A Hubel Wiesel Model of Early Concept Generalization Based on Local Correlation of Input Features

Sepideh Sadeghi, Kiruthika Ramanathan

***Abstract*— Hubel Wiesel models, successful in visual processing algorithms, have only recently been used in conceptual representation. Despite the biological plausibility of a Hubel-Wiesel like architecture for conceptual memory and encouraging preliminary results, there is no implementation of how inputs at each layer of the hierarchy should be integrated for processing by a given module, based on the correlation of the features. In our paper, we propose the input integration framework – a set of operations performed on the inputs to the learning modules of the Hubel Wiesel model of conceptual memory. These operations weight the modules as being general or specific and therefore determine how modules can be correlated when fed to parents in the higher layers of the hierarchy. Parallels from Psychology are drawn to support our proposed framework. Simulation results on benchmark data show that implementing local correlation corresponds to the process of early concept generalization to reveal the broadest coherent distinctions of conceptual patterns. Finally, we applied the improved model iteratively over two sets of data, which resulted in the generation of finer grained categorizations, similar to progressive differentiation. Based on our results, we conclude that the model can be used to explain how humans intuitively fit a hierarchical representation for any kind of data.**

I. Introduction

CONCEPT representation is one of the primary tasks of human cognition. Categorization and generalization of new concepts are part of concept representation. Computationally, it is assumed that generalization of new concepts is based on their correlation with prior concepts. This leads to categorization judgments that can be used for induction. In recent years, research in computational cognitive science has served to reveal much about the process of concept generalization [1-3].

The idea of feature based concept acquisition has been well studied in psychological literature. Sloutsky [10] discusses how children group concepts based on, not just one, but multiple similarities, which tap the fact that those basic level categories have correlated structures (or features). This correlation of features is also discussed in McClelland and Rogers [1,2] who argue that information should be stored at the individual concept level rather than at the super ordinate category level allowing properties to be shared by many items.

Mountcastle [13], showed that parts of the cortical system are organized in a hierarchy and that some regions are hierarchically above others. David Hubel and Torsten Wiesel showed that the hierarchical architecture consists of neurons in the higher levels of the visual cortex representing more complex features with neurons in the IT representing objects or object parts [14]. Computational Hubel Wiesel models have therefore been developed for object recognition [7, 15] proposing a hierarchy of feature extracting simple (S) and complex (C) cells that allow for positional invariance. The combination of S-cells and C-cells, whose signals propagate up the hierarchy allows for scale and position invariant object recognition.

In a recent work Ramanathan et al [4] have extended Hubel Wiesel models of the visual cortex [5, 6] to model concept representation. The resulting architecture, trained using competitive learning units arranged in a modular, hierarchical fashion, shares some properties with the Parallel Distributed Processing model of semantic cognition [1]. To our knowledge, this is the first implementation of a Hubel Wiesel approach to non-natural medium such as text, and has attempted to model hierarchical representation of keywords to form concepts.

Their model exploits the S and C cell configuration of Hubel Wiesel models by implementing a bottom up, modular, hierarchical structure of concept acquisition and representation, which lays a possible framework for how concepts are represented in the cortex.

However, we observe that there is a gap between this model and Hubel Wiesel models of vision [5, 7]. The model ignores the existence of local correlation between the inputs of neighboring learning modules. In visual Hubel Wiesel models, the input features integrated in the learning modules are locally correlated (neighboring dots composing a small tile of the input image and neighboring tiles of the input image composing a bigger tile). Therefore, in such models, there exists a coherent generalization over small to big parts of the picture from bottom to top of the hierarchy. No local correlation is implemented in Ramanathan et al [4].

Local correlation, the phenomenon where neighboring neurons in the brain process similar information, ensures category coherence while economizing wiring length [8]. In models of vision, where the Hubel Wiesel architecture is widely used, applying local correlation is intuitive, by integrating neighboring tiles of information. If we assume that the brain uses a hierarchical Hubel Wiesel-like architecture to represent concepts, it is important to account

Sepideh Sadeghi (sp.sadeghi@gmail.com) and Kiruthika Ramanathan (kiruthika_r@dsi.a-star.edu.sg) are from the Data Storage Institute, Science and Engineering Research Council, Agency for Science, Technology and Research, A-STAR,

for this local correlation factor.

