Proceedings of the 2001 IEEE Congress on Evolutionary Computation Seoul, Korea • May 27-30, 2001

Learning with the molecular-based hypernetwork model

José L. Segovia-Juárez and Michael Conrad
Department of Computer Science
Wayne State University
Detroit, MI 48202
jls@cs.wayne.edu

Abstract-

The hypernetwork model is a hierarchical architecture that has a representation of the molecular, cellular, and organismic levels of biological organization. Influences flow within each level, and through levels, forming dynamic networks of molecular interactions. With its molecular variation-selection learning algorithm, the hypernetwork is able to solve fairly complex tasks such as the (4-10)-input parity task, and the tic-tac-toe endgame problem, with good results. These performance illustrates the learning capabilities of this model.

1 Introduction

Biological organisms learn how to deal with their environment in order to survive. Learning is understood as an adaptive behavioral change where the organism responds in specific ways to environmental influences. The learning process involves structural and dynamical changes at the molecular, cellular, and organismic levels.

Initial computational models of molecular information processing in the nervous system were proposed by Conrad [2, 3]. A memory mediated learning model, based on manipulation of macromolecular structures [3], describes a complex macro-molecule that will form part of a reference neuron scheme for memory.

The hypernetwork model is an attempt to capture and simulate information processing in biological systems[13]. This approach is based on the percolation network model [4, 5, 7, 8, 9], where environmental influences impinging on the organism will percolate down to lower levels, influencing cell to cell interactions, and finally reaching the molecular and atomic dynamics. Micro-level influences are selectively amplified into higher hierarchical levels until they reach the organism level, where they appear as behavior. Learning is a typical process of this nature in higher organisms.

The primary elements of the hypernetwork model are molecules and their interactions. In the model molecules are represented by binary strings and their shape based interactions are modelled with string matching. Sets of molecules of different types form cells. The organism is built from several cells with cell to cell interactions that are based on receptor and effector molecules. There are no other external features.

The organism changes the molecular components by means of a variation-selection algorithm. The organism is trained to solve two classification tasks: the N-input parity task (N=4, 6, 8, and 10), and the tic-tac-toe endboard problem. The organism learns the tasks by changing its molecular structure, and consequently its molecular interactions. The best molecular structures and interactions will be selected, until the complete table is learned or a termination condition is fulfilled.

The key feature of the model is molecular pattern recognition [6], represented by binary string complementarity matching. This leads to the formation of networks of interactions that have influence on higher hierarchical levels. The influences that flow through the cell are filtered up, by the cell to cell interactions, that in turn change the behavior of the organism.

In the following sections we describe the model in detail, explain the variation-selection algorithm, show the learning results for two problems, and state the conclusions.

2 Model description

The model is a crude representation of a neuronal system. In this section we describe every level (molecular, cellular and organismic) in detail.

2.1 Molecules

The structural unit of the model is a molecule reminiscent of a protein. A molecule is represented by a binary string of up to 42 bits. In a previous design [13] the molecule had just one site with excitatory and catalytic properties. Currently the molecule has three parts of up to 14 bits each (see Figure 1). There are two receptor sites, *excitatory* and *inhibitory*, that put the molecule into the active state or the inactive state, respectively. The third site is called the *catalytic* site, by means of which the molecule will activate neighbor molecules if there is complementarity matching (i.e., matching above a threshold) to the receptor sites of the target molecules. The existence of inhibitory and excitatory sites allows the creation of positive and negative feedback regulatory networks with neighbor molecules.

Molecular types The model has four types of molecules: receptors, effectors, internals, and readouts.

- Receptors are molecules that are sensitive to external influences (from the environment or from molecules from other cells).

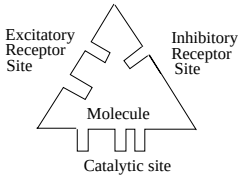

Figure 1: Representation of a molecule with inhibitory, excitatory and catalytic sites.

