a basic building block in network construction will be highlighted.

2.1 Definition of a neural logic network

A *neulonet* differs from other neural networks in that it has an ordered pair of numbers associated with each node and connection as shown in figure 2.1. Let Q be the output node and P_1, P_2, $\ldots P_N$ be input nodes. Let values associated with the node P_i be denoted by $(a_i,\ b_i)$, and the weight for the connection from P_i to Q be $(\alpha_i,\ \beta_i)$. The ordered pair for each node takes one of three values, namely, (1,0) for "true", (0,1) for "false" and (0,0) for "don't know". (1,1) is undefined. Equation (2.1) defines the activation function at the output Q with λ as the threshold, usually set to 1.

$$Act(Q) = \begin{cases} (1,0) & \text{if } \sum_{i=1}^{N}(a_i\alpha_i - b_i\beta_i) \geq \lambda \\ (0,1) & \text{if } \sum_{i=1}^{N}(a_i\alpha_i - b_i\beta_i) \leq -\lambda \\ (0,0) & \text{otherwise.} \end{cases} \tag{2.1}$$

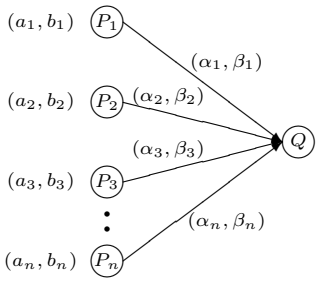

Figure 2.1: Schema of a Neural Logic Network - *Neulonet*.

Due to this ordered-pair system for specifying activation values and weights, we can formulate a wide variety of richer logic that are often too complex to be expressed neatly using standard boolean logic. These can be represented using rudimentary *neulonets* with different sets of connecting weights. We generally divide these *net rules* into five broad categories as shown in figure 2.2.

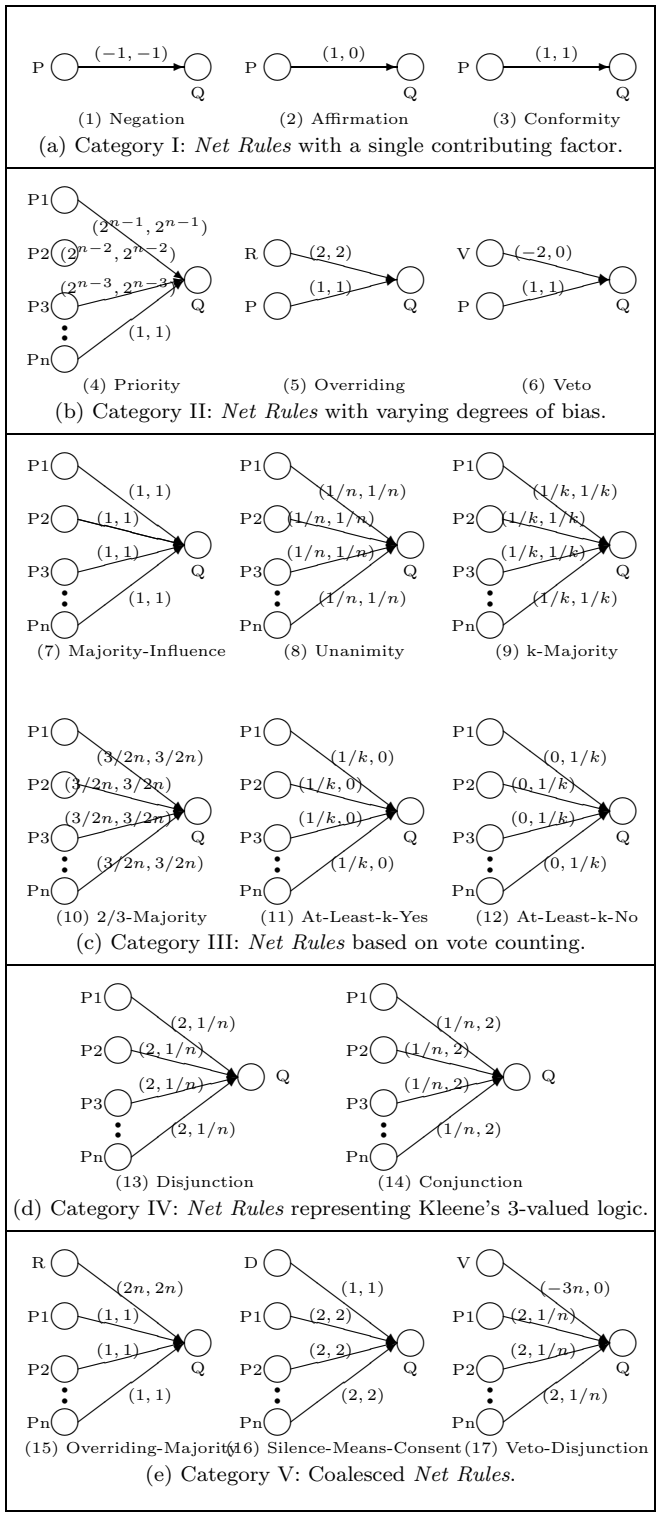

Figure 2.2: Library of *net rules* divided into five broad categories

- *Category I - Single-Decision Net Rules*

  As the name implies, each of the *net rules* in this category is dependent on a single contributing factor as shown in figure 2.2(a). They represent the simplest form of net rules. As an example, rule (1) *Negation* simply outputs a true decision if the input is false and vice-versa. An unknown input results in an unknown output.

- *Category II - Variable-bias Net Rules*

  As shown in figure 2.2(b), each *net rule* is indicated by the varying degrees of importance given to the different contributing factors. For example, rule (5) *Overriding* assigns greater influence to the overriding factor R. The outcome Q will follow R only if the latter is deterministic. Otherwise, Q respects the default decision factor P when R is unknown.

- *Category III - Net Rules Based on Vote-Counting*

  In many situations where every decision maker has equal influence on the outcome of decision, the strategy of vote-counting is often employed to assign the final outcome. It is apparent from figure 2.2(c) that there are different vote-counting schemes. For example, in rule (7) *MajorityInfluence*, the outcome of the decision will depend on the majority of all the votes, while in rule (8) *Unanimity*, a deterministic outcome results only in a common vote among all decision makers.

- *Category IV - Three-Valued Logic Net Rules*

  In order to represent the standard boolean AND and OR logic using *net rule* syntax, the alternative Kleene's three-valued (true, false and unknown) logic system [Kleene, 1952] for conjunction and disjunction is adopted as shown in figure 2.2(d). One example of Kleene's logic system for the *Disjunction* operation (rule 13) is shown in figure 2.3. In a way, it is similar to standard

OR logic, the outcome will be true provided any input is true. However, the outcome is false only if all inputs are false. For the other cases where only some of the inputs are false while the rest are unknown, the outcome remains unknown.

P	Q	P or Q
(1,0)	(1,0)	(1,0)
(1,0)	(0,1)	(1,0)
(1,0)	(0,0)	(1,0)
(0,1)	(1,0)	(1,0)
(0,1)	(0,1)	(0,1)
(0,1)	**(0,0)**	**(0,0)**
(0,0)	(1,0)	(1,0)
(0,0)	**(0,1)**	**(0,0)**
(0,0)	**(0,0)**	**(0,0)**

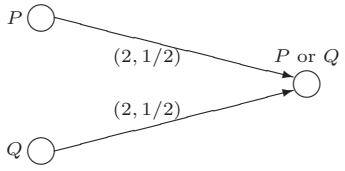

Figure 2.3: *Neulonet* behaving like an "OR" rule in Kleene's logic system.

- *Category V - Coalesced Net Rules*

  This last category of *net rules* in figure 2.2(e) illustrates the ability to represent simple compositions of other net rules using an appropriate set of weights. For example, rule (17) *Veto–Disjunction* comprises constituent rules (6) *Veto* and (13) *Disjunction* where V is the veto decision factor. The semantics of the *Veto* rule is such that only when V is true would it provide the veto power to coerce the outcome Q to be false. If this decision factor is not true, then the outcome will take on the default decision which, in the case of *Veto–Disjunction*, takes on the disjunct of the remaining factors.

2.2 *Net rules* as decision heuristics

Comparing the classes of *net rules* above, we observe a striking resemblance with the heuristics and strategies defined in Section 1.2. Indeed, close scrutiny on the heuristics identified from cognitive science leads us to believe that such heuristics are useful due also to their richness in logic expression. For example, the heuristic of *satisficing* and *elimination-by-aspects* (including the strategy of *Take-the-Best* from

the *fast-and-frugal heuristics* program) have an implicit ordering of importance (or bias) given to the attributes. This is analogous to our bias-based *Overriding* (or more generally *Priority*) *net rule*. In addition, the notion of frequency-counting also underlies the effectiveness of the *majority of confirming dimensions* and *frequency of good and bad features* heuristics identified in human behavioral research, as is the case of category III *net rules*. Furthermore, our library of *net rules* mirror the first assumption of Payne's theoretical framework for understanding human decision making, i.e. people have a repertoire of common heuristics at their dispense when making specific decisions. It is apparent that the library of *net rules* has been painstakingly devised to encompass very much the same spirit as its psychological counterpart, despite the fact there being no mention of any reference to psychological studies. Our innate ability to handle natural frequency counts, as well as assigning bias, make these human decision logic seem all the more viable. These *net rules* are also *boundedly rational* as they do not necessitate that all attributes of the problem domain be checked. Moreover, bias-based *net rules* such as *Priority* is inherently *satisficing*, i.e. a decision can be reached early if a strongly biased (or strongly weighted) contributing factor gives a deterministic signal. Indeed the uncanny parallelism between *net rules* and decision heuristics suggests that we could devise a whole lot more useful rudimentary *net rules* to mimic the actual decision strategies that humans employ.

It should also be noted that a desirable property in using neural logic networks for data classification is the fact that missing or unknown values, which occur frequently in many real-world data sets, are taken care of without the need for any external intervention. Another important property is that *net rules* can be combined in a myriad of ways to form composite *neulonets* to realize more complex decisions. As a simple illustration, figure 2.4 shows how rules (8) *Unanimity*

and (17) *Veto–Disjunction* can be composed to form the XOR operation following Kleene's three-valued logic model. The final outcome would be driven false as long as both P and Q are unanimously true. Otherwise, the outcome takes on the disjunct of P and Q as in rule (13).

P	Q	P xor Q
(1,0)	**(1,0)**	**(0,1)**
(1,0)	(0,1)	(1,0)
(1,0)	(0,0)	(1,0)
(0,1)	(1,0)	(1,0)
(0,1)	(0,1)	(0,1)
(0,1)	(0,0)	(0,0)
(0,0)	(1,0)	(1,0)
(0,0)	(0,1)	(0,0)
(0,0)	(0,0)	(0,0)

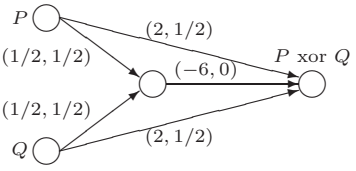

Figure 2.4: An "XOR" composite *net rule*.

We present a more concrete example using the Space Shuttle Landing Database [Michie, 1988]. The decision to use the autolander or to control the spacecraft manually at the last stage of descent is taken on the basis of information about a number of attributes such as visibility, errors of measurement, stability of the craft, a nose-up or nose-down attitude, the presence of head wind or tail wind, and the magnitude of the atmospheric turbulence. This data set comprises 15 instances and 6 attributes. Table 2.1 shows each instance being classified as either to auto-land or not. To conform to the input requirements of the *neulonet* structure, every distinct attribute–value pair has a corresponding boolean attribute in a transformed data set. The valid values for these new attributes can either be *yes*, *no* or *unknown*. In our example, the six-attribute data set transforms to a 16-attribute data set of boolean values. The composite *neulonet* solution that correctly classifies all the training examples is given in figure 2.5. We shall analyze the rule extraction process in the following section.

Table 2.1: The Space Shuttle Landing data set

Stable	Error	Sign	Wind	Magnitude	Vis	Class
?	?	?	?	?	no	auto
xstab	?	?	?	?	yes	noauto
stab	LX	?	?	?	yes	noauto
stab	XL	?	?	?	yes	noauto
stab	MM	nn	tail	?	yes	noauto
?	?	?	?	OutOfRange	yes	noauto
stab	SS	?	?	Low	yes	auto
stab	SS	?	?	Medium	yes	auto
stab	SS	?	?	Strong	yes	auto
stab	MM	pp	head	Low	yes	auto
stab	MM	pp	head	Medium	yes	auto
stab	MM	pp	tail	Low	yes	auto
stab	MM	pp	tail	Medium	yes	auto
stab	MM	pp	head	Strong	yes	noauto
stab	MM	pp	tail	Strong	yes	auto

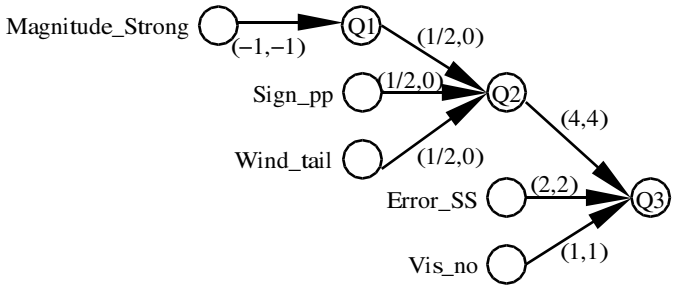

Figure 2.5: A solution to the Shuttle-Landing classification problem.