In this paper, we propose a model for local correlation of inputs, such that neighboring neurons in the Hubel Wiesel model of conceptual memory process similar information. This model, which we call the input integration framework, results in coherent categorizations corresponding to the broadest distinctions in the data, reminiscent of the properties of early concept differentiation. When the input integration framework is integrated with the Hubel Wiesel model, and the model applied iteratively, we observe finer distinctions of categories, similar to progressive differentiation.

We adopt the following terminologies. Given a set of categories generated based on the conceptual (input) features:

1. **Category coherence** [9] refers to the quality of a category being natural, intuitive and useful for inductive inferences. In our model, this is obtained by preserving local correlation of features through the hierarchy.
2. **Concept differentiation** is the ability of the system to distinguish between categories of concepts through the firing of different neurons. It is known that early concept differentiations are broader than later ones and they undergo a continuous change over time [1]. Our model assumes that the changes in the input integration framework of hierarchical memories are one of the sources of progressive differentiation of concepts.

II. Hypothesis

Representative features of a category can be qualitatively regarded as general or specific [1]. General features are more commonly perceived among the members of the category. On the other hand, specific features are only associated with specific members of the category. Therefore, general features are better representatives of a category compared with specific ones. Subsequently, In the process of generalization, general features are weighted over specific features. Sloutsky et al [10] examine the underlying mechanism of early induction (generalization) in light of comparing the role of appearance similarity[1] and kind information[2]. They conclude that early induction is more biased towards the appearance features rather than kind information features. Based on their findings and the details of their experiments, we hypothesize the following:

1. In early generalization, the more frequently perceived prior features are regarded as general.
2. Weighting general features over specific ones (less frequently perceived features) leads to the detection of the broad distinctions of the observed patterns in the domain of the subject's prior knowledge (known features).

A parallel can be drawn between these hypotheses and Sloutsky's work [10] in that the frequently perceived appearance inputs being regarded as general features are weighted over kind information being regarded as specific features, making categorization biased towards appearance similarity.

[1] Visual similarities

[2] Hand coded labeling rules to be used for categorization and induction of hidden attributes of a set of bug like patterns designed by the authors. Use of these labeling rules needed the children to compare the number of fingers with the number of buttons in each pattern. Use of labeling rules were designed to devise a different categorization from the one devised by appearance similarity.

III. System Design and Architecture

A. Architecture

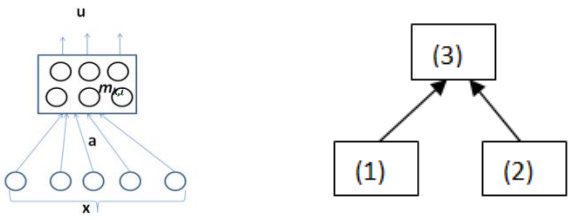
The system that we describe here is organized in a bottom up hierarchy. This means that the conceptual features are represented before the representation of conceptual patterns. Our learning algorithm exploits the property of this hierarchical structure. Each level in the hierarchy has several modules. These modules model cortical regions of concept memory. The modules are arranged in a tree structure, having several children and one parent. In our paper, we call the bottom most level of the hierarchy level 1, and the level increases as one moves up the hierarchy. Each conceptual pattern is defined as a binary vector of conceptual features, where 1 encodes relevance and 0 encodes irrelevance of the corresponding feature to the target pattern. A matrix of all the pattern vectors is directly fed to level 1 as the input. Level 1 modules resemble simple cells of the cortex, in that they receive their inputs from a small patch of the input space. In our model, the input features are distributed amongst the modules at Level 1. Several level 1 modules tile the feature space. A module at level 2 covers more of the feature space when compared to a level 1 module. It represents the union of the feature space of all its child level 1 modules. A level 2 module obtains its inputs only through its level 1 children. This pattern is repeated in the hierarchy. Thus, the module at the tree root (the top most level) covers the entire feature space, but it does so by pooling the inputs from its child modules. In the visual cortex, the level 1 can be considered analogous to the area V1 of the cortex, level 2 to area V2 and so on.

B. Bottom up learning in hierarchical architectures

To understand how the model learns, let us consider the inputs and outputs of a single module $m_{k,i}$ in level k of the system as shown in Figure 1a. Let **x**, representing connections $\{x_j\}$ be the input pattern to the module $m_{k,i}$. **x** is the output of the child modules of $m_{k,i}$ from the level k-1, and **a** represent the weights of the competitive network. The vector **a** is used to represent the connections $\{a_j\}$ between **x** and the neurons in the module $m_{k,i}$ - neuron weight. The output of a neuron in $m_{k,i}$ in response to an input $\{a_j\}$, is 1 if the Euclidean distance between its weight vector and the input is the least compared with other neurons in the module. Otherwise, the output would be zero. The outputs of the neurons being 0 or 1 are called activation values.

