Representing Visual Schemas in Neural Networks for Scene Analysis

Wee Kheng Leow and Risto Miikkulainen
Department of Computer Sciences, University of Texas at Austin,
Austin, Texas 78712, USA
leow@cs.utexas.edu, risto@cs.utexas.edu

Abstract— Using object recognition in simple scenes as the task, this research focuses on two fundamental problems in neural network systems: (1) processing large amounts of input with limited resources, and (2) the representation and use of structured knowledge. The first problem arises because no practical neural network can process all the visual input simultaneously and efficiently. The solution is to process a small amount of the input in parallel, and successively focus on other parts of the input. This strategy requires that the system maintains structured knowledge for describing and interpreting successively gathered information.

The proposed system, VISOR, consists of two main modules. The Low-Level Visual Module (simulated using procedural programs) extracts featural and positional information from the visual input. The Schema Module (implemented with neural networks) encodes structured knowledge about possible objects, and provides top-down information for the Low-Level Visual Module to focus attention at different parts of the scene. Working cooperatively with the Low-Level Visual Module, it builds a globally consistent interpretation of successively gathered visual information.

I. Introduction

Consider the task of recognizing objects in simple scenes. A scene analysis system has to identify the objects in the scene (e.g., an arch and two trees, Fig. 1a) and determine what the scene depicts (e.g., a park). In designing a neural network system that performs this task, we encounter two fundamental problems:

1. How can a fixed, finite neural network process indefinitely large amounts of information?
2. How can a neural network represent and use structured knowledge?

In fact, these problems are also encountered in many other neural network application areas, such as speech understanding and natural language processing. The goal of this research is to develop general solutions to these problems, using scene analysis as a concrete task.

Consider the first problem: limited processing resources. In practice, it is only possible to construct a neural network with a fixed number of input units and internal processing units. The weights and activities have finite precision and are bounded within certain ranges of values. The number of input units may be smaller than the size of the scene (in pixels). Even if the network can capture a large part of the scene at once, it may not be able to process all the information in parallel unless it has an exponential amount of units and connections [1]. The only viable option is to process a small amount of visual input in parallel, and successively focus on different parts of the scene. This strategy also seems to be in use in biological vision systems [2].

Since the network is fixed and finite, it may not have enough storage space for the indefinitely large amounts of input information. It will have to build and maintain a *partial interpretation* of the information gathered so far. Based on gathered information, it estimates the likelihood that the input features belong to some known objects. As more information is received, it strengthens or weakens the tentative estimates. It continues processing other parts of the scene until it has gathered sufficient information to build a consistent interpretation. Each partial interpretation corresponds to an intermediate stable state of the network, and the globally consistent interpretation corresponds to the final stable state.

A system that adopts this strategy requires an internal model, generally known as *schema* in psychological research, for making the interpretations [2]. Thus, the solution to the first problem requires that neural networks encode schemas, or in general, structured knowledge; that is, it requires addressing the second problem. One approach is to represent such knowledge symbolically in neural networks [3, 4, 5]. The approach works in simple cases but does not generalize well to more complex tasks. Neural networks are not very good at manipulating symbols explicitly. However, they are good at feature extraction, association, constraint satisfaction, pattern classification, and making other fuzzy decisions. These tasks are performed through "neural" processes such as cooperation and competition among units and networks.

This paper appeared in the *Proceedings of the International Conference on Neural Networks*, volume III, 1612-1617, 1993.

The VISOR system (VIsual Schemas for Object Representation) is designed to address the two fundamental problems in the domain of object recognition and scene analysis. Simplifications have been made to help focus the research effort on the main issues—the representation and learning of schemas. The scenes considered in this project consist of objects made up of straight lines and simple shapes such as rectangles and triangles. The knowledge that describes objects and scenes involves four positional relationships (left, right, above and below) and one hierarchical relationship (is-part-of). Such knowledge can be conveniently encoded in terms of maps and connections among units. Despite the simplified task, this research aims at deriving general solutions that are applicable to more complex scenes and other tasks.

