

Probabilistic Neural Logic Network Learning: Taking Cues From Neuro-Cognitive Processes

Henry Wai Kit Chia
School of Computing
Nat. Univ. of Singapore
13 Computing Drive
Singapore 117417
hchia@comp.nus.edu.sg

Chew Lim Tan
School of Computing
Nat. Univ. of Singapore
13 Computing Drive
Singapore 117417
tancl@comp.nus.edu.sg

Sam Y Sung
Dept of Computer Science
South Texas College
McAllen TX 78501
sysung@southtexascollege.edu

Abstract

This paper describes an attempt to devise a knowledge discovery model that is inspired from the two theoretical frameworks of selectionism and constructivism in human cognitive learning. The “selectionist” nature of human decision making indicates the use of an evolutionary paradigm for composing rudimentary neural network units, while the “constructivist” component takes the form of neural weight training during the learning process. We explore the possibility of amalgamating these two ideas into a neural learning system for the discovery of meaningful rules in the context of pattern discovery in data.

1. Introduction

An important measure of utility in any machine learning model is the acceptability or amenability of learned knowledge in terms of their usefulness subject to the human user. One way to achieve this goal within the realm of data mining and knowledge discovery from data is to account for cognitive processes that humans employ in decision making [10]. Theories of cognitive development include the two theoretical frameworks of selectionism and constructivism. Selectionists emphasize the competitive selection of neural elements with innate variation as the driving force of cortical representation [3], while, constructivists argue that cortical representation is the result of growth and combination of neural elements that develop with experience [11].

While there has been many attempts in combining artificial neural network learning with evolutionary algorithms that fundamentally addresses inherent dispositions of traditional neural network weight training [15], the premise of our work is to look from the perspective of cognitive development. Humans make complex decisions through

the adaptation of rudimentary decision heuristics and strategies with repeated experiential learning. Briefly, our work adopts 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. The model we propose is made of up two distinct components: the learning component and the interpretation component. Adaptation of rudimentary networks takes place in the learning component (section 2) which remains oblivious to how these adaptations would be used. The task of the interpretation component (section 3) 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 [4] which is complementary to the evolutionary workings of the brain. Section 4 presents empirical findings on some benchmark data sets and discusses the implications of our work.

2. The learning component

We use a probabilistic variant of neural logic networks (*neulonets*) [13] as rudimentary units for representing fundamental human decision logic. A *neulonet* has an ordered pair of numbers associated with each node and connection. The ordered pair for input and output node takes one of three discrete values: (1, 0) for “true”, (0, 1) for “false” and (0, 0) for “don’t know”. (1, 1) is undefined. Equation (1) defines the activation function at the output Q with input denoted by (a_i, b_i) , and the weights (α_i, β_i) .

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

The “Disjunction” *neulonet* shown in figure 1. depicts a disjunction rule whose outcome is in accordance to Kleene’s three-valued logic model. Rudimentary *neulonets* (or *net*

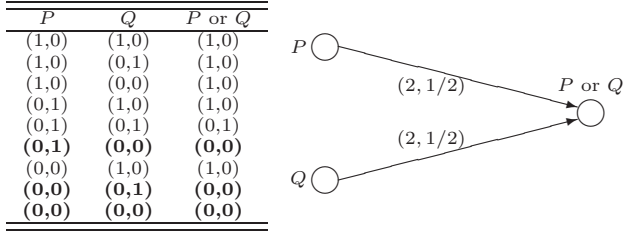


Figure 1. Disjunction net rule.

rules) can be devised to express a variety of human-like logic which can be broadly divided into five categories as shown in figure 2. *Neulonets* can also be enhanced to process probabilistic information by allowing input nodes to take on real-numbered ordered pairs (x, y) with the constraints $0 \leq x \leq 1$, $0 \leq y \leq 1$ and $0 \leq x + y \leq 1$. 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; for the remaining 10% of the time, it is *unknown*. For an input node of value (x, y) , successive uniformly distributed random points can be generated within the interval $[0, 1]$ such that if the point is less than x , the input node fires with a 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 suited for evolutionary learning that inherently involves a large number of trials.

Neulonets can be combined into larger composite *neulonets* to realize more complex decisions. *Neulonet* composition has been demonstrated using a genetic programming platform [12, 2]. One iteration of the learning process involves subjecting the entire population of *neulonets* to a set of training examples and selection of desirable *neulonets* implemented using a fitness measure such that a *neulonet* of higher utility is accorded an exponentially higher chance of selection. Selected *neulonets* then go through genetic operations of reproduction, crossover or mutation. These operations maintain the semantics of the constituent *net rules* as connection weights remain intact, hence facilitating the straightforward extraction of the logic rules they represent.