During learning, each neuron in $m_{k,i}$ competes with other neurons in the vicinity. Of the large number of inputs to a

given module, a neuron is activated by a subset of them using a winner takes all mechanism. The neuron then becomes the spatial center of these patterns. To ensure that there are no garbage neurons, we adopt in our creation of the module, a model of Growing SOM (GSOM) [11].



(a) (b)

Figure 1a. Inputs and outputs to a single module $m_{k,i.}$ b. The concatenation of information from the child modules of the hierarchy to generate inputs for the parent module

When all the modules at level *k* finish training, the subsequent stage of learning occurs. This comprises the process by which the parent modules learn from the outputs of the child modules. Here, consider the case shown in Figure 1b where module 3 is the parent of modules 1 and 2. Let **x(1)** be the output vector of module 1 and **x(2)** be the output vector of module 2. **x(i)** represents a vector of activation values being the outputs of the neurons in the child modules. The input to module 3, $\mathbf{I(3) = x(1)||x(2)}$, is the concatenation of the outputs of modules 1 and 2. A particular concatenation represents a simultaneous occurrence of a combination of concepts in the child module. Depending on the statistics of the input data, some combinations will occur more frequently, while others will not. During this stage of learning, the parent module learns the most frequent combinations of concepts in the levels below it. A GSOM is again used in the clustering of such combinations. The learning process thus defined can be repeated in a hierarchical manner.

C. *Local correlation model*

We propose a model of local correlation of features which implements our hypothesis in the context of the Hubel Wiesel conceptual memory proposed in Ramanathan et al [4]. Our model accomplishes two tasks through the bottom up hierarchy. (a) It marks the inputs of each layer of the hierarchy as general or specific and (b) It biases the categorization of each module on the basis of its general inputs.

The inputs of the model at the bottom most layer are vectors of conceptual features and at the intermediate layers are vectors of activation values generated by the neurons of the child modules. In order to mark the inputs as general or specific, we first need to weight the generality of each input. In this endeavor, we define two parameters: a) feature weight: to weight the generality of the conceptual features at the bottom layer, and b) module weight: computed for each child module in the hierarchy to weight the generality of its output activation values input to a parent module. In this case all the activation values output from a module would be equally weighted by the computed weight value for the module when being input to the parent module.

In each module, the input vectors and the weight vectors of the neurons are of the same dimension. For each element (feature/activation value) being a member of input vector, there is a weight value which will be incorporated to the model as a coefficient magnifying or trivializing the similarity of the given element in the input vector and the corresponding element in the neuron weight. Therefore feature/module weights are different from the neuron weights that are used for training. However they determine what are the most important elements of the neuron weight vector, which need to be similar to the elements in the input vector for the neuron to be activated.

Let $\mathbf{inputData} = \{\mathbf{P}_1, \mathbf{P}_2, \ldots, \mathbf{P}_n\}$ where $\forall i\colon 1 \le i \le n$, $\mathbf{P}_i$ represents an input pattern such that, $\mathbf{P}_i = (f_{i,1}, f_{i,2}, \ldots, f_{i,m})$ and $\forall j\colon 1 \le j \le m$, $f_{i,j}$ is 1 if the feature is present in $\mathbf{P}_i$ and 0 otherwise. The *Presence Number* N_j for each feature f_j is

$$N_j = \sum_{i=1}^{n} f_{i,j} \qquad (1)$$

The *feature weight* w_j for each input feature and *module weight* w_m for each module within the hierarchy is defined as follows.

$$w_j = \frac{N_j}{n} \qquad (2)$$

We compute w_m for a module as a function of its input weights. Two different operations for computing w_m are presented and compared in our paper – *sum-weights* and *max-weight.*

Where *M* represents the modules at level *p-1* of the hierarchy, the *sum-weights* operation defines w_m at level *p* as

$$w_m = \frac{\sum_{k \in inputs\ of\ m}(w_k)}{\sum_{i \in M} w_i} \qquad (3)$$

Whereas the *max-weight* operation evaluates w_m as

$$w_m = \frac{\max(w_k), k\ \epsilon\ inputs\ of\ m}{\sum_{i \in M} w_i} \qquad (4)$$

The *max-weight* operation is expected to have a greater bias of the results towards general features and broader categorizations than the *sum-weights* operation.