II. Related Work

Rumelhart et al. [6] suggested a general method for encoding conceptual schemas in a PDP model. Individual components of schemas, such as sofa, bed, bathtub, and toilet are represented as units in a network. The weight of the connection between two units represents how likely the two components are to be present in a schema, and the activity pattern of the network encodes a schema instantiation. The network does not encode hierarchical relationships among the schemas.

Hinton [7] described three methods for representing hierarchical knowledge. The second method is similar to the one used in VISOR. The units in the network are organized into different levels. The higher the level, the more complex is the object that the unit represents. Lower-level units representing components of objects are connected to one or more higher-level units representing the objects themselves.

The cognitive model of Norman and Shallice focuses on the activation and control of schemas [8, 9]. In this model, domain-specific action schemas and thought schemas can be activated independently of each other. A small subset of schemas to be "run" are selected by two distinct processes known as Contention Scheduling and Supervisory Attentional System. Contention Scheduling is a domain-specific process analogous to conflict resolution in traditional AI systems. It selects schemas according to simple criteria that are domain-specific. Supervisory Attentional System is a general planning system that operates on schemas in every domain. It controls the activation of schemas by biasing the operations of Contention Scheduling. The activation and control of schemas in VISOR are analogous to the Contention Scheduling process.

III. The Architecture of VISOR

VISOR is based on the separation of the "what" and "where" pathways in low-level vision ([10]; Fig. 1). It consists of the Low-Level Visual Module (simulated using procedural programs) and the Schema Module (implemented with neural networks). The Low-Level Visual

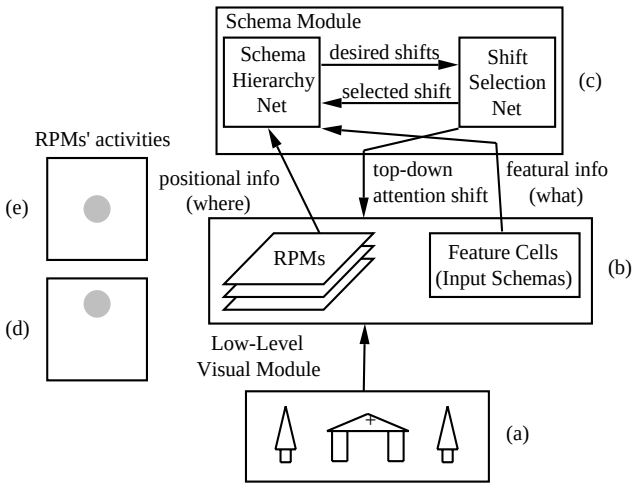


Figure 1: VISOR consists of the Low-Level Visual Module (LLVM, b) and the Schema Module (c). LLVM extracts "what" and "where" information from the scene (a). Figures (d) and (e) indicate the activities of fine-scaled and coarse-scaled Relative Position Maps (RPMs) when attention is focused at the position marked with "+".

Module (LLVM, Fig. 1b) focuses its attention at one position in the scene at a time, and extracts the feature (line, rectangle or triangle) at that location. As its output, the Feature Cells indicate how strongly the LLVM believes a particular feature is present (Fig. 2). The Relative Position Maps (RPMs) encode the relative positions of the features at several scales. For example, suppose that part of the scene contains an arch and two trees (Fig. 1a). Also suppose that attention is currently focused on the triangular roof of the arch. At a fine scale, the RPM identifies the triangle as located above the two rectangles, and gives a peak response at the top part of the map (Fig. 1d). At a coarser scale, the RPM identifies the blob of features constituting the arch as located in the middle of the blobs corresponding to the two trees, and forms a peak response at the center of the map (Fig. 1e). At scales that are larger than that of the retina, the positions of the eyes are taken into account.