2.3 Rule extraction

Rule extraction is a straightforward process because the *neulonets* constructed are just a composition of *net rules* which by themselves fully express the human logic in use. *Net rules* can be composed using a genetic programming learning paradigm as described in chapter 3 of this thesis. Since the weights of the *net rules* remain unchanged, these *net rules* can be easily identified from any *neulonet*. Rule extraction performed on the Space Shuttle Landing problem based on the *neulonet* of figure 2.5 is shown in table 2.2.

Table 2.2: Extracted *net rules* from the shuttle landing data.

Q1 $\Leftarrow$ Negation(Magnitude_Strong)
Q2 $\Leftarrow$ At-Least-Two-Yes(**Q1**, Sign_pp, Wind_tail)
Q3 $\Leftarrow$ Priority(**Q2**, Error_SS, Visibility_no)

Observe that the "Priority" *net rule* **Q3** is biased towards *net rule* **Q2** which is of relatively higher importance as compared to the other two decision factors. The contributing factors that constitute *net rule* **Q2** can also be viewed as belonging to the same class of decision makers where each member in the group plays an equally important role in the outcome. Furthermore, *net rule* **Q2** provides us with a concise way of expressing the appropriate rule activations, rather than enumerating every possible combination of decision factors for each rule activation as in the case of production rules. In layman terms, the decision to auto-land the space shuttle is biased towards any two (or more) factors relating to the spacecraft adopting a nose-up (or climbing) attitude in order to decrease the rate of descent(i.e. "pp" sign), a tail wind, and the absence of strong turbulence. Otherwise, the presence (absence) of an "SS" error of measurement will result in a decision to auto-land (manual-land) the shuttle. If this error is unknown, then the obvious decision is

to auto-land the shuttle if the pilot cannot see; or manually land the spacecraft if visibility is clear.

It is important that we address a blatant misnomer before proceeding further. In much of the literature of rule extraction, each extracted rule is seen as a *functional* component that is also independently *meaningful*. However, the "rules" as shown in table 2.2 violate at least one of these properties. Notice that *net rules* **Q2** and **Q3** cannot act alone, i.e. they are not *functional* nor *meaningful*. Moreover, **Q1**, though meaningful, is still not *functional* as this "rule" alone does not address any part of the inherent logic of the problem. In a way, we should look at all three constituent *net rules* as a single rule, which would then satisfy the two properties. So in effect, we only extract one rule from a given *neulonet*; while this rule is made up of constituent *net rules*. In the literature of rule extraction from artificial neural networks, many researchers believe that an explanation capability should become an integral part of the functionality of any neural network training methodology [Andrews et al., 1995]. Moreover, the quality of the explanations delivered is also an important consideration. We adopted the following three criteria, extracted from [Andrews et al., 1995, Craven and Shavlik, 1999], for evaluating rule quality:

- *Accuracy*: The ability of extracted representations to make accurate predictions on previously unseen cases.
- *Fidelity*: The extent to which extracted representations accurately model the networks from which they were extracted.
- *Comprehensibility*: The extent to which extracted representations are humanly comprehensible.

Neulonet rule extraction belongs to the class of *decompositional* techniques in which the extraction of *net rules* is at the level of individual units within the neural network. As such, one can say that the view of the underlying *neulonet* is one of *transparency*. As we have seen, each unit in the *neulonet* structure can be directly interpreted as a constituent logic *net rule*. This is in contrast to other rule extraction techniques in which the neural network is treated as a black box. In this respect, the straightforward rule extraction from a *neulonet* would attain maximal *fidelity*. Moreover, since the rule extraction process is *lossless*, any concerns with *accuracy* can be referred to the network structure. As for the criterion of *comprehensibility*, there is a general suggestion that the richer human-like logic will result in the extraction of a more compact rule. The size of a rule extracted from a *neulonet* is a measure of the number of constituent *net rules*, as well as the number of antecedent conditions in each *net rule*. Our motivation in utilizing *net rules* that depict human logic decision making is due to the ability to exploit the richer thought processes that are extensively used in human reasoning. This would allow for an improved expression of the underlying logic, as well as the discovery of hidden nuances in the problem domain. Unfortunately, the apparent high degree of interdependence among the *net rules* as reflected earlier is an undesirable intrinsic feature of *neulonet* rule extraction. The set of constituent *net rules*, as depicted in table 2.2, has to be interpreted in its entirety as a connecting set in order to make sense of the underlying logic. Clearly, the logic descriptions will get increasingly complex as the *neulonets* get progressively larger, underlining an adverse effect on comprehensibility. This issue is addressed further in chapter 4.

2.4 Summary

We have introduced the basic schema of a neural logic network and illustrated examples of rudimentary *net rules* that simulates heuristics and strategies in human decision making. In the next chapter, we shall show how neural logic networks are evolved under a basic framework of genetic programming. Although learning in neural logic networks can be performed in a variety of ways, an evolutionary platform that leaves the weights intact is necessary to preserve the semantics of the rules for straightforward rule extraction. As we shall see, bounded rationality is inherent within a evolutionary learning paradigm as we seek to evolve *neulonets* by considering not only their accuracy, but also their size. *Neulonet* generation is also implemented by employing an early stopping technique, which transpires as a bounded constraint with respect to time. The framework is also important as it forms the foundation for extended evolutionary paradigms in association-based evolution and niched-based evolution.

Chapter 3

Neural Logic Network Evolution

In the preceding chapter, we looked at how decision strategies adopted in human decision making can be realized through the use of rudimentary decision heuristics and operations (i.e. neural logic networks). In this chapter, we focus on the learning algorithm *per se*. Different approaches have been devised for *neulonet* learning; these include the resolution of a set of inequalities [Teh, 1995], learning via neural network back-propagation training [Teh, 1995, Tan et al., 1996], and *neulonet* evolution via genetic programming (GP) [Tan and Chia, 2001a, Tan and Chia, 2001b]. In particular, GP *neulonet* learning facilitates the generation of a more effective pattern classifier as compared to the genetic evolution of logic based neural networks representing the standard boolean logic operations [Gaudet, 1996]. Another motivating factor for advocating the evolutionary approach as compared to back-propagation in *neulonet* learning is the fact that the weights of the network are kept intact throughout the genetic evolution process, thereby facilitating the straightforward extraction of logic rules from the evolved *neulonet* solution. Detailed study of the process of adaptation in genetically programmed neural logic networks is important as it culminates in the realization of a "cognitive-inspired" knowledge discovering system.

3.1 Learning via genetic programming

In chapter 2, we have seen how a composition of *net rules* can be used to realize more complicated decisions as shown in the example of Space shuttle landing in section 2.1. This can be achieved using genetic programming [Koza, 1992], which is an extension of the conventional genetic algorithm where instead of subjecting bit patterns to genetic evolution, the individuals in the gene population are computer programs. In the context of *neulonet* evolution, these computer programs are represented in the form of *neulonets*. A brief description of the genetic *neulonet* construction process (detailed in [Tan and Chia, 2001a]) is presented below. Note the following describes genetic programming performed on the system of discrete neural logic network with the three values, i.e. "true" (1,0), "false" (0,1) and "unknown" (0,0). This will form the basis from which evolution of probabilistic neural logic networks can be extended. For a comprehensive survey on evolutionary algorithms for data mining and knowledge discovery, one can refer to [Freitas, 2002]. Another recommended survey on evolutionary neural networks can be found in [Yao, 1999].

3.1.1 *Neulonet* structure undergoing adaptation

Adaptation is performed on the population of *neulonets*, each being a recursive composition of *net rules* and input terminals in a tree-like structure. The input terminals are the attributes of the data set, as well as bias decision nodes depicting default "true", "false" and "unknown" decisions. Each individual of the initial population of *neulonets* is generated in a similar fashion as Koza's "ramped-half-and-half" generative method [Koza, 1992] which is a mixed method incorporating both a "full" method where individuals are grown to a maximum specified initial

depth, and a "grow" method where individuals are grown with variable depth. The method of *neulonet* construction in the initial population accounts for both depth as well as fan-out of the *neulonet*.

3.1.2 Genetic operations

We describe three fitness-proportionate genetic operations for modifying the *neulonet* structure undergoing evolution. The reproduction operation first selects a single *neulonet* from the population, and the selected individual is then copied to the new population. The crossover operation involves swapping the chosen parts of two *neulonets* with constraints imposed on the swapping process to preserve the syntactic integrity. For instance, swapping should not leave any dangling intermediate output nodes as such nodes will not be able to receive input values during firing. Figure 3.1 illustrates a crossover operation on two *neulonets*. The mutation operation involves changes to a chosen *neulonet*. A random mutation point is picked such that the *neulonet* whose root is the mutation point is replaced by another randomly generated *neulonet*. Figure 3.2 shows the effect of the mutation operation on a *neulonet* with the "Conformity" *net rule* replaced by an "Overriding" *net rule*.

It is interesting to note that the "Conformity" *net rule* appears redundant in the activation of a *neulonet* and might be deemed not to be doing anything significant in the evolutionary process. This kind of *net rule* is similar to an *intron* in biology, which is a chromosome that is never expressed and provides spacing between the genes [Angeline, 1994]. However, Levenick notes that *introns* are useful in genetic algorithms [Levenick, 1991].

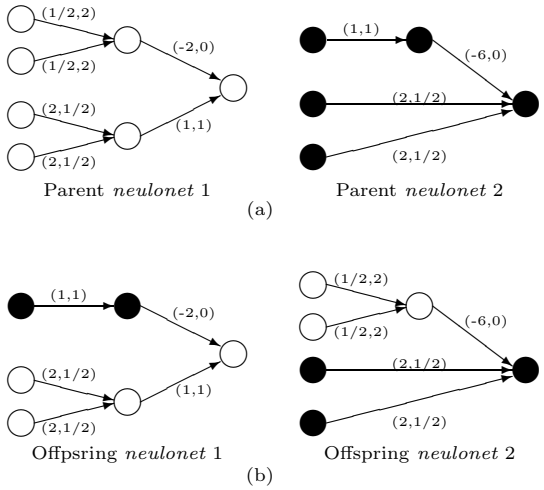

Figure 3.1: Two *neulonets* before and after the crossover operation.

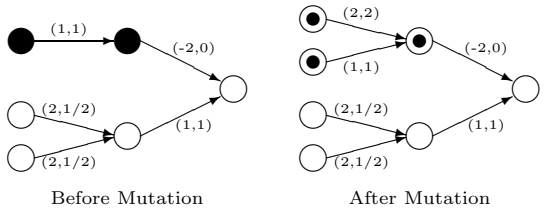

Figure 3.2: *Neulonet* with a mutated *net rule*.

3.1.3 Fitness measure and termination criterion

Each *neulonet* in the population is assigned a normalized fitness measure $f(\epsilon, \sigma; \kappa)$ based on the errors produced by the *neulonet*, ϵ, as well as the size of the *neulonet*, σ as defined in equation (3.1). The fitter the *neulonet*, the larger will be its fitness measure. The factor, $\kappa \in [0, 1]$, is used to weigh the effects of accuracy over size in the fitness measure. A higher value for κ places more emphasis on finding an accurate solution at the expense of the size of the *neulonet*. Moreover, the empirical fine-tuning of an appropriate κ value allows the extraction of a simpler set of *net rules* that avoids unnecessary complications; however, this might be at the expense of the generalization ability. Note that σ_{min} in equation (3.1), denotes the smallest possible size of a *neulonet*.

$$f(\epsilon, \sigma; \kappa) = \frac{1}{1 + \kappa\epsilon + (1 - \kappa)(\sigma - \sigma_{min})} \tag{3.1}$$

Termination of evolution is controlled using a parameter that specifies a period in which the termination criterion is examined upon its elapse. The test for termination involves recording the current generation's fittest *neulonet*, and comparing it against the preceding record. Termination transpires when there is no improvement in the classification accuracy and size of the current recorded *neulonet*, in which case, it is designated as the solution to the problem domain. The algorithm for evolving a *neulonet* using genetic programming is outlined in table 3.1. Additionally, to further simplify the evolved *neulonet*, a set of semantic-based simplification rules are applied recursively to identify component *net rules* for elimination or recombination. For example, an evolved *neulonet* consisting of only multiple *Overriding* rules would eventually be simplified to constitute only one *Priority* rule of a larger fan-out.

Table 3.1: An algorithm for evolving *neulonets* using genetic programming.

1. Generate an initial population of *neulonets*.
2. Iteratively perform the following sub-steps until the termination criterion has been satisfied:
   - 2.1. Fire each *neulonet* in the population and assign it a fitness value using a fitness measure.
   - 2.2. Create a new population of *neulonets* by applying the operations of reproduction, crossover or mutation on *neulonets* chosen with a probability based on fitness.
3. The *neulonet* that is identified to be the best individual is designated as the solution to the problem.