- Effectors are molecules that will transfer information out of a cell into other cells via their receptors. They behave like neurotransmitters in neural tissues.
- Internals are molecules that, influenced by receptors, will form networks of interactions inside the cell.
- Readouts are molecules that obtain information from the effector molecules of output cells and send information to the external world.

Molecular States Molecules have the following states:

- Ready: When a molecule is waiting to be activated by one of its neighbors.
- Active: When a molecule is activated by one of its neighbors, then for one time step, it can activate its neighbors if there is complementarity matching.
- Inactive: After being activated, the molecule will be inactive for one time step before going back to the *ready* state.
- Delayed: Occurring just in receptor molecules, this is the time delay until the molecule is activated. This is to simulate the timing of the interactions among cells.

The state to state diagram of the internal and effector molecules is shown in Figure 2, and that of receptor molecules is shown in Figure 3.

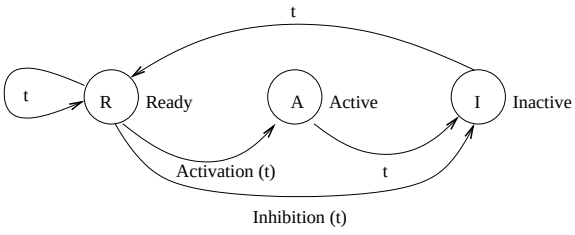

Figure 2: The state to state diagram of the *internal* and *effector* molecules.

Output cells have readout structures that read the state of a particular molecule. If that molecule was activated, then the readout will be in an active state, otherwise it will be in the inactive state. A readout structure reads the state of just one molecule, but this could vary in future experiments.

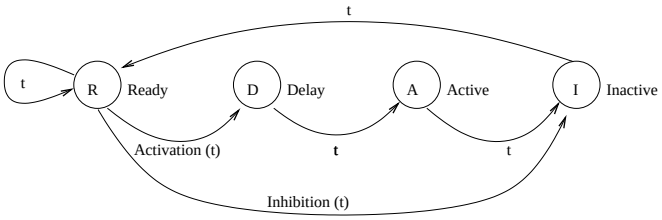

Figure 3: The state to state diagram of the *receptor* molecules.

2.2 Cells

The cell is modelled by a two-dimensional cellular automaton with wrap-around. Every cell has *receptor*, *internal*, and *effector* molecules. The output cells also have *readout* molecules. Molecules are placed randomly in the locations of the grid, and the relationship of neighborhood is shown in Figure 4 where molecule M may have interactions with eight neighbors.

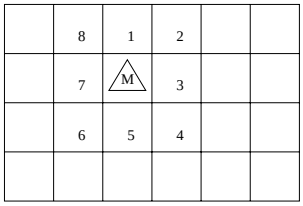

Figure 4: A molecule M, in the cell, interacts with eight neighbors.

2.2.1 Molecular interactions

Molecular interactions form networks of interactions that represent reaction cascades in a cell. Interactions start when the molecule is active. Using its catalytic site, it searches every one of its neighboring molecules for complementarity matching at their excitatory and inhibitory sites. The target molecule will be activated in the next time step if the matching is above a threshold and there is not any inhibitory matching to it. In case there is an inhibitory match, the target molecule will go to the inactive state in the next time step, regardless of any other activation.

The receptor molecules of the input cells are activated for external influences as described below.

2.2.2 Number of cells and molecules

The number of cells and the number of molecules in each cell are initial constants in this version of the model, but this can evolve in future enhancements. Our simulations usually have 25 to 49 molecules for each cell, and from 15 to 36 cells for each organism (see Table 1). In the cell, the number of receptor and effector molecules may vary between four and ten. Output cells have around six readout structures randomly located.

2.2.3 Types of cells

There are three cell types (see Figure 6 as an example):

- Input cells: have receptor molecules that gather influences from the environment.
- Internal cells: do not interact with the external world.
- Output cells: have readout molecules that communicate the state of the system to the environment.