The Schema Module (Fig. 1a) maintains the hierarchy of schemas, integrates successive input information, and determines the next position of attention. It consists of two main neural networks: the Schema Hierarchy Net (SHN) and the Shift Selection Net (SSN). The SHN is a multi-layer network of schema representation nets, or schema-nets for short (Fig. 2). A schema-net consists of four main components: the output unit, the Sub-schema Activity Map (SAM), the Current Position Map (CPM), and the Potential Position Map (PPM). Before describing these components in detail, let us first look at how the schema hierarchy is represented in the SHN.

Each layer of schema-nets corresponds to a level in the schema hierarchy. A schema-net can simultaneously be a sub-schema of higher-level schemas and a super-schema

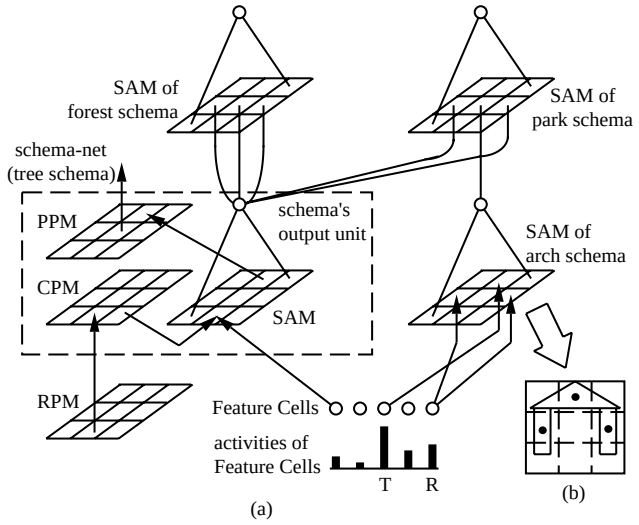


Figure 2: (a) The Schema Hierarchy Net encodes part-whole relationships among the schemas. Arrows represent one-way connections, and solid lines represent both the bottom-up and top-down connections (which are different). For simplicity, the schema-net components are shown only for the tree schema. The Feature Cell marked "T" is sensitive to triangles, and the one marked "R" to rectangles. (b) The arch image encoded by the arch schema. The grid represents the SAM and the black dots denote the positions of the components in the SAM.

of lower-level schemas. The sub-schemas of the first-level schemas consists of the Feature Cells. The connectivity of the SHN encodes the part-whole relationships among the schemas. Consider, for example, the representation of an arch. Fig. 2(b) shows an arch that is made up of three components: a triangular roof and two rectangular pillars. The grid superimposed on the arch represents a map called the Sub-schema Activity Map (SAM) in the arch schema-net. The black dots indicate the positions of the components in the map. For example, the triangle is located at the top-center of the arch map. Corresponding to each black dot, there is a connection from a Feature Cell to a SAM unit. The connection indicates that the feature is a component of the arch schema at the position of the SAM unit.

The SAM units' activities indicate how strongly the sub-schemas are believed to be present in the scene. These activities may change as more information is extracted from the scene. In effect, SAM encodes a summary of current evidence for a schema.

In addition to the dynamic information encoded in SAM, it is necessary to keep information about the static structure of the schema, so that the system can decide where to focus its attention next. Such information is stored in the Potential Position Map (PPM). A high activity in a PPM unit indicates that a sub-schema is expected at the corresponding position.

The current position of attention is stored in the Current Position Map (CPM), coded by the location of a single active unit in the map. Each CPM unit connects multiplicatively to the SAM unit at the corresponding location. If a CPM unit is on, the corresponding SAM unit's activity can be updated. Otherwise, the SAM unit's activity remains unchanged. In other words, only the activities of the sub-schemas that match the current position are propagated upwards.

The certainty, or confidence, that a schema matches the input is summarized in the activity of the schema's output unit (or schema's activity for short). In addition to bottom-up connections from the schema's own SAM units, the output unit receives top-down connections from the super-schemas' SAM units (Fig. 2). If a higher-level schema matches an input object with high confidence, then its sub-schemas are expected to match the object's components as well; hence the top-down feedback. There are also mutually inhibitory connections among the schemas' output units to allow the schemas to compete in interpreting the input. (The detailed schema activation equations are given in the appendix.)