The astute reader will realize that thus far, we have focused only on binary classification problems. However, there are many n-class problems where a large number of representative samples in the data set are available for each of the n classes. One approach for handling multi-category classification problems in genetic programming is to re-formulate an n class problem as n binary problems [Kishore et al., 2000]. As such, n separate *neulonets* for resolving each separate class will be evolved.

3.2 Effectiveness of *neulonet* evolution

The first experimental study on genetically programmed neural logic network learning was conducted to evaluate the effectiveness of using an extended set of *net rules* against the mere use of category IV *net rules* to simulate standard boolean

logic of negation, conjunction and disjunction. Data sets from the publicly available UC Irvine data repository [Blake and Merz, 1998] was used for the study. Due to the discrete nature of three-valued *neulonet* inputs comprising only values of "true" (1,0), "false" (0,1) and "unknown" (0,0), data sets containing discrete attribute data types had to be first transformed to an equivalent binary data set with one attribute for each attribute-value pair in the original set. For the case of data sets having three or more class values, a separate data set that performs a boolean classification for each class value was created. Moreover, data sets containing continuous-valued attributes had to undergo discretization prior to transformation. An entropy-based discretization was chosen amongst many other discretization techniques (see [Liu et al., 2002] for a comprehensive survey). The results are tabulated in Table 3.2 with details of the experiment described in [Tan and Chia, 2001b].

In terms of classification accuracy, *neulonet* evolution provided a comparable, if not better, result than standard logic evolution in all cases. This was especially apparent for data sets having an ordered logic reasoning as in the case of the Space Shuttle Landing Domain problem, or when the classification rules encompassed a "quantification" of standard boolean logic. For example, in the Monk-2 data set, the given classification rule is as follows:

$$\text{Exactly two of } a_i = 1 \;\; \forall i \in \{1, 2, 3, 4, 5, 6\}$$

Clearly, it would be difficult to achieve high classification accuracy using only a composition of standard boolean logic units. However, in the case of *neulonet* evolution, the expressive power inherent in the set of *net rules* allowed for a more

Table 3.2: Experimental results depicting classification accuracy for the best individual. The numbers below the accuracy value denotes the number of decision nodes and (number of generations).

Data Set	*Neulonet*	Standard
Shuttle-landing	100% 3.0 (34.7)	100% 5.5 (48.3)
Iris-setosa	100% 1.0 (18.0)	100% 1.0 (12.0)
Iris-versicolour	99.8% 5.6 (230.0)	99.3% 8.0 (286.7)
Iris-virginica	99.6% 5.3 (154.0)	98.8% 3.0 (133.3)
Monk-1	100% 5.6 (31.6)	100% 4.7 (24.0)
Monk-2	99.8% 19.2 (456.4)	93.2% 20.8 (544.5)
Monk-3	100% 3.0 (26.6)	99.1% 4.0 (30.3)
Voting-records	99.6% 14.3 (356.3)	97.7% 13.0 (538.7)
Breast-cancer	99.5% 20.0 (552.0)	99.4% 20.2 (666.0)
Mushroom	100% 6.5 (46.2)	100% 6.25 (71.3)

accurate solution tree to be evolved. From the experimental analysis, it was observed that*net rule* evolution gives a final solution that is more accurate in significantly fewer generations, as shown in the accuracy profile for both approaches in figure 3.3.

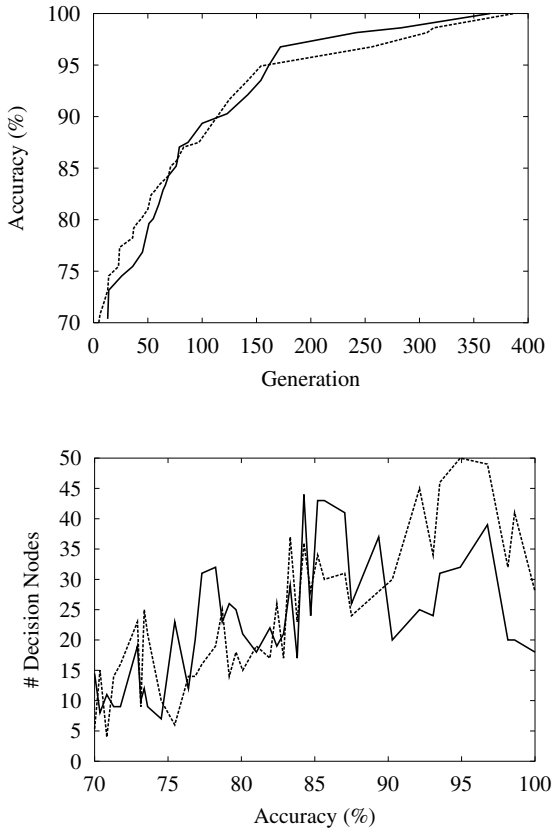

Figure 3.3: Accuracy and size profiles for *net rule* (solid-line) versus standard boolean logic (dotted-line) evolution in the Monks-2 data set.

The expressive power inherent in the entire set of *net rules* allowed for a more accurate solution to be evolved. Another aspect worth noting is that due to the *neulonet's* ability to handle more complex decisions, *net rule* evolution provided

more compact solutions than their standard boolean counterparts for the same classification accuracy, particularly for data sets having non-trivial classification rules. Using the best evolution runs for both empirical approaches in the Monk-2 data set, the size profiles for the number of decision nodes is shown in figure 3.3 for accuracies between 70% to 100%. Observe that the profiles for both approaches are comparable in the case of classification accuracies of less than 90%, indicating that the classification rule learned thus far was still relatively simple. However, as the required accuracy increased, the apparent power of *neulonets* in deriving complex decision rules resulted in a significantly smaller solution size.

A general drawback in *net rule* evolution was the longer time needed to converge to an optimal solution due to the larger size of the component *net rule* library. It has to be noted that the experiment conducted actually gave the standard logic an advantage in that the rule set is small, while the population sizes in both *neulonet* and standard logic evolution approaches were kept the same at 10,000 individuals. As a result, there were more opportunities for the small set of standard logic *rules* to quickly evolve to ideal individuals within the population. Thus for problems involving simpler decisions, standard logic evolution attained the required accuracy faster. However, there were cases in which *neulonet* evolution was faster. This was observed from the accuracy profiles for the Monk-2 data set in figure 3.3. For accuracies of less than 95%, standard logic evolution required slightly less generations. However, as the demand in accuracy increased, it became increasingly difficult to evolve a solution using only standard logic rules. The added advantage in expressing complex rules during *neulonet* evolution, on the other hand, produced a faster rate of convergence.

In the construction of the rudimentary *net rules*, representing decision heuristics and strategies that depict human-like concepts is seen as a positive step towards the building of an intelligible knowledge discovery system. Using the genetic programming platform for *neulonets* learning as a starting point, we investigate how this learning can be suitably extended to probabilistic neural logic networks to account for uncertain information and decision processes.

3.3 An illustrative example

An example is provided here to illustrate an intrinsic feature of *neulonet* rule extraction, i.e. the high degree of dependence among rules, which is evident when the data set is relatively more complex. We use the US Congressional Voting Records Database [Schlimmer, 1987] to illustrate rule-based learning by composing *net rules*. This data set consists of 435 instances and 16 boolean attributes with unknown or missing values. Each instance is classified as either *republican* or *democrat*. GP *neulonet* learning generated a solution with a manageable number of rules while being sufficiently accurate. A desirable result was achieved by setting κ to 0.94. The evolved *neulonet* solution is shown in figure 3.4 with a classification accuracy of 97.7%. The set of logic rules extracted for this *neulonet* is given in table 3.3. Note that a "true" outcome denotes the *republican* class.

Observe that the bias-based "Veto" *net rule* **Q4** has rule **Q2** as the veto factor, which implies that **Q2** plays a more important role than **Q3**. This veto power will come into effect only when **Q2** returns a positive decision. Application of backward induction indicates that such a situation occurs only if the majority of the contributing factors of rule **Q2** are true. One instance occurs when *synfuels-corporation-cutback(s-c-c)* and *mx-missile(m-m)* are true. The contributing factors

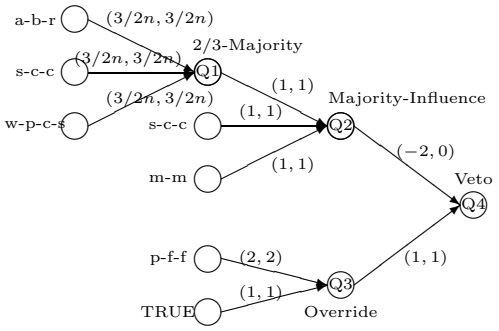

Figure 3.4: Evolved *Neulonet* solution for voting records.

Q1 ⇐ 2-out-of-3-Majority[a-b-r=y, s-c-c=y, w-p-c-s=y]

Q2 ⇐ MajorityInfluence[**Q1**, s-c-c=y, m-m=y]

Q3 ⇐ Overriding[p-f-f=y, TRUE]

Q4 ⇐ Veto[**Q2**,**Q3**]

Table 3.3: Extracted *net rules* from the voting records data.

comprising rules **Q1** and **Q2** can be viewed as belonging to two different classes of decision makers. The fact that rule **Q1** forms part of rule **Q2** leads us to deduce the relative importance between the classes of contributing factors in both the rules. Moreover, the presence of the attribute *s-c-c* in both rules signifies its higher level of participation in the overall decision. *Net rules* **Q1** and **Q2** provide a concise way of expressing the appropriate rule activations, rather than enumerating every possible combination of decisions factors for each rule activation using separate rules as in the case of general production rules. Finally, if the outcome of **Q2** is false or unknown, then the higher priority *physician-fee-freeze(p-f-f)* attribute in rule **Q3** will influence the final outcome, provided that this attribute value is deterministic, i.e. "true" or "false". Otherwise, rule **Q4** will assume a true outcome. From the above analysis, it can be seen that the set of extracted *net rules* does provide an alternative perspective in rule-based learning that is more expressive. GP *neulonet* learning could be used to exploit the "richer" thought processes that are extensively used for human reasoning, The library of *net rules* is therefore a positive contribution towards the induction of human logic rules that are more expressive in nature while remaining concise. This is especially useful particularly in real-world problem domains.

As evident in the extracted rules in table 3.3, there is a high degree of interdependence among the rules. The set of rules has to be interpreted in its entirety as a connecting set in order to discover the underlying logic of the problem domain. From the layman interpretation of the rules given above, it is apparent that the logic descriptions will get increasingly complex as the *neulonet* solutions get progressively larger. This complexity is especially apparent when the *neulonet* solution contains deeply-nested *net rules*. This interdependence among the set of

logic rules captures the logic relationship between rules. However, empirical observation shows that this same property becomes more of a liability when the *net rules* are nested at more than three levels deep. An ideal classification system should, in essence, be able to exploit this interdependence property albeit to a lesser degree.

3.4 Summary

This chapter focuses on the aspect of adaptability, and the use of an evolutionary computation paradigm of learning is suggestive of such a notion. In terms of ecological rationality or simply its fit to the environment, the "environment" in consideration is primarily the instances of the problem space or data set, and genetic programming with an accuracy-based fitness measure evolves individuals that fit to this environment which is based primarily on accuracy, while still maintaining the elements of simplicity, comprehensibility and generalizability.

Our preliminary investigation into the human amenability of evolutionary neural logic networks have seen both strengths as well as weaknesses. Employing human-like concepts in the construction of rudimentary *net rules* is seen as a positive step towards the building of an intelligible knowledge discovery system. In particular, neural logic network learning via genetic programming has lately been deployed in medical diagnosis [Tsakonas et al., 2004] and bankruptcy prediction [Tsakonas et al., 2006] However, some inadequacies have also been addressed recently [Chia and Tan, 2004a, Chia and Tan, 2004b]. These include the evolution of multiple compact *neulonets* as well as improving the fitness function in genetic programming to incorporate rule evaluation measures.

As we describe in the next chapter, all these were accounted for by extending the current implementation to the association-based classification platform. Experimental studies on benchmark data sets have shown the effectiveness of the new system in terms of better generalization accuracy as well as smaller and more interpretable rule sets, as compared to general association-based classifiers using the implicit conjunct.

Chapter 4

Association-Based Evolution

In its most rudimentary form, genetic algorithm/genetic programming for supervised learning requires that all individuals of the evolving population be evaluated against the entire problem space of known instances. Selection pressure drives the population towards the most highly fit individual. While mutation might starve off perfect convergence to a single individual, it is unarguable that the effect of fitness proportionate selection is the loss of low-quality diversity. In many applications of genetic evolution, this uniform convergence is undesirable. For example, multiple objective problems might entail the discovery of multiple solutions with different tradeoffs among the multiple objectives. Each objective in a multiple objective problem can be viewed as analogous to a niche in the environment of niches.