2.3 The organism

A spatially organized group of cells constitutes an organism (see Figure 6). The arrangement is given primarily by the cellular function. Input cells gather influences from the environment, and output cells deliver the global state of the organism to the environment, through the states of their readout molecules. In the experiments the organisms have two layers of internal cells.

In Figure 6 the potential cell to cell interactions are shown with dotted lines. The actual interactions are between the effector and receptor molecules of the respective cells (Figure 5). The cell-cell interactions can change depending on the influences of a particular input.

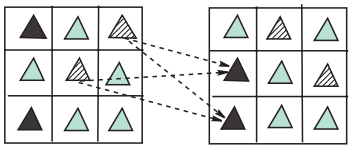

Figure 5: Two cells with their molecules. Receptors are black, effectors are dashed, internal molecules are gray. Dotted lines show the potential effector-receptor interactions.

3 The variation-selection algorithm

The algorithm is designed to learn to classify a set of input vectors. The input and output data are binary vectors. First we evaluate the performance of the organism for all the input vectors. If the organism does not perform well, we reproduce it with mutations. Then, we evaluate the performance of the mutant, as explained below.

3.1 Testing the performance of an organism

The next three steps are performed for every input vector.

- Input of influences into the hypernetwork.

  The input vector is split into small two bit fractions. Each of these will activate an input cell (via activation of its receptors). Molecular receptors have two-bit molecules at their receptor site. Every input cell has at least four different receptor molecules, thereby assuring that at least one of them will be activated by each fraction of the input vector.

- Spread of influences.

  Once the receptor molecules of the input cells are activated, they will activate their neighbors. In this way the influences travel through the cell until they reach effector molecules. Once the effector molecules are active, they will search for receptor molecules of other cells, according to the cell to cell topology. In this way the cell-cell interactions are formed dynamically.

- Generation of output vectors.

  Eventually, influences will activate the effector molecules of the output cells. When this happens the output of the cell is evaluated according to the state of the readout molecules. The global state of each output cell is “1” if there is at least one readout molecule activated, otherwise its global state will be ”0”. The output vector is formed by the concatenation of all the cellular binary states. Then, the distance from this output vector to the desired output vector is measured.

We obtain the performance of the organism from the sum of distances evaluated previously for each input vector.

3.2 Molecular evolution

The organism is reproduced with mutation if its performance was below 100% learning. When reproducing the individual, every molecule has a small probability of mutation (less than 1%). If a molecule is going to change, it does so in a fraction (usually 30%) of its bits, randomly chosen. Every bit has the same chance of keeping its value or flipping.

We test the performance of the mutant. If it has better performance than its parent, we retain it as the better individual, otherwise we mutate the parent again. In case both the parent and the mutant have the same performance, we choose randomly which one to keep as the best organism.

4 Experiments

In this paper we report the performance of the hypernetwork in solving two problems. The first one was the N-input parity problem; the second was the tic-tac-toe endgame problem. The results shown are from a preliminary set of experiments.

4.1 The N-input parity problem

The N-input parity problem is to compute the odd parity of N-binary inputs. The network must output a ”one” if the input has an odd number of ”one” bits, and ”zero” otherwise. Two-input parity (XOR function) is known to be impossible to solve for first order perceptrons [12]. Tesauro and Janssens [15] found that a neural network with back-propagation could learn up to N=8, but it did not converge for N=10. The *NOVEL* method, a global optimization method in a neural network, can learn up to 99.8% in the case of N=10 [14].

Results of the experiments are shown in Table 1. The hypernetwork learns up to 100% in the case of N=4, N=6 and

N=8. For N=10, we got 93.75% correct classification in the best run of four.