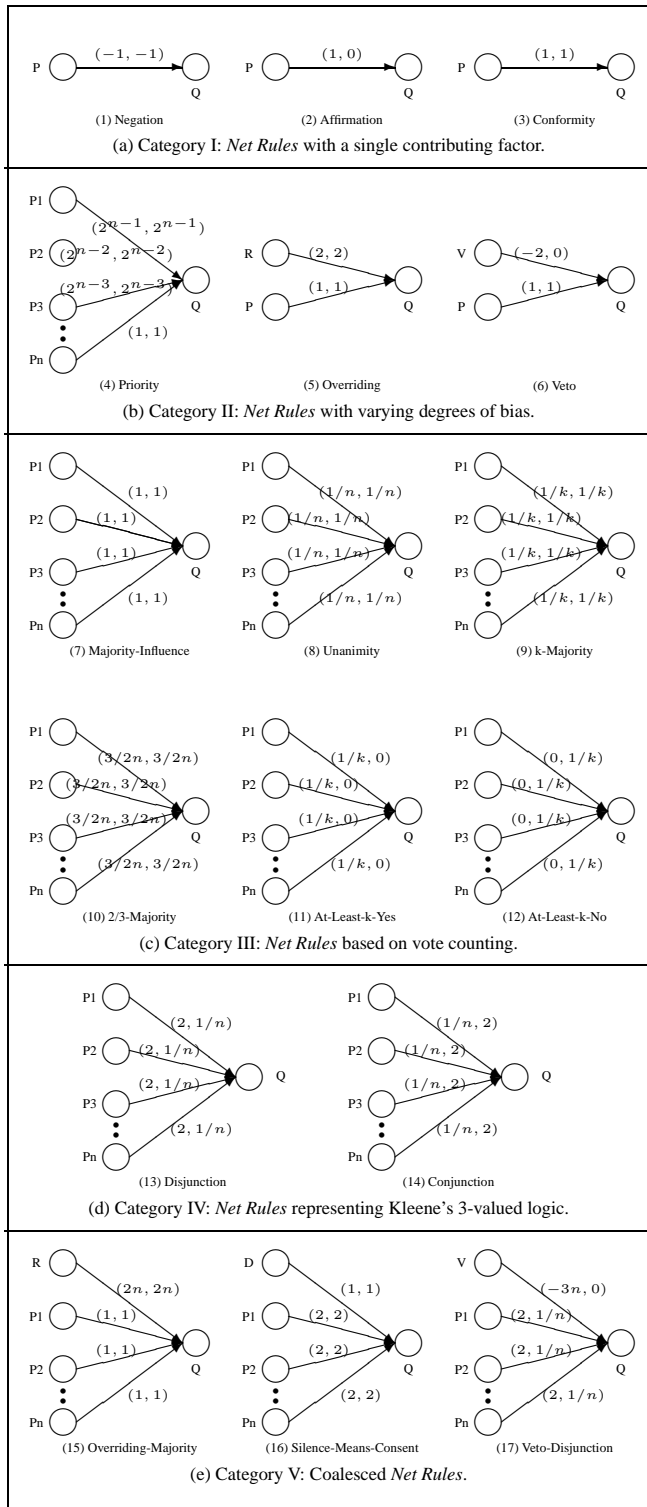


Figure 2. Library of net rules divided into five broad categories

2.1 Selectionist and Constructivist Learning

The preceding work of evolutionary neural logic network learning sees each *neulonnet* accorded a particular class label during initialization. Evolution aims to generate fit *neulonnets* with respect to their assigned classes. We extend the model of the *neulonnet* to include a layer of neurons representing k class labels of a k -class problem as shown in figure 3. With the presentation of each training example to a *neulonnet*, the real-valued weight w_i connecting Q to c_i is updated according to the weight update rule in equation (2). κ 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 . The optimum weight w^* in equation (3) corresponds to the classification of the *neulonnet*.

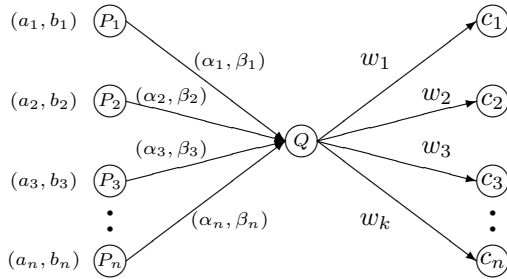


Figure 3. Neulonnet with connecting weights to output nodes representing class labels.

$$w_i \leftarrow w_i + \kappa(o_i - w_i)Q_\alpha \quad (2)$$

$$w^* = \max_i w_i \quad (3)$$

The learning rule applied on each w_i makes w_i a recency-weighted estimate of o_i . Figure 4 shows the typical optimum weight w^* profiles of two *neulonnets* using the weight update rule with learning rate κ set to 0.01. The two *neulonnets* 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. Moreover, a larger learning rate results in larger weight fluctuations.