To prevent the best individual in the population from replacing all copies of competing rivals (i.e. the best solutions of every niche), some kind of niche specialization (or simply niching) is necessary. Niching aims to induce restorative pressure to balance the convergence pressure of selection. Some would argue that we can always decrease selection pressure by some form of premature or early stopping, so that the population does not coverge in the specified time frame of interest, but

such a naive strategy leaves much to be desired. There are many recommendations for having distributed populations for the discovery and maintenance of diverse schemas, with each schema covering a subset of the problem space (a *neulonet* is an example of a schema). However, distributed populations only prolong the inevitable collapse of the population distribution by a number of generations that grows linearly with population size and number of subpopulations. Therefore, in our context of classification, we seek an evolutionary paradigm that ensures the maintenance of a diverse group of high-quality individuals (or rules), such that together they can classify the examples at hand.

The ensuing research was motivated in part by a novel data mining technique of employing association rule mining concepts [Agrawal et al., 1993] for classification, or simply *association-based classification*, which is an alternative to purely accuracy-based classifier systems (see [Freitas, 2000] for the differences between association rule mining and classification rule mining). Among the many associative classifiers in the literature, the CBA system [Liu et al., 1998] is of particular interest. CBA comprises two distinct stages: a rule-generation stage followed by a classifier-building stage. In the rule-generation stage, all independent *class association rules* (CARs) are generated. The classifier-building stage, on the other hand, uses a variant of sequential covering to group these CARs together to form an eventual classifier. Clearly, this two stage process is easily adaptable for the proposed *neulonet* associative classification technique. Rather than generating the CARs that only involve an implicit conjunction operator, the novelty lies in evolving compact *neulonet association rules* (NARs) in their place. These NARs would then be utilized for classifier building without any major modifications to the existing classifier-building algorithm in CBA. On the microscopic level, an NAR would be able to exploit the interdependence property in *neulonet* rule-inference, while

on the macroscopic level, the relevant NARs could collectively form an accurate classifier.

4.1 Notions of confidence and support

Association rule mining is a common technique used for market basket analysis [Agrawal et al., 1993]. Assigned to every association rule is the two important notions of *confidence* and *support*. Basically, the *confidence* of a rule provides a measure on the accuracy of the rule. On the other hand, the *support* of a rule is an indication of the amount of data that is consistent with the rule. In order to incorporate genetically-programmed *neulonet* learning into the associative classification domain, these two notions have to be accounted for in the fitness measure during *neulonet* evolution.

In CBA, a class association rule is expressed in the form

$$a_1, a_2, \ldots, a_k \rightarrow y$$

where each of the terms a_i denotes a boolean attribute, and y denotes the class label. For ease of explanation, denote the LHS of the above expression as the *condset* (the set of conditional attributes). For each CAR, the condition support count, c, is the number of cases in the data set $\mathbf{D}$ that satisfy the *condset*. On the other hand, the rule support count, r, is the number of cases in $\mathbf{D}$ that satisfy the *condset*, with the additional requirement that each of these cases should be labeled with class y. The support, S, and confidence, C, of a CAR is defined as follows:

$$S = \frac{r}{|\mathbf{D}|} \tag{4.1}$$

$$C = \frac{r}{c} \tag{4.2}$$

Implicit in the *condset* of the CAR is the conjunction operator that operates on the conditional attributes. As such, the CAR can be re-expressed as

$$a_1 \wedge a_2 \wedge \cdots \wedge a_k \rightarrow y$$

Without loss of generality, we express a *neulonet* into an alternative syntax. For example, if a *neulonet* comprises only of the *Conjunction net rule*, then

$$y : \mathbf{Q1} \leftarrow \texttt{Conjunction}(a_1, a_2, \ldots, a_k)$$

The label y denotes that the above *neulonet* (or the extracted rule) is associated with the class y, so that an outcome of "true" in **Q1** will signify the class y. We redefine c as the number of examples in the training data for which the *neulonet* solution returns a "true" outcome, and r as the number of examples for which the *neulonet* solution returns a "true" outcome, with the additional requirement that each of these cases must be labeled with the class y.

The usual fitness measure for *neulonet* evolution given in equation (3.1) is based upon the error rate ϵ and size σ of the evolved *neulonet*. In the context of associative classification, ϵ needs to be replaced by a penalty function $\Psi(S, C)$ or $\Psi(r, c, |\mathbf{D}|)$, that accounts for the support and confidence as shown in equation (4.3).

$$f[\Psi(\cdot), \sigma; \kappa] = \frac{1}{1 + \kappa\Psi(\cdot) + (1 - \kappa)(\sigma - \sigma_{min})} \tag{4.3}$$

In accordance with CBA's precedence ordering of rules, it is crucial that the fitness measure be able to express an implicit ordering where precedence is based on the confidence followed by support. A preliminary measure was devised in [Chia and Tan, 2004a] and subsequently revised in [Chia et al., 2006]. Specifically, we defined a precedence function, $p(c, r, |\mathbf{D}|)$, as follows.

$$p(c, r, |\mathbf{D}|) = \frac{r}{c + \delta} \tag{4.4}$$

We desire to find a suitable upper bound for δ that would ensure conformance to the precedence ordering. Let c_i and r_i be the respective condition and rule support counts of rule i. Suppose rule 1 has a higher confidence than rule 2.

$$\frac{r_1}{c_1} > \frac{r_2}{c_2} \Rightarrow r_1 c_2 - r_2 c_1 > 0 \tag{4.5}$$

We want the precedence value $p_1 > p_2$, so

$$\frac{r_1}{c_1 + \delta} > \frac{r_2}{c_2 + \delta} \Rightarrow \frac{r_1 c_2 - r_2 c_1}{r_2 - r_1} > \delta \tag{4.6}$$

Since $\delta > 0$, combining the constraints in both 4.5 and 4.6 gives $r_2 > r_1$ and $c_2 > c_1$. To find a lower bound for δ, we need to find

$$\delta < \min\left(\frac{r_1 c_2 - r_2 c_1}{r_2 - r_1}\right) \tag{4.7}$$

subject to the following constraints:

$$0 < r_1 \leq c_1 < c_2; r_1 < r_2 < c_2; c_2 \leq |\mathbf{D}|$$

To maximize the denominator, we set the values of r_1 and r_2 to 1 and $|\mathbf{D}| - 1$ respectively. Consequently, substituting the valuyes of c_1 with 1 and c_2 with $|\mathbf{D}|$ in 4.7, we get

$$\delta < \frac{|\mathbf{D}| - (|\mathbf{D}| - 1)}{(|\mathbf{D}| - 1) - 1} = \frac{1}{|\mathbf{D}| - 2} \tag{4.8}$$

From the above results, we adopt the precedence function $r/(c + \delta)$ by setting δ to $1/|\mathbf{D}|$. We can then inject the precedence function directly into the penalty function.

$$\Psi(r, c, |\mathbf{D}|) = \left(1 - \frac{r}{c + 1/|\mathbf{D}|}\right)|\mathbf{D}| \tag{4.9}$$

This ensures that the effect of the weighting factor κ is consistent across both the fitness measures with the penalty function now defined such that the range of values of $\Psi(\cdot)$ in equation (4.3) is consistent with that of $\epsilon \in [0, |\mathbf{D}|]$ in equation (3.1) with lower values corresponding to better individuals. Our modified fitness function implicitly follows CBA's approach of selecting class association rules with higher *confidence* followed by higher *support*. Moreover, CBA ensures that CARs

generated have *support* and *confidence* levels that are above their specified minimum levels, respectively. This same strategy is adopted during *neulonet* evolution where *neulonets* with low support and/or confidence are assigned zero fitness.

4.2 Rule generation and classifier building

With the fitness measure reformulated, all that is needed now is to build the classifier. *Neulonet* associative classification involves two phases. The motivation for the first stage of *neulonet* association rule (NAR) generation (table 4.1) is simply to find a minimal ordered sequence of NARs that would cover the required hypothesis space. During the second stage of classifier building (table 4.2), the best NARs from all sequences will then compete to be included into the eventual classifier. Both algorithms are variants of Michalski's AQ algorithm [Michalski et al., 1986] for sequential covering.

Rule generation is applied n times (possibly in parallel) on a n-class problem, each generating a sequence of NARs that resolves a distinct class label. The resolution of each class i entails the repeated evolution of optimal NARs and incremental pruning of the data set to achieve a maximal coverage of the data with a minimal sequence of NARs evolved. A final round of data set coverage is performed during the classifier building phase to amalgamate the best of all NAR sequences produced during rule generation. However, as the addition of more NARs into a final list of rules might cause the misclassification rate to increase, every inclusion of a new NAR would require a default label (which is the majority class of the remaining uncovered data instances) to be considered and the number of misclassifications to be recorded. From this eventual list of NARs, we find the first NAR with the lowest misclassification rate and prune away all succeeding NARs.

Table 4.1: Algorithm to generate *neulonets* for class i.

Algorithm: **GenerateNARs(i)**
1. Initialize queue $\mathbf{C}_i$ as empty.
2. Iteratively perform the following until data set **D** no longer contains any instances with class value i, or no other NAR can be evolved.
2.1. Evolve a NAR that resolves class i.
2.2. For each instance **d** in **D**, mark **d** if
(a) Firing NAR with **d** returns "true"; and
(b) The class label of **d** is i.
2.3. Insert NAR at end of queue $\mathbf{C}_i$.
2.4. Remove all marked instances in **D**.
3. Return queue $\mathbf{C}_i$.

We use the Voting Records Database to analyse the NARs generated. To maintain the comprehensibility of each *neulonet association rule*, the evolved NARs are kept small at a depth of at most two, with each *net rule* having a fan-out not exceeding three. The minimum support and confidence levels were set at 0.25 and 0.5 respectively. *Neulonet* associative classification produced a classifier with 18 NARs as compared to CBA's classifier of 29 CARs. Table 4.3 shows an extract of the first three NARs which would have achieved an accuracy of 94.9%. In fact, the first rule **Q1** by itself would already have achieved an accuracy of 94.0% with *republican* as the default class; the layman interpretation being: if the *physician-fee-freeze (p-f-f)* attribute is true, then the decision would be *republican*, since **Q1** is not satisfied. Otherwise, the decision is *democrat* when either *absorption-of-budget-resolution (a-b-r)* or *synfuels-corporation-cutback (s-c-c)* are true). In

Table 4.2: Algorithm for classifier building.

Algorithm: **BuildClassifier**
1. Initialize classifier **C** as empty.
2. Iteratively perform the following until all $\mathbf{C}_i$ queues are empty
2.1. Compare all NARs at head of each $\mathbf{C}_i$ and remove NAR with highest fitness.
2.2. For each instance **d** in data set **D**, mark **d** if
(a) Firing NAR with **d** returns "true"; and
(b) The class label of **d** is the same as the class resolved by the chosen NAR.
2.3. Perform the following if some **d** is marked.
(i) Insert NAR at end of **C**.
(ii) Remove all marked instances in **D**.
(iii) Compute default class of **D** (majority class of remaining instances).
(iv) Compute number of errors of **C**.
3. Find the first NAR in **C** with the lowest error and drop all NARs after that.
4. Add default class of last NAR to the end of **C**.
5. Return classifier **C**.

contrast, the first five CARs generated in CBA achieved a relatively lower accuracy of 93.8%. Rule-for-rule comparisons show that *neulonet* associative classification does indeed perform better than CBA as expected. This is again attributed to the more expressive human logic *net rules* used in classification. Furthermore, as the interdependence property is kept to a minimum, each NAR remains easily comprehensible in layman terms.

Resolving democrat :
Q1 ⇐ Veto-Disjunction[p-f-f=y, a-b-r=y, s-c-c=y]
$Conf$: 1.00; Sup : 0.554; DefaultClass:rep; Acc : 0.940
Resolving republican :
Q3 ⇐ 2-out-of-3-Majority[**Q2**, p-f-f=y, s-c-c=n]
Q2 ⇐ Silence-Means-Consent[i=y, a-b-r=y, d-f-e=n]
$Conf$: 1.00; Sup : 0.303; DefaultClass:rep; Acc : 0.940
Resolving democrat :
Q4 ⇐ Veto[a-s-t-b=n, c=n]
$Conf$: 1.00; Sup : 0.345; DefaultClass:rep; Acc : 0.949

Table 4.3: First three NARs generated for the Voting Records data.

4.3 Empirical study and discussion

An empirical study is conducted with the aim of comparing the generalization capability, in terms of predictive accuracy and size of classifiers, constructed from CBA with that of *neulonet* association-based classification using the entire library of *net rules* (NACfull) and also the restricted set of Category IV *net rules* to simulate the use of standard boolean logic (NACstd). Experiment result based on a ten-fold cross validation on 24 data sets from UCI are reproduced in Table 4.4. Specific details of experiments can be found in [Chia et al., 2006]. Briefly, *neulonet* associative classification using the entire library of *net rules* is generally the better choice in terms of generating considerably smaller classifiers with higher classification accuracies. Whether using the entire set of *net rules* or the restricted set of standard boolean logic rules, *neulonet* associative classification outperformed CBA for the majority of the data sets. Note that exceptions include the *australian* and *waveform* data sets where the number of CARs being generated is noticeably more

Table 4.4: Experimental results depicting predictive errors (percent) of different classifiers. Values in bold indicate the lowest error. For CBA/NACfull/NACstd, values within brackets denote the average number of CARs/NARs in the classifier. NAC results are also depicted with their mean and variance.