N	No. Epochs	Correct %	Organism I(n),I1(n),I2(n),O(n)
4	18,928	100 %	2(25),6(36),6(36),1(49)
	13,665	100 %	2(25),6(36),6(36),1(49)
	3,132	100 %	2(25),6(36),6(36),1(49)
	50,000	75 %	2(25),6(36),6(36),1(49)
	4,767	100%	2(25),6(36),6(36),1(49)
6	17325	100%	3(16),3(20),3(20),1(25)
	18914	100%	3(16),3(20),3(20),1(25)
	12483	100%	3(16),3(20),3(20),1(25)
	11002	100%	3(16),3(20),3(20),1(25)
8	41,659	100%	4(36),6(36),6(36),1(49)
	110,101	100%	4(36),6(36),6(36),1(49)
	130,000	75%	4(36),6(36),6(36),1(36)
	150,000	75%	4(36),6(36),6(36),1(36)
10	146,000	82.04 %	5(64),6(100),6(100),1(100)
	133,700	80.86 %	5(64),6(36),6(36),1(36)
	150,700	81.25 %	5(64),6(36),6(36),1(36)
	161,200	93.75 %	5(64),6(36),6(36),1(36)

Table 1: Preliminary results for the N-input parity problem, N ranging from 4 to 10. One experiment is shown in each row. I=input cells; I1,I2=internal cells; O=output cells. Shown in parentheses is the number of molecules/cell. Running time varies from minutes in the case of N=4, to two weeks in the case of N=10, on a Pentium II 450MHz.

4.2 Tic-tac-toe endgame problem

The hypernetwork was evaluated on the tic-tac-toe endgame problem, which contains 958 possible legal endgame boards [1, 10, 11]. About 65.3% of these instances are positive (i.e., winners for a player "x", assumed to have played first). The task is to learn to classify the endgame board configuration into winners or losers for player "x".

The original data (obtained from www.ics.uci.edu/pub/mlearn/databases/tic-tac-toe/) was given with three values for each cell of the board (player "x", player "o", and blank "b"), and the output can be positive or negative. We transform the value "x" into "01", "o" into "10", and "b" into "00". In order to obtain the input vector we concatenate the board configuration into an input vector of 18 binary digits. We construct the training set with a randomly chosen 70% of the initial 958 vectors, using the rest (30%) as the test set.

Aha [1], with variations of an instance based learning algorithm, obtained performance between 82% and 99%.

The tic-tac-toe experiments were run with two hypernetwork configurations. The first one, called *organism A*, had 9 input cells (each of which reads two bits from the input vector), 10 internal cells in each of the two internal layers, and an output cell (see Figure 6). The second, called *organism B*, had two larger internal cells in each of the second and third internal layers (see Figure 7).

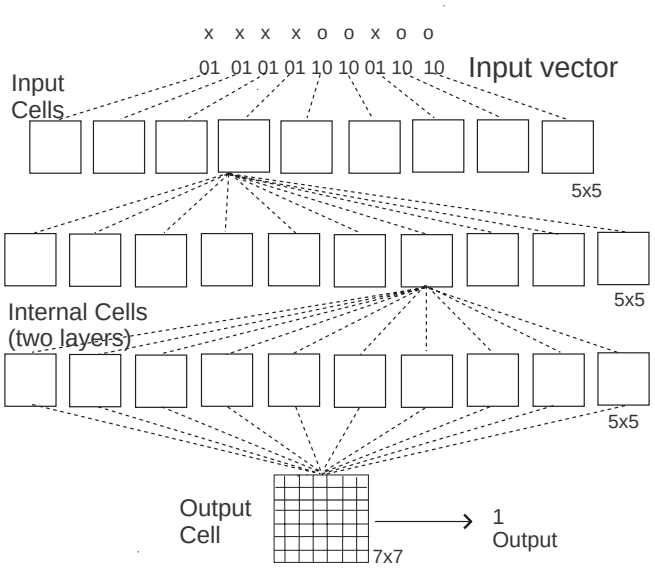

Figure 6: Organism A, with 9 input cells, 10 cells in each of two internal layers, and one output cell. The size of the cells is shown. Only part of the potential cell to cell interactions is shown.