D. *Marking features/ modules as general or specific*

The weight value of each feature or module represents its generality or specificity as seen by the system. In short, the higher the weight value, the more general the feature/module. The pseudo code below is used to label a set of features/modules as being general or specific, depending on the user defined parameters τ (for feature

marking) and μ (for module marking), , where τ and μ are greater than or equal to 1.

Mark features/modules as general_specific()
1. Sort the features/modules in a decreasing order on the basis of their weights and push them into the queue **Q**
*2. while ~(isEmpty(***Q***))*
a. Pop τ/μ features/modules from the front of **Q** *and push them into the queue* **G** *(general features/modules)*
b. pop one feature/module from the rear of **Q** *and push it into the queue* **S** *(specific features/modules)*

E. Generalization

In the process of generalization across the hierarchy, our model weights general features/modules over specific ones by performing two main operations – input management and prioritization.

Input management ensures that the number of general features/modules input to each module of the hierarchy is greater or at least equal to the number of specific features/modules. The following pseudo code explains input management at the most bottom layer of the hierarchy with $\tau=2$. Let $nFeature$ represent the number of features per module. $nSpecific$ encodes the number of available features in the queue **S**, including unused specific features. Biasing the generalization of a pattern towards its general features occurs in the modules of the hierarchy. Therefore, to ensure the relative weight down of the specific features of a pattern to its general features, it is desired to capture both specific and general features of a pattern in the same module. Hence, it is desired to input specific features into a module which shares a pattern with a general feature already added to the module. This is performed by $sharedPattern$ which returns a Boolean indicating whether there is any pattern in which the values of the feature f_j and at least one of the previously added features of the module are one. The performance is dependent on the number of input features/children per module (user defined parameters) and the values of τ and μ.

Input features to the Module(nFeature)
1. if (isOdd(nFeature))
a. pop one feature from the rear of the queue **G** *and push it into the Module*
2. for i=1:floor(nFeature/2)
a. pop one feature from the front of the queue **G** *and push it into the Module*
*b. if ~(isEmpty(***S***))*
i. feature = ***Pop specific(Module)***
ii. push feature into the Module
c. else
i. pop one feature from the rear of the queue **G** *and push it into the Module*

Pop specific(Module)
1.added = 0
2. for i= nSpecific:-1:1
*a. if (sharedPattern(Module,***S**(i)*))*
i. pop **S***(i) from the queue* **S** *and push it into the Module*
ii.added = 1
iii. break
3. if ~(added)
a. pop one feature from the rear of the queue **S** *and push it into the Module*

Prioritization is a weighted similarity measure that interferes in the process of similarity measurement of the conceptual patterns. In our paper, we define the similarity of any two concepts as the Euclidean distance between the representative neurons[3]. The prioritization operation magnifies or trivializes the similarity values of the pair-wise elements in the neuron weights and the input vector on the basis of their corresponding input weights. From equation 5, we can observe that the similarity values of general features with high feature weights would be more significant in the process selection of similar concepts and generalization. In equation 5, $gNum$ and *sNum* represent the number of general and specific features in the module respectively, such that $gNum \geq sNum$. The indices *P* and *C* refer to the pattern (input vector) and the cluster (neuron weight vector).

$$distance(\mathbf{Pattern}, \mathbf{Cluster}) = \sqrt{\sum_{i=1}^{gNum}(f_{i_P} - f_{i_C})^2 \times w_i + \sum_{j=1}^{sNum}(f_{j_P} - f_{j_C})^2 \times w_j} \quad (5)$$

IV. EXPERIMENTS

Every feature in a dataset can divide the pattern space of the data into two separate categories. Our model is to weight the corresponding categories of general features over specific features in the process of categorization. On this basis, two types of data *unique structured* and *multiple structured* can be discussed.

We call a data unique structured if, for every two general features f_i and f_j from the database, where $w_j \leq w_i$, one the following conditions hold. Under first condition, both features should categorize the pattern space similarly. Under second condition, f_j should not divide the pattern space of more than one of the categories created on the basis of f_i. A unique structured data displays a binary hierarchical structure. In contrast, a data which does not fit a binary hierarchical structure or might possibly fit in multiple binary hierarchies would not hold any of conditions above and is regarded as multiple structured data. In the context of this article we call a data unique structured if the top most categorization (focusing on the broadest distinctions) of the patterns is unique in all different hierarchies corresponding to the data. Correspondingly, if different hierarchies of the data demonstrate contradicting categorizations at the top most level, we call the data multiple structured.