Data Set	C4.5rules	RIPPER	NBC	CBA		msCBA	NACfull		NACstd	
annealing	6.5	4.6	2.7	3.6	(34)	2.1	**0.3 ± 0.02**	(18.3 ± 0.3)	0.8 ± 0.08	(15.4 ± 0.2)
australian	13.5	15.2	14.0	**13.4**	(148)	14.6	13.9 ± 2.5	(80.6 ± 6.1)	15.7 ± 3.3	(77.6 ± 8.6)
auto	29.2	23.8	32.1	27.2	(54)	19.9	**17.5 ± 4.3**	(34.6 ± 2.2)	19.8 ± 3.5	(31.4 ± 2.2)
breast-w	3.9	4.0	**2.4**	4.2	(49)	3.7	3.2 ± 0.2	(41.4 ± 0.4)	3.9 ± 0.3	(38.7 ± 1.4)
cleve	18.2	21.1	17.1	16.7	(78)	17.1	**15.6 ± 1.0**	(41.3 ± 2.0)	18.1 ± 1.2	(39.5 ± 1.2)
crx	15.9	14.6	14.6	14.1	(142)	14.6	**13.7 ± 0.7**	(84.8 ± 6.5)	15.6 ± 0.7	(69.6 ± 7.5)
diabetes	27.6	25.3	24.4	25.3	(57)	25.5	24.5 ± 1.3	(20.8 ± 3.0)	**24.0 ± 2.3**	(18.2 ± 1.5)
german	29.5	27.8	**24.6**	26.5	(172)	26.5	25.4 ± 1.4	(64.1 ± 4.1)	25.1 ± 1.8	(64.1 ± 8.5)
glass	27.5	35.0	29.4	27.4	(27)	26.1	**22.7 ± 4.8**	(23.6 ± 1.1)	24.6 ± 4.3	(21.1 ± 1.5)
heart	18.9	19.6	18.1	18.5	(52)	18.1	14.7 ± 2.4	(38.2 ± 2.6)	**14.6 ± 1.4**	(23.9 ± 2.2)
hepatitis	22.6	17.5	15.0	15.1	(23)	18.9	**14.1 ± 2.2**	(15.2 ± 1.2)	14.5 ± 3.2	(15.4 ± 1.3)
horse	16.3	14.7	20.6	18.7	(97)	17.6	14.7 ± 2.4	(38.1 ± 2.8)	**14.6 ± 1.4**	(24.0 ± 2.3)
hypo	1.2	**0.8**	1.5	1.7	(35)	1.0	1.2 ± 0.1	(44.2 ± 2.5)	0.9 ± 0.1	(47.0 ± 2.8)
iono	8.0	11.4	12.0	8.2	(45)	7.7	7.3 ± 2.0	(21.8 ± 1.5)	**5.4 ± 1.9**	(22.8 ± 2.7)
iris	**5.3**	**5.3**	6.0	7.1	(5)	**5.3**	5.7 ± 2.2	(6.2 ± 1.3)	5.7 ± 2.2	(6.1 ± 1.3)
labor	21.0	16.5	14.0	17.0	(12)	13.7	**6.6 ± 2.4**	(3.2 ± 1.1)	6.8 ± 2.2	(8.4 ± 2.5)
lymph	21.0	20.8	24.4	19.6	(36)	22.1	**12.0 ± 3.0**	(21.3 ± 1.2)	12.4 ± 4.3	(23.8 ± 2.2)
pima	27.5	26.3	**24.5**	27.6	(45)	27.1	24.7 ± 3.1	(19.8 ± 2.0)	24.8 ± 2.6	(17.4 ± 1.6)
sick	2.1	**1.9**	3.9	2.7	(46)	2.8	2.7 ± 0.1	(49.8 ± 3.5)	2.8 ± 0.1	(35.7 ± 4.4)
sonar	27.8	27.9	23.0	21.7	(37)	22.5	**14.7 ± 4.4**	(21.6 ± 1.5)	19.2 ± 3.1	(23.8 ± 2.0)
vehicle	33.6	31.4	40.1	31.1	(125)	31.0	**29.6 ± 1.2**	(35.8 ± 2.5)	30.8 ± 1.4	(32.5 ± 2.6)
waveform	24.6	20.5	**19.3**	20.6	(386)	20.3	26.7 ± 0.3	(62.0 ± 10.4)	26.5 ± 0.6	(41.7 ± 6.0)
wine	7.9	8.5	9.5	8.4	(10)	5.0	0.9 ± 0.26	(5.2 ± 0.3)	**0.7 ± 0.11**	(4.9 ± 0.2)
zoo	7.8	11.0	13.7	5.4	(7)	3.2	**2.3 ± 0.7**	(8.0 ± 1.1)	2.6 ± 0.6	(8.6 ± 1.0)

than the NARs generated in the NAC classifiers. In the case of NACfull, the significantly smaller classifier size does not adversely affect the classification accuracy as the corresponding increase in predictive error is minimal. In essence, it is more desirable to have a much smaller classifier if the drop in accuracy is inconsequential. As is the case of the first empirical study, the large variety of *net rules* used to represent the common human decision processes is once again the paramount factor in producing better classifiers due to the "richness" in logic expression.

4.4 Summary

We have discussed an attempt at neural logic network learning based on the genetic programming paradigm with an association-based classification approach. In terms of ecological rationality or its fit to the environment, rather than generating a single *neulonet* to resolve the entire problem space, the use of notions of support and confidence within an association-based implementation allows us to evolve *neulonets* within their specialized niches. The association-based approach seems to provisionally suggest that we have moved from representing the whole environment as one niche to several niche environments. However, it is worthwhile to note that the rule generation phase underlies a sequential covering approach. Although each *neulonet* remains interpretable, the "macroscopic" comprehensibility and amenability of the entire list of *neulonets* is somewhat questionable. [Minsky, 1991] noted that the approach of incrementally adding rules would usually work well at first, but whenever we try to move beyond the realm of toy problems and start accumulating more rules, we usually get into trouble because each additional rule is increasingly likely to interact in unexpected ways with the others. So in a way, sequentially added rules have also undesirable effects in terms of utility. Incrementally evolving *neulonets* (or adding rules) merely changes the environment progressively such that instances that are resolved early are purged while those that are yet unlearnt remain. In a sense, *neulonets* are generated sequentially such that the latter ones would address the increasingly subtle intricacies and nuances of the problem domain.

To seek a proper notion of a niche, we turn to evolutionary social psychology (and even evolutionary personality psychology). From an evolutionary perspective, competition is keenest among those pursing the same strategy. As one niche becomes more and more crowded with competitors, success of those in the niche can suffer

compared with those seeking alternative niches. Selection favors mechanisms that cause some individuals to seek niches in which the competition is less intense, and hence in which the average payoff maybe higher. It is apparent that niche specialization occurs in multiple niches simultaneously. In retrospect, there is only one single niche at any instant of time in the sequential covering approach, and clearly, this sole niche is accessed by all individuals during fitness evaluation; so the process of minimizing the problem space merely replaces the current niche with a smaller one, and as a consequence, does not truly mimic an evolutionary niching mechanism. This then provides the impetus for the next stage of our research – niched based evolution of neural logic networks.

Chapter 5

Niched Evolution of Probabilistic *Neulonets*

Following up on the identification of cognitive processes suitable for developing a cognitive-inspired learning architecture, we now turn towards an important branch of cognitive psychology – neuro-cognitive science. Indeed, biological inspiration from the human brain spearheaded the advent of a wide variety of connectionist neural network models which are widely recognized as mechanized implementations of the biological neurons. These connectionist learning models have contributed greatly to formulating theories of cognitive development, and conversely, some of these models are inspired from the approaches of developmental cognitive neuroscience in exploring interactions between brain and cognitive development. Within the field of cognitive development, there is the inevitable debate between two theoretical frameworks: selectionism and contructivism. Selectionists emphasize the competitive selection of neural elements with innate variation as the driving force of cortical representation. In contrast, constructivists argue that cortical representation is the result of growth and combination of neural elements that develop with experience. Representative work from these two camps

include the theory of neuronal group selection [Edelman, 1987] and neural constructivism [Quartz and Sejnowski, 1997] respectively. The work described in this paper is based on insights from the selectionist and constructivist schools of thought.

There has been many attempts in combining artificial neural network learning with evolutionary algorithms that amalgamates two primary forms of adaptation, i.e. neural network learning and evolution [Yao, 1999]. While learning in neural networks is typically accomplished by adjusting connection weights with a weight update rule through the iterative presentation of training situations or examples, the evolutionary aspect can be viewed at three levels: evolving connection weights, evolving entire neural architectures, or evolving algorithmic parameters. All three levels of evolution fundamentally address different inherent dispositions of traditional neural network learning. In our work, the premise is to look from the perspective of cognitive development. The theory of neuronal group selection describes a "primary repertoire" of rudimentary neural machinery that all human are born with. As discussed in chapter 1, a similar view is echoed in the form of adaptive decision heuristics and evolutionary psychology. Humans make complex decisions through the adaptation of rudimentary decision heuristics and strategies with repeated experiential learning.

As such we continue our endeavour of adopting a library of rudimentary neural networks, each of which prescribes a particular decision logic, and through adaptive evolution, these networks are composed together to form more intricate decisions with respect to a problem domain. As described in [Chia et al., 2009], the model is made of up two distinct components: the learning component and the interpretation component. Adaptation of rudimentary networks takes place in the learning component which remains oblivious to how these adaptations would be

used. The task of the interpretation component is then to aggregate the networks in the learning component and present them in a human amenable manner in the form of rules. Notably, the interpretation component resembles the idea of "left hemisphere interpretation" in [Gazzaniga, 1998] which is complementary to the evolutionary workings of the brain. There are two notable differences within the learning component from previous attempts at *neuolonet* evolution: the ability to deal with probabilistic information, as well as adopting a purely niched-based evolutionary paradigm. We shall first look at the variant of probabilistic neural logic networks.

5.1 Probabilistic neural logic network

Neural Logic Network Learning can be enhanced with the capability of processing probabilistic information [Teh, 1995] by allowing input nodes of the *neulonet* to take on real-numbered ordered pairs (x, y) in which the following constraint conditions hold.

$$0 \leq x \leq 1$$

$$0 \leq y \leq 1$$

$$0 \leq x + y \leq 1$$

A probabilistic *neulonet* with an input, say $(0.6, 0.3)$, implies that the input could be *true* 60% of the time and *false* 30% of the time; otherwise (i.e. for the remaining 10% of the time) it is *unknown*. In general, for an input node of value (x, y), Monte-Carlo simulation can be employed to generate successive uniformly distributed random points within the interval $[0, 1]$ such that if the point is less than x, the input node fires with a crisp value of $(1, 0)$; if the point is greater than $1 - y$, the input node fires with a value of $(0, 1)$; otherwise it fires with $(0, 0)$. The usual propagation and activation of the *neulonet* from input nodes to output node

can then proceed as normal. Suppose the simulation is performed on a *neulonet* for a sufficiently large number of trials, say n, and of these n trials, m_t resulted in a *true* outcome, and m_f resulted in a false outcome. Then the probabilistic outcome of the *neulonet* is given by $(\frac{m_t}{n}, \frac{m_f}{n})$. Compared to analytically solving the probabilistic outcomes, this naive methodology is attractive since evolution is inherently a stochastic process involving a large number of trials.

5.2 Adaptation in the learning component

Adaptation in the original work of neural logic network learning using genetic programming is purely evolutionary-based. Each *neulonet* is accorded a particular class label during initialization and evolution aims to generate fit *neulonets* with respect to their assigned classes. In this work, we take an alternative approach that extends the model of the *neulonet* to include a layer of neurons representing k class labels of a k-class problem as shown in figure 5.1. With the presentation of each training example to a *neulonet*, the real-valued weight w_i connecting Q to c_i is updated according to the weight update rule in equation (5.1). ρ represents the usual learning rate in traditional neural network learning and o_i takes on the value of 1 if the training example belongs to class i, or 0 otherwise. Q_α is the truth value of the outcome Q, i.e. 1 if Q is true and 0 otherwise. The optimum weight w^* in equation (5.2) of a particular iteration signifies the best action the *neulonet* could take at that point of time.

$$w_i \leftarrow w_i + \rho(o_i - w_i)Q_\alpha \tag{5.1}$$

$$w^* = \max_i w_i \tag{5.2}$$

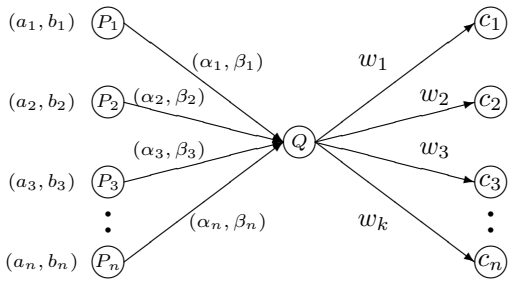

Figure 5.1: Neulonet with connecting weights to output nodes representing class labels.