The results of four experiments with organism A is shown in Table 2. The best organism learned 93.23% in the training set, and had 90% of the answers correct in the test set.

Exp. No.	Correct % Training	Correct % Test
1	88.21%	79.57%
2	90.28%	86.03%
3	93.23%	90.32%
4	89.40%	85.34%

Table 2: Results for four "A" organisms. The number of epochs is 150,000

Results with organism B are shown in Table 3. We obtain 94.11% in two runs, with up to 92.11% in the test set. The learning curves of the experiments are show in the Figures 8 and 9.

Exp. No.	Correct % Training	Correct % Test
1	94.11%	91.04%
2	94.11%	92.11%
3	89.25%	84.59%

Table 3: Results for three "B" organisms. Number of epochs is 150,000.

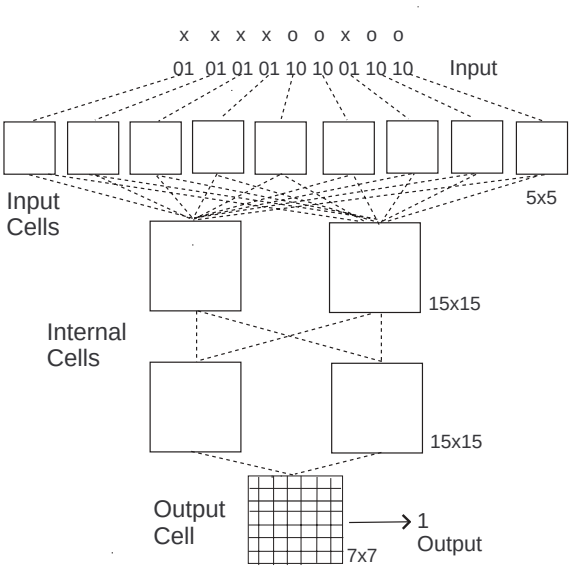

Figure 7: Organism B, with 9 input cells, 2 cells in two internal layers, and one output cell. The size of the cells is shown.

5 Conclusions

We have described in detail the hypernetwork model. It has a hierarchical point of view and is based on molecular interactions. Influences impinging on the organism generate the formation of networks of interactions. The model allows negative feedback regulation at the molecular level. The system evolves by molecular mutations selected by a variation-selection algorithm.

The paper investigates the performance of the hypernetwork for the N-input parity classification task, and the tic-tac-toe endboard problem. We obtain up to 100% and 93.75% learning with the 8-input, and 10-input parity, respectively. With the tic-tac-toe we obtain up to 94% with the training set, and 92% of learning in the testing set.

The results show that learning these fairly complex tasks is possible with a model based solely on molecular interactions in the hypernetwork architecture.

Acknowledgements

This work was supported in part by NSF under grants ECS-9704190, and CCR-9610054, and in part by NASA under grant NCC-2-1189.

Bibliography

[1] Aha, D. H. (1991). Incremental constructive induction: An instance-based approach. In *Proceedings of the Eighth International Workshop on Machine Learning*, pages 117–121. Morgan Kaufmann, Evanston, ILL.

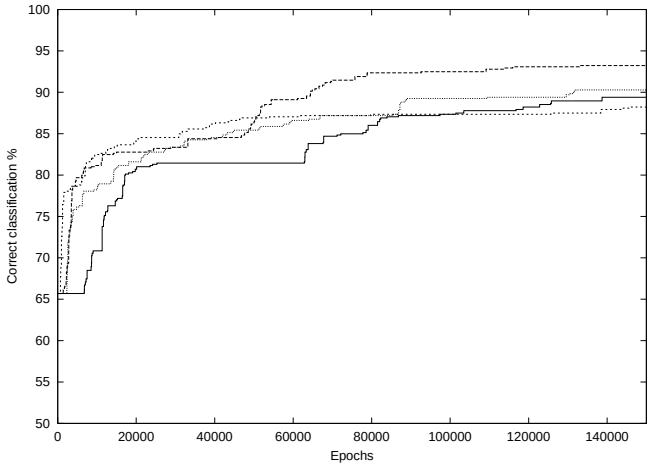

Figure 8: Learning curves for the four "A" organisms from Table 2. Percentage of correct classifications is shown on the vertical axis.