Table 1, illustrates three sets of data that have been applied in this paper to test the model. In all the experiments, the parameter 'τ' is equal to 2 and the parameter 'μ' is equal to 1 (each parent module is fed with one general module and one specific module).

[3] As can be seen from equation 5, the effect of prioritization can be observed only when an integration of pair-wise feature similarity is used to measure concept similarity.

Table 1: Datasets used in the simulations

Label	Source	Data type	Remarks
Set A	McClelland et al, 2004	Unique	Whole set
Set B	McClelland et al, 2004	Multiple	Patterns: only animals Features: all – {'has legs'}
Set C	Kemp et al, 2008	Multiple	Whole set

A. *Generalization*

Figure 2 illustrates the contribution of local correlation to the categorization results of the Hubel Wiesel conceptual memory over Set A and Set C. We tested the model under different hierarchical structures, initialized by different number of modules and different number of features per module at the bottom of the hierarchy. As can be seen the local correlation operations, regardless of the structure of the hierarchy and the type of the dataset (unique structured or multiple structured), successfully biases the categorization towards a broad coherent categorization. The resulted categorization over both set A and set C corresponds to the broadest biological distinction of their patterns. The categorization over set A reveals two basic kingdoms of patterns and the categorization over set C reveals two phylums of animals (Arthropods versus ~ Arthropods). Based on our results, when local correlation model is not included, the categorization of data is incoherent and also alternates per runtime.

B. *Categorization operations and computational parameters*

In this section, we compare the categorization performance of the *sum-weights* and *max-weight* operations with respect to the effect of different computational parameters. Growth threshold [11] is a computational parameter used in the learning modules of our model. This parameter controls the growth of the neurons (categories) inside a module by applying a threshold on the distance values of the input patterns and the closest existing neuron weight in the module. If the corresponding distance value for an input pattern is larger than the threshold, a new neuron will be initialized in the modules. Therefore, lower values of growth threshold facilitate the generation of more number of categories and consequently finer distinctions within the corresponding module.

Figures 3 and 4 illustrate the effect of the growth threshold over set A and set C. The specific categories obtained by applying two different correlation operations and various thresholds to the multiple structured set B, is shown in Tables 2 and 3.