The learning rule of equation (5.1) applied on each w_i makes w_i a recency-weighted estimate of o_i. Figure 5.2 shows the typical optimum weight w^* profiles of two *neulonets* using the weight update rule with learning rate ρ set to 0.01. The two *neulonets* correctly classifies the training instances 70% and 95% of the time over 2000 iterations; the latter is represented by the darker profile. Clearly, the variance is smaller for the better individual, while the weaker individual fluctuates more. Moreover, a larger learning rate results in larger weight fluctuations. As such, we adopt a small learning rate throughout our work. Weight update is activated only if the *neulonet* responds positively to the input training example, i.e. the outcome of Q is $(1, 0)$. This gives rise to an implicit niching-effect [Horn et al., 1994] which promotes the evolution of a diverse population.

Neulonets are evolved within their niches, thus avoiding the entire population from being overwhelmed by a single best individual. Genetic evolution immediately follows weight update after *every* presentation of a training example. This is in stark contrast to the previous approaches where evolution follows after individual *neulonets* have fired against all instances of the entire training data set. Roulette wheel selection is performed using the fitness values of the *neulonets* based on their

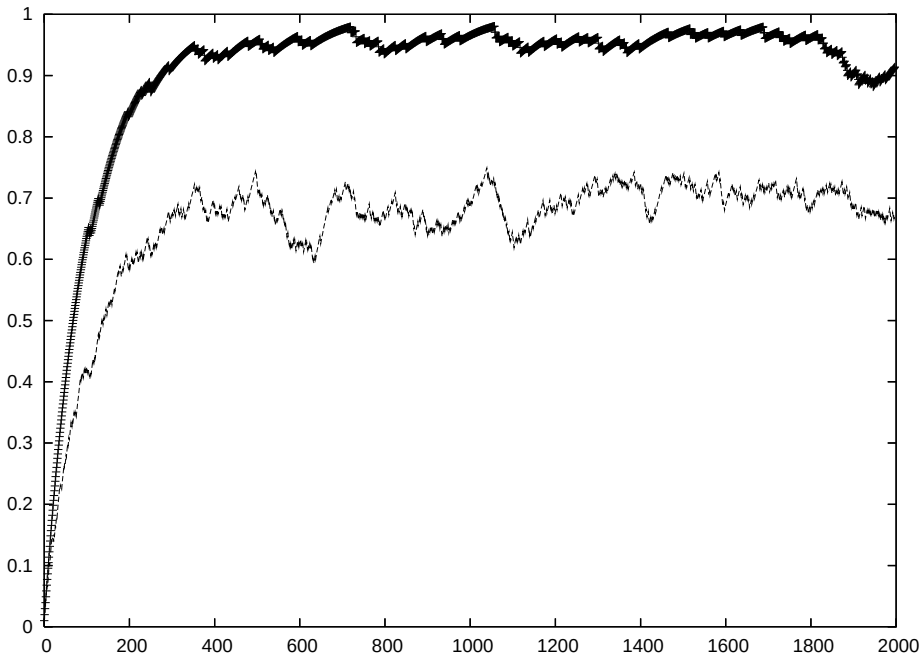

Figure 5.2: Weight profiles of two *neulonets*.

weight value. The fitness function employed is given in equation (5.3).

$$F = (w_c Q_\alpha)^\eta \tag{5.3}$$

Assuming that the training example belongs to class c, the fitness function is a power function of the corresponding weight w_c. The use of the power function is motivated from the original fitness function used in genetic programming. In its most rudimentary form, the fitness function is given by

$$f = \frac{1}{1+\epsilon}$$

where $\epsilon \in [0, |\mathbf{D}|]$ typically represents the number of misclassified instances in the data set $\mathbf{D}$. Let f_i denote the fitness for which $\epsilon = i$. Notice that f_0 (solution

with the optimum fitness) is 1, while f_1 (the next best solution) is 0.5. That is to say the fitness is halved. In the weight update rule of equation (5.1), we similarly desire that the solution with optimum weight $w_c = 1$ be accorded the fitness value of 1, while the next best solution with weight $w_c = 1 - \rho$ be accorded with fitness value of 0.5.

Let η be the power factor. We have

$$(1 - \rho)^{\eta} = 0.5 \Rightarrow \eta = -\frac{\log 2}{\log(1 - \rho)} \tag{5.4}$$

Once again, note that Q_α in equation (5.3) is required for niched evolution, i.e. only *neulonets* that respond to the training example will undergo selection and subsequent adaptation.

The evolution algorithm is summarized in table 5.1. Not only are genetic operations applied for every presentation of data examples, new offsprings are generated and included to the current population, rather than having the offsprings replace their parents. The specific genetic operator applied on a selected individual is probabilistic with a significantly higher probability placed on the reproduction operator. Throughout our investigation, the probability of subjecting an individual to reproduction is 80%, crossover 15% and mutation 5%. These values have been empirically found to provide adequate variation during evolution while maintaining a consistent pool of fit individuals [Koza, 1992]. Since crossover does not involve replacement, this higher rate of reproduction results in a larger number of copies (or higher *numerosity*) of fit individuals as evolution progresses. It is also interesting to note that in accordance to the *generalization hypothesis* proposed

in [Wilson, 1995], these fitter individuals are also maximally general, i.e. the individuals cover as much data examples as possible while remaining within some accuracy criterion. The hypothesis posits that in the presence of two individuals advocating the same action (or resolving the same class), and with one individual being a generalization of the other, the fact that the general one responds towards more training examples results in more reproductive opportunities. This would result in more copies of the general individual which would subsequently displace the specific individual.

Table 5.1: An algorithm for evolving *neulonets*.

1. Generate an initial population of *neulonets*.
2. Iteratively perform the following sub-steps for every presentation of a data example:
   - 2.1. Fire each *neulonet* in the population against the data instance
   - 2.2. Perform weight update on each *neulonet* with respect to the data instance
   - 2.3. Assign each *neulonet* with a fitness measure
   - 2.4 Apply genetic operations on *neulonets* chosen with a probability based on fitness to produce new offsprings.

5.3 The interpretation component

With the primary function of the learning component being the sole adaptation of neural networks, it is left to the interpretation component to make sense of these adaptations. We illustrate the use of the interpreter as a rule discovery system using the iris data set [Blake and Merz, 1998]. The data set consists of 150 data

instances. As the attributes are continuous valued, the data set undergoes a pre-processing phase of transforming the continuous values to ordered paired values amenable to *neulonet* training. Pre-processing involves discretizing the continuous data into discrete intervals using a discretization algorithm [Liu et al., 2002]; in this work we chose an entropy-based discretizer [Kohavi and Sahami, 1996]. Typically, the class labels present in the data set are used in entropy based discretizers so as to compute the entropy or information content. Intuitively, entropy based discretization aims to find the best split so that the discrete intervals have the majority of the values corresponding to the same class label. Formally, it is characterized by finding the split with the maximal information gain. Mere discretization produces only a crisp (true or false) decision as to whether a continuous value belongs to a given interval. However, probabilistic valued input can be simulated using an approach motivated from Parzen-based probability density function estimation using a triangular kernel [Parzen, 1962]. Intuitively, a continuous value located at the boundary of two adjacent discretized intervals gets an equal chance of falling on either interval. This leads to a triangular function which is dependent on the interval width. Specifically, discretization on the "sepal length" attribute of the iris data set produced three intervals with cutoff points located at 5.55 and 6.15. A point with value 5.7 induces a triangular kernel as shown in figure 5.3. A modified triangular Parzen kernel function was adopted for finding the degree of "fit" of a value with respect to its interval. Given a value k belonging to an interval $[I_i, I_{i+1}]$, i.e. $I_i < k < I_{i+1}$, the parzen kernel $W(x)$ is defined as:

$$W(x) = \begin{cases} 0 & \text{if } x \leq 2I_i - k \\ \frac{x - I_i}{2(k - I_i)} + \frac{1}{2} & \text{if } 2I_i - k < x \leq k \\ \frac{I_{i+1} - x}{2(I_{i+1} - k)} + \frac{1}{2} & \text{if } k > x > 2I_{i+1} - k \\ 0 & \text{if } x \geq 2I_{i+1} - k \end{cases} \tag{5.5}$$

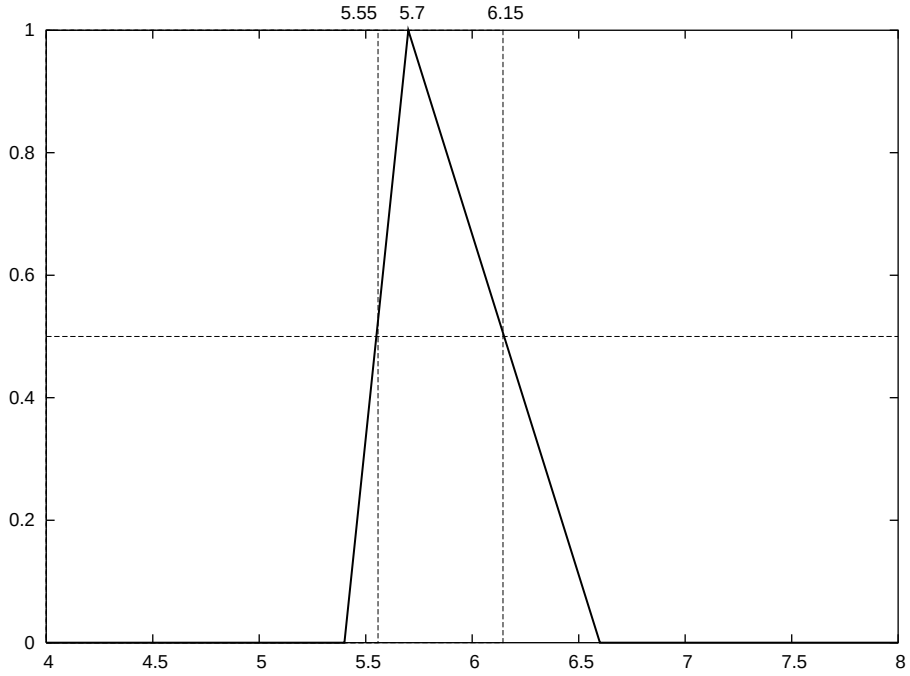

Figure 5.3: Triangular kernel for the value 5.7.

Superposition of kernels for all points with respect to the "sepal length" interval produces the profile shown in figure 5.4. The value 5.7 is associated with the "true" values of 0.288, 0.806 and 0.111 with respect to the three intervals $(-\infty, 5.55)$, $[5.55, 6.15)$ and $[6.15, \infty)$. The ordered pair inputs associated with the three intervals are $(0.239, 0.761)$, $(0.669, 0.331)$ and $(0.092, 0.908)$ respectively upon normalization. This pre-processed iris data set then undergoes adaptation in the learning component. An initial population of 1000 random *neulonets* is created, with each *neulonet* consisting of no more than two rudimentary *net rules*. The learning rate ρ is set at 0.01. Figure 5.5 depicts the number of copies of the first one hundred distinct *neulonets* sorted in descending order of numerosity at

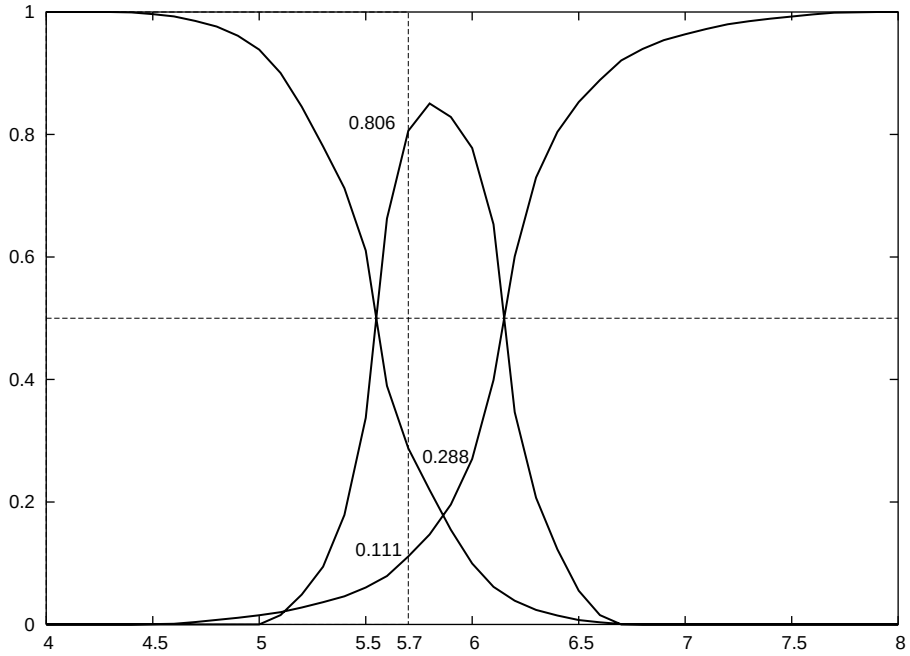

Figure 5.4: Final profiles of the three intervals for the "sepal length" attribute in iris.