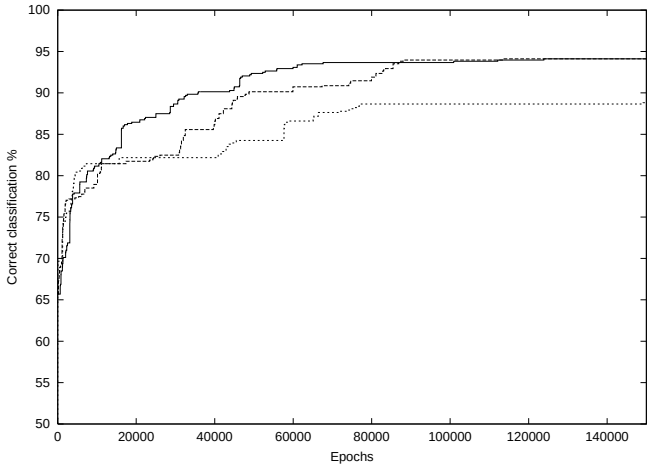

Figure 9: Learning curves for the three "B" organisms from Table 3. Percentage of correct classifications is shown on the vertical axis.

[2] Conrad, M. (1974). Molecular information processing in the central nervous system. Part II: Molecular data structures. In Conrad, M., Güttinger, and Dal Cin, M., editors, *Physics and Mathematics of the Nervous System*, pages 108–127, Springer-Verlag. Berlin-Heildelberg-New York.

[3] Conrad, M. (1976). Molecular information structures in the brain. *Journal of Neuroscience Research*, 2:233–254.

[4] Conrad, M. (1979). Hierarchical adaptability theory and its cross-correlation with dynamical ecological models. In Halfon, E., editor, *Theoretical Systems Ecology*, pages 132–150. Academic Press, New York.

[5] Conrad, M. (1984). Microscopic-macroscopic interface in biological information processing. *Biosystems*, 16:345–363.

[6] Conrad, M. (1992). Molecular computing: The lock-key paradigm. *Computer*, 25(11):11–20.

[7] Conrad, M. (1993). Emergent computation through self-assembly. *Nanobiology*, 2:5–30.

[8] Conrad, M. (1995a). Cross-scale interactions in biomolecular information processing. *BioSystems*, 35:157–160.

[9] Conrad, M. (1995b). Multiscale synergy in biological information processing. *Optical Memory and Neural Networks*, 4(2):89–98.

[10] Matheus, C. J. (1990). Adding domain knowledge to sbl through feature construction. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 803–808, Boston, MA. AAAI Press.

[11] Matheus, C. J. and Rendell, L. A. (1989). Constructive induction on decision trees. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 645–650, Detroit, MI. Morgan Kaufmann.

[12] Minsky, M. and Papert, S. (1969). *Perceptrons: An Introduction to Comptutational Geometry*. MIT Press, Cambridge, MA.

[13] Segovia-Juárez, J. and Conrad, M. (1999). Hypernetwork model of biological information processing. In *Proceedings of the 1999 Congress on Evolutionary Computation*, pages 511–515.

[14] Shang, Y. and Wah, B. W. (1996). Global optimization for neural network training. *IEEE Computer*, 3:45–54.

[15] Tesauro, G. and Janssens, B. (1988). Scaling relationships in back-propagation learning. *Complex Systems*, 2:39–84.