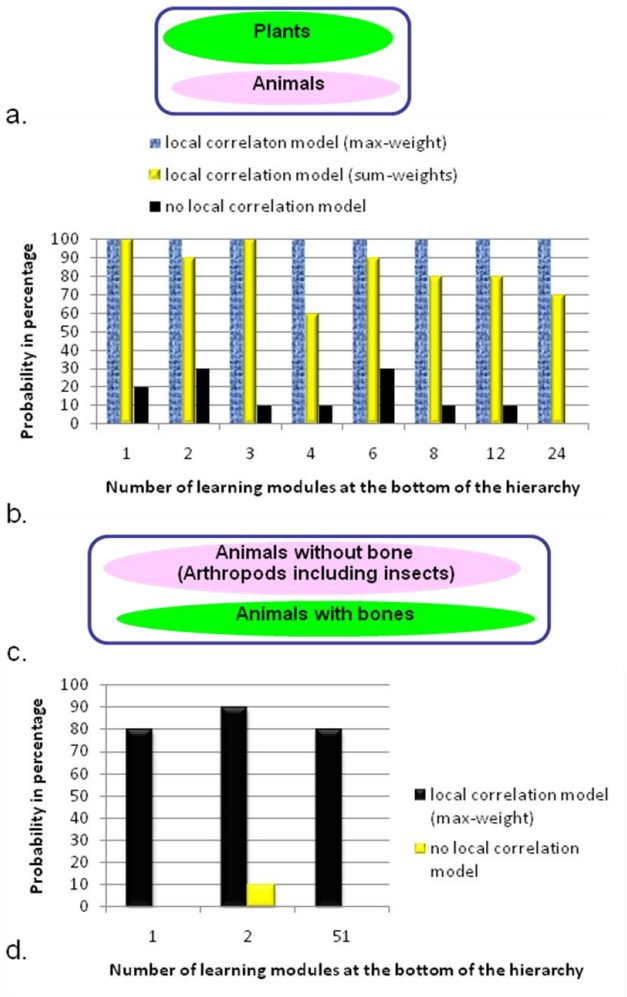


Figure 2: (a) the most frequent outcome categorization of dataset A by local correlation model – successful categorization. (b) Illustrating the probability of successful categorization over set A, being obtained in a set of trials using *sum-weights*, *max-weight* and no correlation model under different hierarchies of learning. Each probability demonstrates the ratio of the number successful categorizations obtained over 10 trials carried out using a specific correlation operation and under specific hierarchy of learning(c) the most frequent categorization of dataset C by local correlation model – successful categorization. (d) Illustrating the probability of successful categorization over set C, being obtained in a set of trials using *sum-weights* and *max-weight* operations under different hierarchies of learning. Each probability is computed in the same way as explained in (b).

According to Figure 2.b, Figure 3, Tables 2 and 3, regardless of the hierarchical structure, type of data and growth threshold values, the *max-weight* operation is always more significant than *sum-weights* in biasing the categorization. As can be seen, this conclusion is admitted by higher probability values reported for *max-weight* dominant categorizations in comparison with those reported for *sum-weights*.

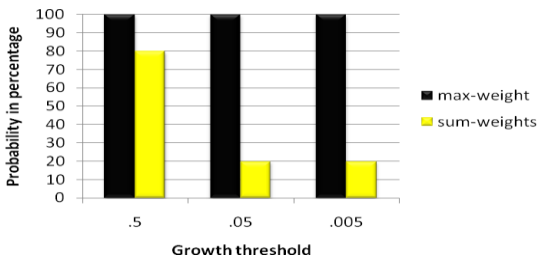


Figure 3: The probability of categorization in Figure 2.a over dataset A. A comparison of *sum-weights* and *max-weight* under different growth thresholds (8 learning modules at the bottom layer)

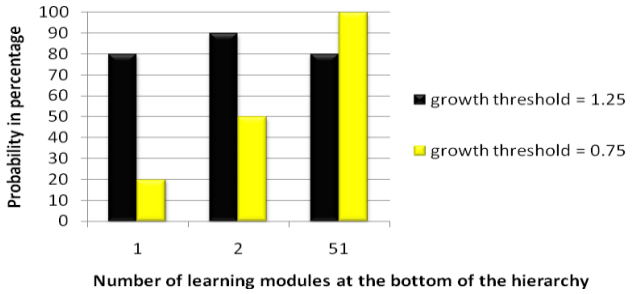


Figure 4: The probability of categorization in Figure 2.c over dataset C. using *max-weight* operation under different growth thresholds in different hierarchical structures.

As can be seen in Figure 3, using *sum-weights* over a unique structured data, probability of getting broad distinctions decreases with the decrease of growth threshold. However, this probability stays robust when using *max-weight* operation. On the other hand, according to Figure 4, applying *max-weight* over a multiple structured data, probability of getting broad distinctions does not stay robust against changes in growth threshold. It is also important to notice that in this case, the probability of getting broad distinctions does not necessarily decrease with the decrease of the growth threshold (Figure 4, 51 modules). This evidence, suggests that the geometry of the hierarchy is another effective factor that along with growth threshold and the structure of data influences the broadness and possibly coherence of the resultant categorization.

According to Tables 2 and 3, using *max-weight* operation over a multiple structured data the dominant categorization gets finer and more coherent (naturally descriptive of data) with the decrease of growth threshold. It is also noticeable that the same effect is not observed using *sum-weights*.

C. *Building hierarchical structures of data*

In this section we use the *max-weight* in a top-down hierarchical manner to build a binary hierarchical structure of the data. Given a dataset, we first apply the model over whole data (the root node of the hierarchy) which results in the creation of several categories. Each of these categories - containing a portion of input patterns and a set of features with variant values among the patterns of the category - is

Table 2: The effect of growth threshold on the quality of categorization biasing, using *max-weight* operation over dataset B (7 modules at the bottom layer).

Growth threshold	The most probable categorization (Dominant Categorization)	Probability of the dominant categorization
0.5	Categorization1	100%
0.05	Categorization 2	60%
0.005	Categorization 3	60%

Categorization 1: (robin, canary, sparrow, sunfish, salmon, flounder, cod), (dog, cat, mouse, goat, pig, penguin)
Categorization 2: (robin, canary, sparrow, sunfish, salmon, flounder, cod), (dog, cat, mouse, goat, pig), (penguin)
Categorization 3: (sunfish, salmon, flounder, cod), (robin, canary, sparrow), (dog, cat, mouse, goat, pig), (penguin)

Table 3: The effect of growth threshold on the quality of categorization biasing, using *sum-weights* operation over dataset B (7 modules at the bottom layer).