100 and 1000 iterations. Notice that in each iteration, there is a significantly larger number of fitter individuals, and this has an overwhelming effect as the number of iterations increases.

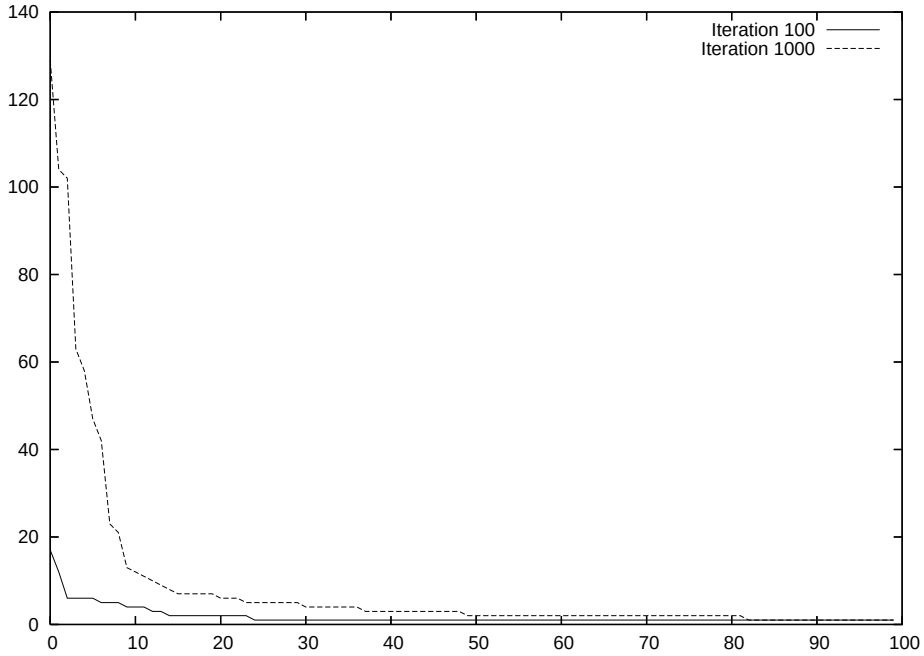

Figure 5.5: Numerosity profile of the hundred fittest *neulonets* in iterations 100 and 1000.

The interpretation component can now take on the role of a classifier and discover rules inherent in the problem domain. A similar sequential covering approach for classifier building in association-based *neulonet* evolution described the preceding chapter is used here to extract a minimal set of useful rules from the learning component [Michalski et al., 1986]. As an aside, note that the rule generation phase in the association-based approach is somewhat analogous to the learning component here, except that there is no pruning of data in the latter, and hence retains the essence of niched-based evolution. An outline of the algorithm is given in

table 5.2. Figure 5.6 depicts the classification accuracy over 3000 iterations after

Table 5.2: The sequential covering algorithm

1. Initialize classifier *RuleList* as empty
2. Iteratively perform the following until the data set is empty
   - 2.1. Select the *neulonet* with the largest numerosity from the learning component
   - 2.2. Delete from data set all examples covered by the *neulonet*
   - 2.3. If data was deleted, extract the rule from the *neulonet* and add it to the *RuleList*

employing sequential covering in intervals of 100 iterations. An early-stopping strategy is adopted and termination transpires when there is no improvement in accuracy over a period of 1000 iterations. The final classifier at iteration 1900 attains the accuracy of 95.3% using seven rules.

5.4 Empirical study and discussion

An empirical study was conducted using several continuous valued data sets from the UCI Data Repository [Blake and Merz, 1998]. Ten-fold cross validation is employed in which that training set is used for *neulonet* evolution and the discovered rules are tested against the test set. Results using our approach (SNC) are compared against the preceding work of *neulonet* association-based classification [Chia et al., 2006] (NAC) in which sequential covering is similarly used for rule discovery. Results are tabulated in table 5.3. Generally, our approach produces more accurate classifiers but at the expense of a greater number of rules. This is explained by the fact that the sequential covering implemented within the interpretation component uses only the accuracy of the rule list generated, coupled

Figure 5.6: Classification accuracy over a period of 3000 iterations for iris data set.

with early stopping, to decide on the optimal classifier. The size of the rule list is not taken into consideration, though it has to be mentioned here that the size of the *neulonets* generated are kept small. This is because the size of the *neulonet* is a good proxy for generality, and these more general rules make up the majority of the population, i.e. they have higher numerosities.

It has to be mentioned here that NAC employs an initial population of 10000 individuals and this number is maintained throughout the evolution process. However, our approach starts off with 1000 individuals, and incrementally adds individuals as evolution proceeds. This gives rise to a steady but linear increase in the population with respect to the training iterations. The increase in population does not pose a system limitation as ineffective *neulonets* gets purged. Purging is implemented by taking note of an individuals "age". *Neulonets* which are deemed too

Table 5.3: Experimental results depicting predictive errors (percent) of different classifiers. Values within brackets denote the average number of rules in the classifier.

Data Set	NAC		SNC	
glass	22.7	(23.6)	24.6	(32.4)
iono	7.3	(21.8)	6.7	(16.5)
iris	5.7	(6.2)	0.9	(11.0)
pima	24.7	(19.8)	21.8	(31.0)
wine	9.0	(5.2)	5.6	(16.2)

old are removed from the population if the total number of individuals exceed a stipulated bound on the population, which is set to 10000.

Copies of similar *neulonets* can be simulated with a *numerosity* parameter attached to every *neulonet*. This facilitates the fast discovery of similar individuals and speeds up the evolution process. In addition, the *neulonets* picked by the interpretation component are simplified using a set of simplification rules prior to presentation. However, simplification is not performed on the evolution population *per se*, as it is desirable to retain variability and diversity during evolution.

5.5 Summary

The work in this chapter describes an approach of adapting a library of rudimentary neural networks through evolution and minimal weight modifications to facilitate the process of rule discovery and classification, which is inspired from the selectionist and constructivist schools of thought in cognitive neuroscience. The separation of the learning component from the interpretation component allows for task separation. This is well suited for situations involving dynamic data where the problem

domain constantly changes. Since evolution is performed for every data instance, *neulonet* learning can assimilate the variances encountered as time progresses without the need to restart the evolution procedure from scratch. Although the rule generation and classifier building phases in association-based *neulonet* evolution suggest a similar separation, the delineation is less clear. In the case of dynamic data, when new data instances arrive, rule generation has to be performed from the beginning as the task of rule generation is to generate a list of *neulonets* for every class label, which in turn will be filtered during classifier building.

More importantly, the connotations of "ecological rationality" are completely imbued in the niched evolutionary platform, i.e. the rules generated from the interpretation component can be interpreted independently from one another. Similar work has also been done in the form of a learning classifier systems and its associated variants [Holland, 1986, Wilson, 1995] in which a genetic algorithm is typically used for niched evolution of bit patterns for both supervised and reinforcement learning.

Extending the crisp variant of neural logic networks to allow for probabilistic values also makes the system more robust. In the previous attemps of *neulonet* evolution, learning with continuous data is dealt solely with discretization such that all values that belongs to an interval are "binarized" to be either within that interval or not. There is no relationship of the occurrence of a data point with respect to all other points within that interval in terms of its probability density. Incorporating Parzen probability density estimation with entrophy discretization retains constructive information on the reliability of the continuous value with respect to its interval, and its effects on neighbouring intervals. Coupled with Monte-Carlo simulation as part of evolution, the resulting system can now deal with probabilistic information in situations of uncertainty which is rampant in human decision making.

Chapter 6

Towards a Cognitive Learning System

The work described in this thesis is driven by the need to account for human cognitive processes during decision making which paves a way for other endeavours in realizing "cognitive-inspired" knowledge discovery systems that generate useful models or rules for human analysis and verification. We began with a literature review that is primarily focused on decision making in cognitive psychology and behavioural decision science to address the problems of rule comprehensibility and utility in knowledge discovery systems. Two fundamental aspects were identified. Firstly, human decision making is based upon the application of rudimentary heuristics and strategies that people amass through the course of experiential learning, coupled with the way to handle uncertain situations which plays an important part in the decision being made. Secondly, the suggestion from evolutionary psychology that people are adaptive decision makers with respect to their environments. This forms the pillars of research into the eventual realization of a cognitive-inspired knowledge discovery. Specifically, the advent of neural logic

networks enables us to mimic the repertoire of decision heuristics, with the probablistic variant used to represent situations involving uncertain or vague decision making. In addition, there is a requirement to make rational (both bounded and ecological) decisions through the use of a genetic programming platform by evolving *neulonets* (probabilistic or otherwise) to find an optimal solution to a problem (the "problem" here refers classification or supervised learning)

Neural logic network research, which began more than a decade ago, provided us with the starting platform from which this research is based. Through the early empirical studies on evolution of neural logic networks, it was observed that the goal of mimicking "human decision making" in order to discover useful "human amenable" knowledge was still found wanting. As such, studies in human decision making in cognitive sciences were looked into in order to improve the "human amenable" aspect of the knowledge discovery system. Indeed, it was heartening to see that neural logic network with its fundamental *net rule* library was a realization of the toolbox of decision heuristics that humans employ in decision making. A noteworthy mention should be given to these *net rules*. Each *net rule* is basically a neural network with only one-level of connecting weights between input and output nodes, with no hidden nodes. As such, they represent simple oblique decision hyperplanes or linearly separable rules, and thus meet the elementary requirement of basic decision making units, without being overly complex like a multi-layer network. Moreover, only a limited set of weights are applicable for *net rule* specification.

In order to maintain the structural integrity of the *neulonet* while learning, evolutionary learning is employed with carefully devised genetic operators for crossover and mutation. The endeavour to evolve useful *neulonets* began with a rudimentary

genetic programming platform that evolves a single optimum *neulonet* for classifying two-class problems, with binarized data attributes. This required immense computational time and resources. The need to perform evolution within a boundedly rational context resulted in an attempt to evolve multiple compact *neulonets* in association-based evolution for solving multi-class problems. These *neulonets* are composed together in a final rule list to form an eventual classifier. Despite the evolution of separate *neulonets*, the fact that *neulonet* generation involved the incremental purging of data instances, resulted in a dependence between the rules. This led to a niched-evolution platform that evolves *neulonets* within an essentially ecologically rational context, i.e. each *neulonet* is evolved within its intended environment. In addition, the schema of the neural logic network is extended to include a layer of output nodes with connecting weights for the different class values in the data set. This is motivated by the notion of "constructivism" in neuroscience where some form of weight update is deemed necessary in neural learning. This weight update serves the purpose of tracking its "strength" with respect to the different classes, and does not play a part in the genetic operations. As such the semantics of each *neulonet* is still intact. However, the fact that *neulonet* evolution still begins with a primary repertoire of networks supports the other "selectionist" school of thought.

Uncertainty reasoning in neural logic network learning is achieved using the probabilistic variant in which degrees of truth and falsity are specified. This allows the system to cater to the notion of non-determinacy, i.e. ambiguity or vagueness. Rather than analytically solving the probabilistic outcomes, we have adopted the naive strategy of employing Monte Carlo simulation since the underlying evolutionary platform is ultimately a stochastic procedure. This allows us to maintain the simplicity in implementation of the learning system.

Till date, knowledge discovery from data is still a process that involves the interaction between a data mining expert and the domain expert. Generally, feedback from the domain expert is used to fine-tune the system by adjusting system parameters in order to find an acceptable and agreeable solution. The absence of quantifiable parameters or describable processes with respect to the novelty, utility and understandability of the inferences generated by the system makes the above a difficult and laborious task involving many iterations of trial-and-error adjustments. As such, more research based on cognitive processes identified from cognitive psychology and behavioural decision science can be done. For mined knowledge to be accepted, the biases of human learners are of utmost relevance. There is a vast and rich literature on human learning, category representations, and cognitive psychology methodologies which can be integrated into varying aspects of a learning system. Much of the findings in cognitive psychology are undoubtedly relevant to the KDD and machine learning community, as we expect much more from knowledge discovery tools other than the creation of accurate classification and prediction models. The long-term goal is to fully realize the benefits of data mining by paying attention to the cognitive factors that make the learned models coherent, credible, easy to use and communicate to others.

Bibliography

[Agrawal et al., 1993] Agrawal, R., Imielinski, T., and Swami, A. (1993). Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216.

[Andrews et al., 1995] Andrews, R., Diederich, J., and Tickle, A. B. (1995). A survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-Based Systems*, 8(6):373–379.

[Angeline, 1994] Angeline, P. J. (1994). Genetic programming and emergent intelligence. In Kinnear, K. E., editor, *Advances in Genetic Programming*, pages 75–97. MIT Press, Cambridge Mass.

[Blake and Merz, 1998] Blake, C. and Merz, C. (1998). UCI repository of machine learning databases. University of California, Irvine, Dept. of Information and Computer Sciences.

[Carruthers, 2006] Carruthers, P. (2006). Simple heuristics meet massive modularity. In Carruthers, P., Laurence, S., and Stitch, S., editors, *The Innate Mind: Culture and Cognition*. Oxford University Press.