Genetic evolution immediately follows weight update for each presentation of a training example. Roulette wheel selection is performed using the fitness values of the *neulonnets* based on their weight value. The fitness function employed is given in equation (4). Assuming that the training example belongs to class c , the fitness function is a power function¹ of the corresponding weight w_c .

$$F = (w_c Q_\alpha)^\eta \quad (4)$$

¹We adopted an η value of $-\frac{\log 2}{\log(1-\alpha)}$ which is inspired from the original fitness function in [7]; the derivation is left out due to space constraints.

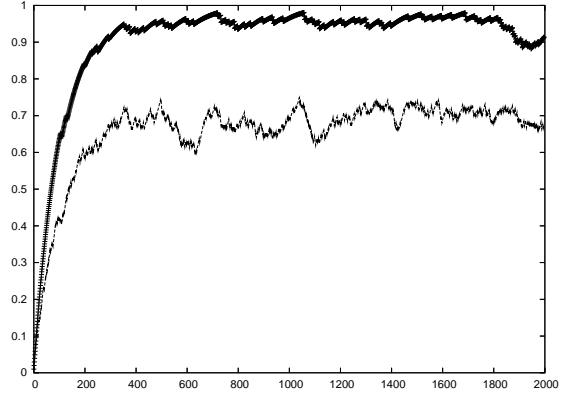


Figure 4. Weight profiles of two neuronets.

The evolution algorithm is summarized in table 1. 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%.

Table 1. An algorithm for evolving neuronets.

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

Weight update and evolutionary selection is activated only if the *neulonnet* responds positively to the input training example, i.e. the outcome of Q is $(1, 0)$ (or $Q_a = 1$). This gives rise to an implicit niching-effect [5] where *neulonnets* are evolved within their niches, thus avoiding the entire population from being overwhelmed by a single best individual. A high reproduction rate also 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 [14], these fitter individuals are also maximally general, i.e. the individuals cover as much data examples as possible while remaining within some accuracy criterion. In the presence of two individuals advocating the same action, with one individual being a generalization of the other, the general one responds towards more training examples and results in more reproductive opportunities.

3 The interpretation component

The interpretation component for rule discovery is illustrated using the iris data set of 150 data instances with continuous attributes. Each attribute is discretized into intervals with an entropy-based discretizer [6], followed by Parzen-based probability density function estimation using a triangular kernel [9], to generate probabilistic ordered-pair values. Intuitively, a continuous value located at the boundary of two adjacent discretized intervals gets an equal chance of falling on either interval. Using the “sepal length” attribute as an example, discretization produced three intervals with cutoff points located at 5.55 and 6.15. A point with value 5.7 induces an interval-width dependent triangular kernel as shown in figure 5. Superposition of

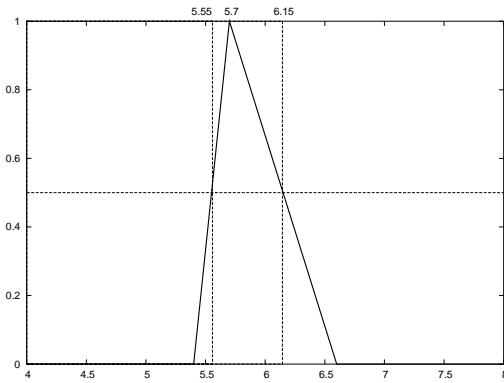


Figure 5. Triangular kernel for the value 5.7.

kernels for all points produces the profiles in figure 6. The value 5.7 is associated with the “true” values of 0.288, 0.806 and 0.111 with respect to the three intervals $\{(-\infty, 5.55), [5.55, 6.15), [6.15, \infty)\}$. The ordered pair inputs associated with the three intervals are (0.239, 0.761), (0.669, 0.331) and (0.092, 0.908) upon normalization.

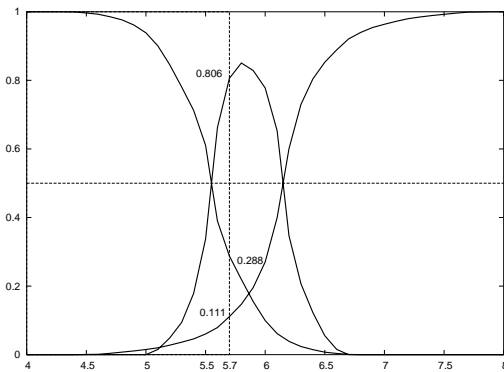


Figure 6. Profiles for “sepal length” attribute.

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 κ is set at 0.01 to avoid significant weight fluctuations. Figure 7 depicts the number of copies of the first one hundred distinct *neulonets* sorted in descending order of numerosity at 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.