Growth threshold	The most probable categorization (Dominant Categorization)	Probability of the dominant categorization
0.5	Categorization4	50%
0.05	Categorization 5	40%
0.005	Categorization 6	40%

Categorization 4: (sunfish, flounder, cod, cat, mouse, penguin, robin, canary, sparrow), (salmon, dog, goat, pig)
Categorization 5: (sunfish, salmon, flounder, cod, dog, cat, mouse, goat, pig), (robin, canary, sparrow, penguin)
Categorization 6: (sunfish, salmon, flounder, cod, dog, cat, mouse, goat, pig), (robin, canary, sparrow, penguin)

regarded as a new dataset (nodes branching from the root node). We apply the model over new datasets (subsets of patterns in a category) iteratively until the desired depth and breadth of the hierarchy in different branches is reached.
The results of applying this procedure over dataset A, and dataset C are provided in Figure 6. It is important to note that changing the growth threshold of the model can change the resulted categorization and the number of categories (the number of branches stemming from the corresponding node). Therefore our model is not only limited to binary hierarchical structures and by changing the growth threshold of the model over a category (node) at level i and branch j of the hierarchy, we can change the emergent hierarchical structure stemming from that particular node.

We assume that humans are capable of performing categorization and subsequently labeling over any given set of patterns represented in the format of feature data [3]. Since, category labels can always be organized into hierarchies [12], therefore regardless of the underlying structural form of the data [3], human mind is considered to be capable of fitting any given feature data into a hierarchical structure. For example, geographical places are naturally organized in a spherical structural form, while human mind is capable of projecting geographical data in hierarchical structure through developing and using concepts

like continent, country, state, and city. In other words, we assume that one of the cognitive properties of human mind is the ability to build hierarchical structure for any given feature data which makes it able to develop abstract but not necessarily natural knowledge about its environment.

Furthermore, it can be discussed that the same set of entities can be represented within different structural forms each of which captures a different aspect of the relationship among the entities. For example, the temporal relationship among seasons, months, and weeks can be captured within cycles while their spatial relationship can be represented in hierarchies (Figure 5). Additionally, different spatial representations of the data within a given structural form reflect different levels of the abstraction of the patterns' relationship. For instance, we can categorize the whole set of animal entities into predators and ~ predators or we can first categorize them into mammals, birds and subsequently categorize each of these categories into predators and ~ predators instances.

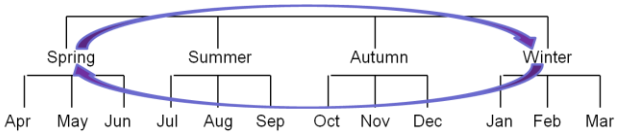


Figure 5: Temporal (cycle) and spatial (hierarchy) relationships of seasons and months

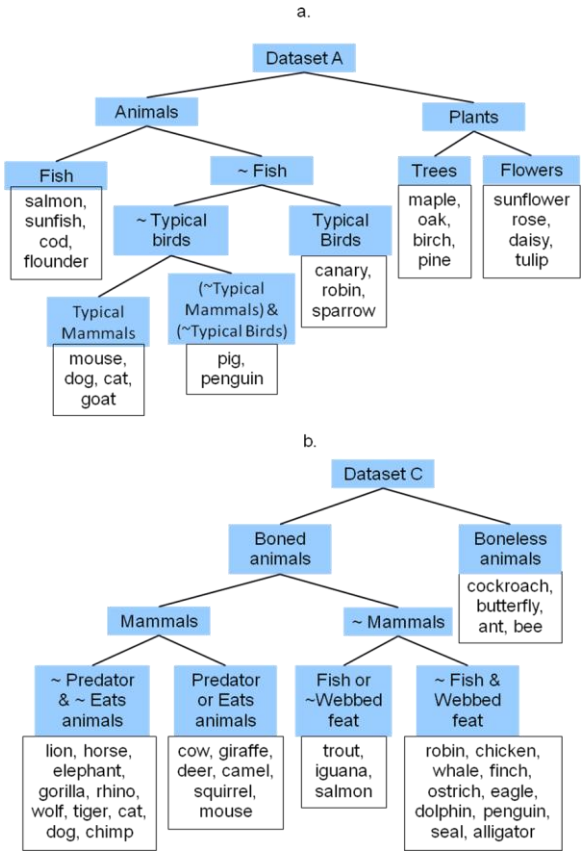


Figure 6: (a) Hierarchical structure of dataset A. (b) Hierarchical structure of dataset C