[Chia and Tan, 2004a] Chia, H. W.-K. and Tan, C.-L. (2004a). Association-based evolution of comprehensible neural logic networks. In Keijzer, M., editor, *Late*

Breaking Papers at the 2004 Genetic and Evolutionary Computation Conference, Seattle, Washington, USA.

[Chia and Tan, 2004b] Chia, H. W.-K. and Tan, C.-L. (2004b). Confidence and support classification using genetically programmed neural logic networks. In Keijzer, M., editor, *Proceedings of the 2004 Genetic and Evolutionary Computation Conference*, volume 2, pages 836–837, Seattle, Washington, USA.

[Chia et al., 2006] Chia, H. W. K., Tan, C. L., and Sung, S. Y. (2006). Enhancing knowledge discovery via association-based evolution of neural logic networks. *IEEE Transactions on Knowledge and Data Engineering*, 18(7):889–901.

[Chia et al., 2009] Chia, H. W. K., Tan, C. L., and Sung, S. Y. (2009). Probabilistic neural logic network learning: Taking cues from neuro-cognitive processes. In *Proceedings of the 2009 21st IEEE International Conference on Tools with Artificial Intelligence*, pages 698–702.

[Cosmides and Tooby, 1997] Cosmides, L. and Tooby, J. (1997). Evolutionary psychology: A primer. http://www.psych.ucsb.edu/research/cep/primer.html.

[Craven and Shavlik, 1999] Craven, M. and Shavlik, J. (1999). Rule extraction: Where do we go from here? University of Wisconsin Machine Learning Research Group Working Paper 99-1.

[Domingos, 1999] Domingos, P. (1999). The role of occam's razor in knowledge discovery. *Data Mining and Knowledge Discovery*, 3(4):409–425.

[Edelman, 1987] Edelman, G. M. (1987). *Neural Darwinism – The Theory of Neuronal Group Selection*. Basic Books.

[Edmonds, 1999] Edmonds, B. (1999). Modelling bounded rationality in agent-based simulations using the evolution of mental models. In *Computational Techniques for Modelling Learning in Economics*, pages 305–332. Kluwer.

[Fayyad et al., 1996] Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI Magazine*, 17:37–54.

[Fisher and McKusick, 1989] Fisher, D. H. and McKusick, K. B. (1989). An empirical comparison of ID3 and back-propagation. In Sridharan, N., editor, *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, pages 788–793, Detroit, MI, USA. Morgan Kaufmann.

[Forster, 1999] Forster, M. R. (1999). How do simple rules 'fit to reality' in a complex world? *Minds and Machines*, 9:543–564.

[Frawley et al., 1992] Frawley, W. J., Piatetsky-Shapiro, G., and Matheus, C. J. (1992). Knowledge discovery in databases - an overview. *Ai Magazine*, 13:57–70.

[Freitas, 2002] Freitas, A. (2002). A survey of evolutionary algorithms for data mining and knowledge discovery. In Ghosh, A. and Tsutsui, S., editors, *Advances in Evolutionary Computation*, chapter 33, pages 819–845. Springer-Verlag.

[Freitas, 1999] Freitas, A. A. (1999). On Rule Interestingness Measures. *Knowledge-Based Systems*, 12(5-6):309–315.

[Freitas, 2000] Freitas, A. A. (2000). Understanding the crucial differences between classification and discovery of association rules—a position paper. *SIGKDD Explorations*, 2(1):65–69.

[Gaudet, 1996] Gaudet, V. C. (1996). Genetic programming of logic-based neural networks. In Pal, S. K. and Wang, P. P., editors, *Genetic Algorithms for Pattern Recognition*. CRC Press, Boca Raton.

[Gazzaniga, 1998] Gazzaniga, M. S. (1998). *The Mind's Past*. University of California Press.

[Gigerenzer, 2001] Gigerenzer, G. (2001.). The adaptive toolbox. In Gigerenzer, G. and Selten, R., editors, *Bounded Rationality: The Adaptive Toolbox*. MIT Press.

[Gigerenzer, 2002] Gigerenzer, G. (2002). The adaptive toolbox: Toward a darwinian rationality. In Backman, L. and von Claes Hofsten, editors, *Psychology at the Turn of the Millennium: Cognitive, Biological and Health Perspectives*. Taylor & Francis.

[Gigerenzer and Todd, 1999] Gigerenzer, G. and Todd, P. (1999). *Simple heuristics that make us smart*. Oxford University Press, New York.

[Holland, 1986] Holland, J. H. (1986). Escaping brittleness: The possibilities of general-purpose learning algorithms applied to parallel rule-based systems. In Michalski, R. S., Carbonell, J. G., and Mitchell, T. M., editors, *Machine Learning: An Artificial Intelligence Approach: Volume II*, pages 593–623. Kaufmann, Los Altos, CA.

[Horn et al., 1994] Horn, J., Goldberg, D. E., and Deb, K. (1994). Implicit Niching in a Learning Classifier System: Nature's Way. *Evolutionary Computation*, 2(1):37–66.

[Kahneman and Tversky, 1982] Kahneman, D. and Tversky, A. (1982). Variants of uncertainty. *Cognition*, 11:143–157.

[Kishore et al., 2000] Kishore, J. K., Patnaik, L. M., Mani, V., and Agrawal, V. K. (2000). Application of genetic programming for multicategory pattern classification. *IEEE Transactions on Evolutionary Computation*, 4(3):242–258.

[Kleene, 1952] Kleene, S. C. (1952). *Introduction to MetaMathematics*. van Nostrand, Princeton.

[Kohavi and Sahami, 1996] Kohavi, R. and Sahami, M. (1996). Error-based and entropy-based discretization of continuous features. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 114–119.

[Koza, 1992] Koza, J. R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, Mass.

[Levenick, 1991] Levenick, J. R. (1991). Inserting introns improces genetic algorithm success rate: Taking a cue from biology. In Belew, R. K. and Booker, L. B., editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 123–127, San Matro, CA.

[Lipshitz and Strauss, 1997] Lipshitz, R. and Strauss, O. (1997). Coping with uncertainty: A naturalistic decision-mkaing analysis. *Organizational Behavior and Human Decision Processes*, 69(2):149–163.

[Liu et al., 1998] Liu, B., Hsu, W., and Ma, Y. (1998). Integrating classification and association rule mining. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pages 80–86.

[Liu et al., 2002] Liu, H., Hussain, F., Tan, C. L., and Dash, M. (October 2002). Discretization: An enabling technique. *Data Mining and Knowledge Discovery*, 6:393–423(31).

[Manson, 2005] Manson, S. M. (2005). Genetic programming as an isomorphic analog to bounded rationality in agent-based models. In *Proceedings of GeoComputation 2005*.

[Michalski et al., 1986] Michalski, R. S., Mozetic, I., Hong, J., and Lavrac, N. (1986). The multi-purpose incremental learning system AQ15 and its testing application to three medical domains. In *Proceedings of the 5th national conference on Artificial Intelligence*, pages 1041 – 1045, Philadelphia.

[Michie, 1988] Michie, D. (1988). The Fifth Generation's unbridged gap. In Herken, R., editor, *The Universal Turing machine : a half-century survey*, pages 467–489. Oxford University Press, Oxford.

[Minsky, 1991] Minsky, M. (1991). Logical Versus Analogical or Symbolic Versus Connectionist or Neat Versus Scruffy. *AI Magazine*, 12(2):34–51.

[Murphy and Pazzani, 1991] Murphy, P. M. and Pazzani, M. J. (1991). ID2-of-3: Constructive induction of m-of-n concepts for discrminators in decision trees. In *Proc. of the Eighth International Workshop on Machine Learning*, pages 183–187, Evanston, IL.

[Newell and Simon, 1972] Newell, A. and Simon, H. A. (1972). *Human Problem Solving*. Prentice-Hall, Englewood Cliffs, NJ.

[Parzen, 1962] Parzen, E. (1962). On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076.

[Payne et al., 1993] Payne, J. W., Bettman, J. R., and Luce, M. F. (1993). *The Adaptive Decision Maker*. Cambridge University Press.

[Payne et al., 1998] Payne, J. W., Bettman, J. R., and Luce, M. F. (1998). Behavioral decision research: An overview. In Birnbaum, M. H., editor, *Measurement, Judgment, and Decision Making*. Academic Press, San Deigo.

[Pazzani, 2000] Pazzani, M. J. (2000). Knowledge discovery from data? *IEEE Intelligent Systems*, 15(2):10–13.

[Piatetsky-Shapiro and Matheus, 1994] Piatetsky-Shapiro, G. and Matheus, C. J. (1994). The interestingness of deviations. In Fayyad, U. M. and Uthurusamy, R., editors, *AAAI Workshop on Knowledge Discovery in Databases (KDD-94)*, pages 25–36, Seattle, Washington. AAAI Press.

[Quartz and Sejnowski, 1997] Quartz, S. R. and Sejnowski, T. J. (1997). The neural basis of cognitive development: A constructivist manifesto. *Behavioral and Brain Sciences*, 20:537–596.

[Quinlan, 1993] Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA.

[Rivest, 1987] Rivest, R. L. (1987). Learning decision lists. *Machine Learning*, 2(3):229–246.

[Schlimmer, 1987] Schlimmer, J. C. (1987). *Concept acquisition through representational adjustment*. PhD thesis, Department of Information and Computer Science, University of California, Irvine, CA.

[Shafer and Pearl, 1990] Shafer, G. and Pearl, J. (1990). *Readings in Uncertain Reasoning*. Kaufmann, San Mateo, CA.

[Silberschatz and Tuzhilin, 1995] Silberschatz, A. and Tuzhilin, A. (1995). On subjective measures of interestingness in knowledge discovery. In *Knowledge Discovery and Data Mining*, pages 275–281.

[Simon, 1955] Simon, H. A. (1955). A behavioral model of rational choice. *Quaterly Journal of Economics*, 69(1):99–118.

[Simon, 1990] Simon, H. A. (1990). Invariants of human behavior. *Annual Review of Psychology*, pages 1–19.

[Stenning and van Lambalgen, 2008] Stenning, K. and van Lambalgen, M. (2008). *Human Reasoning and Cognitive Science*. MIT Press, Cambridge, MA, USA.

[Tan and Chia, 2001a] Tan, C. L. and Chia, H. W. K. (2001a). Genetic construction of neural logic network. In *Proceedings of INNS-IEEE International Joint Conference on Neural Networks*, volume 1, pages 732–737, Piscataway, N.J.

[Tan and Chia, 2001b] Tan, C. L. and Chia, H. W. K. (2001b). Neural logic network learning using genetic programming. In *Proceedings of Seventeenth International Joint Conference on Artificial Intelligence*, pages 803–808, Menlo Park, Calif.

[Tan et al., 1996] Tan, C. L., Quah, T. S., and Teh, H. H. (1996). An artificial neural network that models human decision making. *Computer*, 29(3):64–70.

[Tan et al., 2002] Tan, P.-N., Kumar, V., and Srivastava, J. (2002). Selecting the right interestingness measure for association patterns. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 32–41, New York, NY, USA. ACM Press.

[Teh, 1995] Teh, H. H. (1995). *Neural Logic Network, A New Class Of Neural Networks*. World Scientific, Singapore.

[Tooby and Cosmides, 1992] Tooby, J. and Cosmides, L. (1992). The psychological foundations of culture. In Barkow, J. H., Cosmides, L., and Tooby, J., editors, *The Adapted Mind: Evolutionary Psychology and the Generation of Culture*, pages 19–136. Oxford University Press, New York.

[Towell and Shavlik, 1993] Towell, G. G. and Shavlik, J. W. (1993). Extracting refined rules from knowledge-based neural networks. *Machine Learning*, 13:71–101.

[Towell and Shavlik, 1994] Towell, G. G. and Shavlik, J. W. (1994). Knowledge-based artificial neural networks. *Artificial Intelligence*, 70(1-2):119–165.

[Tsakonas et al., 2004] Tsakonas, A., Aggelis, V., Karkazis, I., and Dounias, G. (2004). An evolutionary system for neural logic networks using genetic programming and indirect encoding. *Journal of Applied Logic*, 2:349–379.

[Tsakonas et al., 2006] Tsakonas, A., Dounias, G., Doumpos, M., and Zopounidis, C. (2006). Bankruptcy prediction with neural logic networks by means of grammar-guided genetic programming. *Expert Syst. Appl.*, 30(3):449–461.

[Utgoff and Brodley, 1991] Utgoff, P. E. and Brodley, C. E. (1991). Linear machine decision trees. Technical Report UM-CS-1991-010, University of Massachusetts.

[Wilson, 1995] Wilson, S. W. (1995). Classifier fitness based on accuracy. *Evolutionary Computation*, 3(2):149–175.

[Windschitl and Wells, 1996] Windschitl, P. D. and Wells, G. L. (1996). Measuring psychological uncertainty: Verbal versus numeric methods. *Journal of Experimental Psychology: Applied*, 2:343–364.

[Yao, 1999] Yao, X. (1999). Evolving artificial neural networks. *PIEEE: Proceedings of the IEEE*, 87.