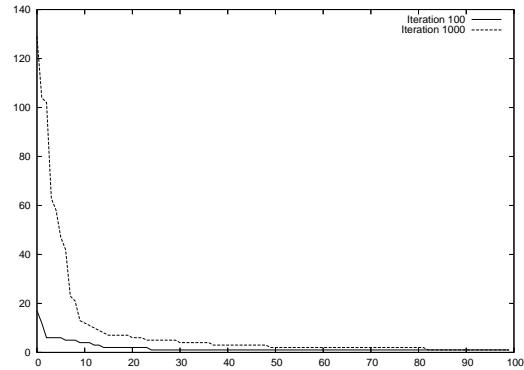


Figure 7. Numerosity profile of the hundred fittest neulonets in iterations 100 and 1000.

The interpretation component can now take on the role of a classifier and discover rules inherent in the problem domain. A sequential covering approach is used to extract a minimal set of useful rules from the learning component [8]. An outline of the algorithm is given in table 2. For the iris data set, sequential covering is applied periodically after 100 evolution iterations. An early-stopping strategy is adopted and termination transpires when there is no improvement in accuracy over a period of 1000 iterations. In a typical simulation run, the classification accuracy exceeds 90% after the first 1000 iterations.

Table 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 for which the *neulonet* produces a “true” outcome.
 - 2.3. If data was deleted, extract the rule from the *neulonet* and add it to the *RuleList*.

4 Empirical Study and Discussion

An empirical study was conducted using a number of continuous valued data sets from [1]. Ten-fold cross validation results using our approach (SNC) are compared against the work of [2] (NAC) in which sequential covering is similarly used for rule discovery. Results of average predictive errors (as well as the average size of the rules in brackets) are tabulated in table 3. Generally, SNC produces more accurate classifiers with more rules as only the accuracy of the rule list generated, coupled with early stopping, is used to decide on the optimal classifier. The size of the rule list is not taken into consideration. Notably, the *neulonets* generated are small 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. NAC employs an initial population of 10000 individuals throughout the evolution process. However, SNC starts off with 1000 individuals, and incrementally adds individuals as evolution proceeds. This gives rise to a steady linear increase in the population with respect to the training iterations. The increase in population poses a system limitation as the number of ineffective *neulonets* gets larger. Currently, we are experimenting with various weight de-

Table 3. Experimental results depicting predictive errors (percent) of different classifiers.

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)

caying strategies to stave off the weak individuals in the population. Moreover, 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. Much of the time, the evolved *neulonets* can be simplified using a set of simplification rules. For example, a cascade of “Overriding” rules can be simplified to a single “Priority” rule. The careful application of simplification is required as we would like to maintain the diversity of the *neulonet* population.

This paper describes an approach of adapting a library of rudimentary neural networks through evolutionary and minimal weight modifications to facilitate the process of rule discovery and classification. 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. These weights can be expressed using constraints which caters to different types of human-like decision logic. Future work would involve investigation into these “cognitive” constraints and devising more novel neural units.

Acknowledgement

This work is partially supported by NUS/MDA grant R252-000-325-279.

References

- [1] C. Blake and C. Merz. UCI repository of machine learning databases, 1998. University of California, Irvine, Dept. of Information and Computer Sciences.
- [2] H. W. K. Chia, C. L. Tan, and S. Y. Sung. Enhancing knowledge discovery via association-based evolution of neural logic networks. *IEEE Transactions on Knowledge and Data Engineering*, 18(7):889–901, 2006.
- [3] G. M. Edelman. *Neural Darwinism – The Theory of Neuronal Group Selection*. Basic Books, 1987.
- [4] M. S. Gazzaniga. *The Mind’s Past*. University of California Press, 1998.
- [5] J. Horn, D. E. Goldberg, and K. Deb. Implicit Niching in a Learning Classifier System: Nature’s Way. *Evolutionary Computation*, 2(1):37–66, 1994.
- [6] R. Kohavi and M. Sahami. 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, 1996.
- [7] J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, Mass., 1992.
- [8] R. S. Michalski, I. Mozetic, J. Hong, and N. Lavrac. 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, 1986.
- [9] E. Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.
- [10] M. J. Pazzani. Knowledge discovery from data? *IEEE Intelligent Systems*, 15(2):10–13, 2000.
- [11] S. R. Quartz and T. J. Sejnowski. The neural basis of cognitive development: A constructivist manifesto. *Behavioral and Brain Sciences*, 20:537–596, 1997.
- [12] C. L. Tan and H. W. K. Chia. 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., June 2001.
- [13] H. H. Teh. *Neural Logic Network, A New Class Of Neural Networks*. World Scientific, Singapore, 1995.
- [14] S. W. Wilson. Classifier fitness based on accuracy. *Evolutionary Computation*, 3(2):149–175, 1995.
- [15] X. Yao. Evolving artificial neural networks. *PIEEE: Proceedings of the IEEE*, 87, 1999.