V. CONCLUSIONS

In summary, we propose an input integration framework for a Hubel Wiesel conceptual memory to bias the generalization process such that it contributes to the categorization of concepts in two ways. First, it increases the probability of obtaining a unique, coherent categorization. . Second, it improves the probability of achieving the broadest distinction (the quality of early concept differentiation due to progressive differentiation phenomena [1]) of the data. Assuming that changes in input integration framework of a hierarchical memory is one of the sources of the progressive differentiation of concepts, further work is in progress to simulate the later developments of the progressive concept differentiation (detection of finer distinctions) on the basis of prior broad distinctions and smooth changes over the input integration framework.

Two operations were designed to perform integration: *max-weight* and *sum-weights.* The potential performance of these operations have been studied and compared under different situations including, different hierarchical structures, different growth thresholds, and different types of input data. These two operations are different in the way they handle the strength of biasing the categorization towards general features. The quality of features being general or specific is subject to continuous change upon receiving new inputs – new features and new patterns. Therefore, it is questionable whether or not *max-weight* operation which gives a very high weight to the detected general features within a single entry might be a brain-like operation. Though, our simulations show that the *max-weight* operation produces more coherent results which are also consistent with the expected broad distinctions perceived in early childhood. May be in early stages of learning, an operation like *max-weight* is used to perceive broad distinctions and build the basic wirings in the brain. While, later a more moderate operation like *sum-weights* is used which does not bias the categorization as strongly as max-weight does.

Our model can be also used to fit any given feature data into a hierarchical structure and provides a possible explanation on how human mind assigns a hierarchical structure to a given data.

REFERENCES

[1] J. L McClelland and T. T Rogers, The parallel distributed processing approach to semantic cognition. *Nature Reviews Neuroscience*, vol. 4, pp310-322, 2003

[2] T. T Rogers and J. L. McClelland Précis of Semantic Cognition, A parallel distributed processing approach. *Brain and Behavioral Sciences*, vol.31, pp689-749, 2008

[3] C Kemp and J. B. Tenenbaum. The discovery of structural form. *Proceedings of the National Academy of Science*, vol. 105 no.31, pp10687-10692, 2008

[4] K Ramanathan et al., A Hubel Wiesel model for hierarchical representation of concepts in textual documents. *The Annual Meeting of the Cognitive Society (COGSCI,)* pp1106-1111, 2010

[5] T. Serre et al., Object recognition with cortex-like mechanisms, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29 no.3, pp411-426, 2007

[6] Reisenhuber M and Poggio T, Hierarchical models of object recognition in cortex, *Nature Neuroscience*, vol. 2 no.11, pp1019-1025, 1999

[7] C. Cadieu et al., A Model of V4 Shape Selectivity and Invariance. Journal of Neurophysiology vol. 98, pp1733-1750, 2007

[8] C Koch, *The Quest for Consciousness, A Neurobiological Approach*, Roberts and Company Publishers, 2004

[9] G. Lmurphy and D.LMedin. The role of theories in conceptual coherence. *Psychological review*, vol. 92, pp289-316, 1985

[10] V.M Sloutsky et al., When looks are everything: appearance similarity versus kind information in early induction. *Psychological Science*, vol. 18 no. 2, pp179-185, 2007

[11] D Alahakhoon et al., Dynamic Self Organizing maps with controlled growth for Knowledge discovery, *IEEE Transactions on neural networks*, vol. 11 no.3, pp601-614, 2000

[12] E Rosch. Principles of categorization. Cognition and Categorization, eds Rosch E, LIoyd BB (Lawrence Erlbaum, New York), pp. 27-48, 1978.

[13] Mountcastle V, *An Organizing Principle for Cerebral Function: The Unit Model and the Distributed System*, The Mindful Brain (Gerald M. Edelman and Vernon B. Mountcastle, eds.) Cambridge, MA: MIT Press, (1978)

[14] Hubel D and Wiesel T, Receptive fields and functional architecture in two non striate visual areas (18 and 19) of a cat, Journal of NeuroPhysiology vol. 28, pp229-289, 1965

[15] K Fukushima, Neocognitron for handwritten digit recognition 1, Neurocomputing , vol. 51 no. C, pp161-180, 2003