After processing the information at a particular position in the scene, VISOR will shift its attention to a new position. The Shift Selection Net (SSN) determines this position (Fig. 1c). As will be described in more detail in Section IV, it makes its decision based on the schemas' activities and their desired shift vectors.

IV. VISOR Operation

At the beginning of the scene analysis process, all the schemas are reset to their initial states. That is, none of their CPM units is on (no current position of attention), and the activities of their SAM units are 0 (nothing has been found).[1] After each focusing of attention, the Schema Module processes the featural and positional information received from the LLVM in four main stages: (1) setting current positions of attention within schemas, (2) updating schemas' activities, (3) selecting schemas' desired next position of attention, and (4) selecting one of the next positions for actual attention shift. Let us briefly go through the events of one processing cycle:

1. Setting Current Positions. After the LLVM has shifted its attention to the selected location in the scene, the schemas update their current positions of attention. If a schema is not attending to anything, that is, none of its CPM units is on, its current position is set at the peak position in the RPM (Fig. 2). If one of the CPM units is on, the current position is shifted in the direction and by the amount encoded in the shift vector received from the SSN. If the amount of shift goes beyond the spatial extent of the CPM, then the schema is first reset to its initial

[1] The initial activities could be set to any values between 0 and 1. In effect, schemas with higher initial activities would then be expected to match the objects and the scene better than those with lower initial activities; in other words, expectation and bias could be modeled.

state, and then its current position is set at the peak position in the RPM.

2. Schema Activation. At this stage, one of the CPM units is active, and its position indicates the current position of attention. The activity of the SAM unit at the corresponding map position is updated (Appendix). Other SAM units' activities remain unchanged. The activity of the schema's output unit is also updated according to how well the schema matches the input (See the appendix for details). If it matches well, its activity increases as a result of increased SAM activity; otherwise, its activity decreases as a result of mutual inhibition among the schemas. The activity of a schema in turn feeds back to its sub-schemas and boosts their activities. This feedback signal corresponds to top-down expectation: if a schema matches an object well, then its sub-schemas are expected to match the object's components. The activities are updated asynchronously for several cycles until they stabilize.

3. Selection of Desired Next Positions. After the activities have stabilized, each schema chooses a position at which it would like the system to focus its attention. These are the positions where a schema expects to discover features that will contribute to increasing its activity. The schema's selection is based on the following criteria:

1. Select a position where a sub-schema is expected, that is, a position where there is high activity in the PPM unit.
2. Prefer a position that has low SAM activity. For practical (and biological) networks, the activities of the units are finite and bounded within certain ranges of values. Focusing attention at positions with already high SAM activities is not effective in increasing the activity of the schema.
3. Prefer a position close to the current position so that attention shift is minimized.

The selected position is encoded as a shift vector (x-shift, y-shift) and is sent to the SSN.

4. Selection of The Actual Next Position. The SSN receives the desired shift vectors from all schema-nets as its input, and selects one of them to be adopted. It prefers a small shift desired by a highly active schema. This criteria favors the interpretation of the visual input in terms of the best-matched schema while minimizing the amount of attention shift. Finally, the selected shift vector is propagated to all the schemas and to the LLVM.

V. Experimental Results With VISOR

Three experiments on object recognition and scene analysis were performed. The first experiment illustrates the recognition of a perfect instance of an object, the second that of distorted instances, and the third that of a complete scene. All the schemas were handcoded in the SHN.

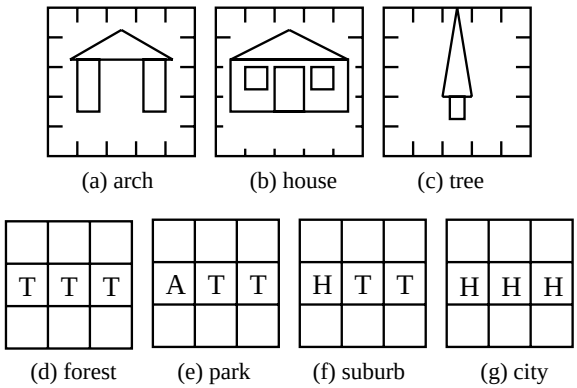


Figure 3: Handcoded schemas used in the experiments. Figures (a)-(c) depict first-level schemas (map size = 5×5), (d)-(g) second-level schemas (map size = 3×3). A = arch, H = house, T = tree.

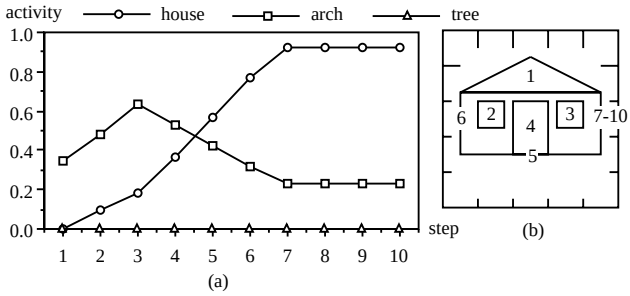
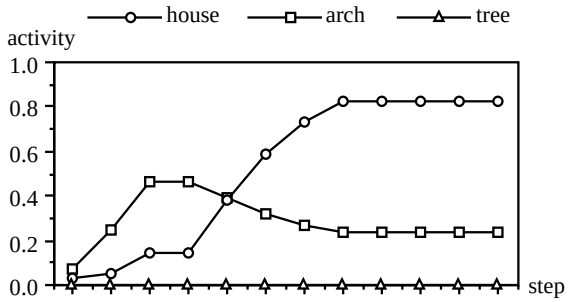


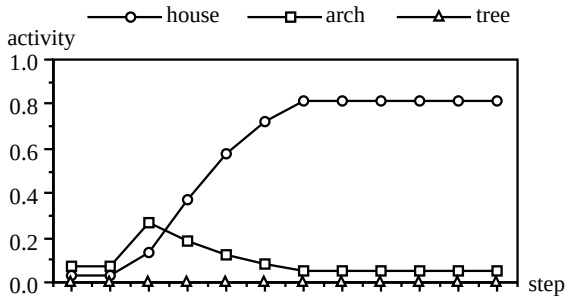
Figure 4: Experimental results of processing a house image. (a) Activities of first-level schemas. (b) The sequence of positions of attention.

The first level of the SHN consisted of the arch, the house, and the tree schemas (Fig. 3a-c). Of these, the arch and the house schemas are very similar. Both have flat triangular roofs, and the rectangular pillars of the arch may be confused with the square windows of the house. The second-level schemas (used in the third experiment) were forest, park, suburb and city (Fig. 3d-g). These schemas are very similar as well. For example, if the scene is either a forest, park or suburb, and VISOR scans from right to left, it will be unable to disambiguate until the object on the far left is identified. Note that these second-level schemas are not intended to be general representations of these scenes. They were conjured up to test the performance of VISOR in highly ambiguous situations.

In the first experiment, a house image was input to VISOR. Fig. 4(a) is a plot of the activities of the first-level schemas as VISOR processes the scene. The positions of attention at each time step are shown in Fig. 4(b). The system was purposedly set to start in an ambiguous state—it focused on the triangular roof of the house. Initially, VISOR thought that the object was most likely an arch. After the fifth step, the activity of the house schema increased and surpassed that of the arch schema. After the eighth step, VISOR reached the final stable state and concluded that the image was (most likely) a house.



(a) House image with flat roof.



(b) House image with no roof.

Figure 5: Activity patterns of schemas with two different house images.

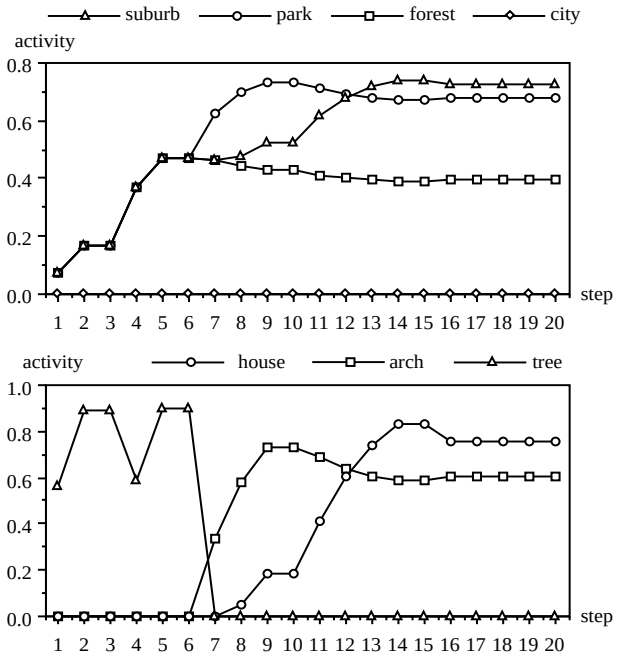


Figure 6: Experimental results of processing a suburb image. The bottom graph shows the activities of the first-level schemas, the top graph those of the second-level schemas.

The second experiment illustrates the processing of distorted images. Two variations of the house image were shown to VISOR. The first had a flat roof, and the second had no roof. In both cases, VISOR started by attending to the left window. Fig. 5 illustrates the schemas' activities for the two cases. The effect of featural distortion is most apparent at the second time step when VISOR was attending to the roof. The more the image differs from that represented in the schema, the lower the activities of the arch and the house schemas. That is, VISOR is less certain about the identity of the object. However, in both cases, VISOR was finally able to conclude that the input object was most likely a house.

In the third experiment, VISOR received a suburb image that exactly matched the suburb schema. VISOR was set to initially attend to the triangle of the rightmost tree (Fig. 3). Note that this state is ambiguous because the forest, park and suburb schemas all have a tree as the rightmost component. At step 2, the rightmost tree was identified (Fig. 6). At step 5, the middle tree was identified. At this time, detailed information of the middle tree was stored in the SAM of the tree schema, but detailed information of the rightmost tree was lost. The previous activity of the tree schema (corresponding to the rightmost tree) was stored only in the SAMs of the second-level schemas. Throughout the first 6 time steps, VISOR was unable to determine whether the input scene was a suburb, a park or a forest. At step 7, VISOR focused attention at the triangular roof of the house. It thought that the object on the far left was most likely an arch, and that the whole image was most likely a park. Final disambiguation occurred at step 13 after attending to the left wall of the house. After this time, the house schema became most active at the first level indicating that the last attended object was a house. Consequently, the suburb schema became the most active second-level schema. Once the activities have stabilized, there is no need to focus attention at other parts of the scene, and the process terminates.

VI. Conclusions

The goal of this research is to develop representation and learning schemes for visual schemas in neural networks. The representation scheme supports integration of successive information so that scene analysis can be accomplished with limited processing resources. The system is implemented simply in terms of maps and cooperative and competitive networks. We are currently working on a method for VISOR to learn schema representations from examples of visual scenes. In a real environment, there can be more than two trees in a park scene, and the arch can be anywhere among the trees. Methods for representing such variations are also currently being investigated.

References

[1] J. K. Tsotsos. How does human vision beat the computational complexity of visual perception? In Z. W. Pylyshyn, editor, *Computational Processes in Human Vision*. Ablex, Norwood, New Jersey, 1988.

[2] J. E. Hochberg. *Perception, 2nd Ed.* Prentice-Hall, Englewood Cliffs, New Jersey, 1978.

[3] D. S. Touretzky. BoltzCONS: Reconciling connectionism with the recursive nature of stacks and trees. In *Proceedings of 8th Annual Conference of Cognitive Science Society*, pages 522–530, 1986.

[4] J. Pollack. Recursive auto-associative memory: Devising compositional distributed representation. In *Proceedings of 10th Annual Conference of Cognitive Science Society*, pages 33–39, 1988.

[5] D. S. Touretzky and G. E. Hinton. A distributed connectionist production system. *Cognitive Science*, 12:423–466, 1988.

[6] David E. Rumelhart, P. Smolensky, James L. McClelland, and Geoffrey E. Hinton. Schemata and sequential thought processings in PDP models. In James L. McClelland and David E. Rumelhart, editors, *Parallel Distributed Processings*. MIT Press, Cambridge, Massachusetts, 1986.

[7] Geoffrey E. Hinton. Representing part-whole hierarchies in connectionist networks. In *Proceedings of 10th Annual Conference of Cognitive Science Society*, pages 48–54, 1988.

[8] D. A. Norman and T. Shallice. Attention to action: Willed and automatic control of behavior. Technical Report 99, Center for Human Information Processing, Univ. of California, San Diego, La Jolla, California, 1980.

[9] T. Shallice. Specific impairments of planning. *Philosophical Transactions of the Royal Society of London B*, 298:199–209, 1982.

[10] David C. Van Essen and C. H. Anderson. Information processing strategies and pathways in the primate retina and visual cortex. In S. F. Zornetzer, J. L. Davis, and C. Lau, editors, *Introduction to Neural and Electronic Networks*. Academic Press, Orlando, Florida, 1990.

Appendix: Schema Activation Equations

Below, the equations governing the activation of units in a schema's Sub-schema Activity Map (SAM), and the activation of the schema's output unit are presented (see also Fig. 2). The following notation is used:

U_i: output unit of schema i
A_i: activity of U_i
u_{ix}: SAM unit of schema i at position x
a_{ix}: activity of u_{ix}
c_{ix}: activity of CPM unit of schema i at position x
W_{ixj}: bottom-up connection weight from U_j to u_{ix}
M_{ixj}: top-down connection weight from u_{ix} to U_j
w_{ix}: feedforward connection weight from u_{ix} to U_i
m_{ix}: feedback connection weight from U_i to u_{ix}
e_{ij}: inhibitory connection weight from U_i to U_j
$\alpha, \beta, \gamma, \delta$: parameters, $0 < \alpha, \beta, \gamma, \delta < 1$

When $c_{ix} = 1$, the SAM units' activities are updated according to

$$a_{ix} = \sum_j W_{ixj} A_j + \alpha m_{ix} A_i \qquad (1)$$

When $c_{ix} = 0$, a_{ix} remains unchanged. The first term $\sum_j W_{ixj} A_j$ sums over all the sub-schemas j of schema i and represents the total bottom-up contribution from the sub-schemas. The second term $m_{ix} A_i$ is the feedback from schema i's output unit to its SAM unit.

The schemas' output activities are updated according to

$$A_i = f(\beta \sum_x w_{ix} a_{ix} + \gamma \sum_{j,y} M_{jyi} a_{jy} - \delta \sum_j e_{ji} A_j) \qquad (2)$$

The activation function $f(z)$ is a sigmoidal function of the form

$$f(z) = \begin{cases} 0 & \text{if } z < 0 \\ z & \text{if } 0 <= z <= 0.9 \\ 1/(1 - e^{-s(z-b)}) & \text{if } z > 0.9 \end{cases} \qquad (3)$$

where $s = -5.493$ and $b = 0.500$. The first term $\sum_x w_{ix} a_{ix}$ sums over all the SAM units u_{ix} of schema i. It is the total feedforward contribution from the SAM. The second term $\sum_{j,y} M_{jyi} a_{jy}$ sums over all the SAM units u_{jy} of all the super-schemas j of schema i. It gives the total top-down contribution from all the super-schemas. The last term $\sum_j e_{ji} A_j$ sums over all the schemas j at the same level as schema i giving the total inhibition from those schemas.