Configurational Entropy Stabilizes Pattern Formation in a Hetero-Associative Neural Network

A. J. Maren
Accurate Automation Corporation
1548-B Riverside Drive
Chattanooga, TN 37406
and
Computer Science Department and Brain Research Center
Radford University, Radford, VA 24142

E.Schwartz and J. Seyfried
Computer Science Department and Brain Research Center
Radford University, Radford, VA 24142

Abstract **We report on a prototype implementation and preliminary studies of a new class of computational engine. This engine introduces statistical mechanical considerations into a simple neural network design, affording greater stability in the pattern classes generated in response to different input stimulus. It will exhibit continuous processing, whether or not input is given, which makes it distinct from typical neural networks. The current instantiation of the "engine" consists of two 1-D layers, with feedforward connections between the "input" layer and the "computational" layer. The computational layer achieves its total configuration via response to many factors, including input activations obtained from the input layer, and minimization of a Gibbs Free Energy function. Minimizing the Gibbs Free Energy involves changing the bistate activation of some domains (processing elements) to create an ensemble configuration which balances clustering of like domains with the distribution of different local patterns of domain activation. The clustering stabilizes pattern formation in the computational layer.**

We use a Hamming distance metric to assess the difference between intra-class patterns and inter-class patterns. We find that the inter-class distance between prototype patterns produced in response to different inputs is an order of magnitude greater than the intra-class distance in the computational layer patterns produced in response to a given input.

I. INTRODUCTION

To meet the needs of next-generation intelligent systems, we are developing a new class of computational engine which will exhibit properties different from those of classical neural networks [1]. In particular, this new class of engine will be able to operate continuously, regardless of the presence of an input pattern. The system will not only be able to perform pattern recall, as is true with heteroassociative neural networks, but will also be able to perform free association by activating different stored, stable patterns when the input is turned off. We report here on preliminary results in characterizing the properties of a prototype 1-D implementation of this computational engine, emphasizing its pattern storage and recall abilities.

In current form, the prototype computational engine consists of two layers; an input layer and a computational layer, as shown in Fig. 1. Each of these layers is composed of bistate neural "domains," the basic processing elements of this engine. (We describe our processing elements as "domains," since in later implementations of our system we will give these elements processing abilities which are more complex than those found in simple artificial neurons, and which can be described by the statistical mechanics of ensembles of bistate units.) Each domain in the "input layer" sends weighted values of its bistate (0,1) activation to each domain in the "computational layer." If the sum of these inputs surpasses a threshold, the domain can become "fixed" in the appropriate on or off state as long as the input pattern remains active.

Computational layer domains will receive activations from the input layer, from each other, and from a simulated Gaussian noise source. These domains will also undergo temporal activation decay. The distribution of "active" domains in the "computational layer" of a bilayer hetero-associative network is initially determined by the above factors and is further driven by continuous minimization of an ensemble Gibbs Free Energy function. This combination of processes will afford the engine the ability to move from one stable state to another in a "free association manner," without freezing into a single configuration.

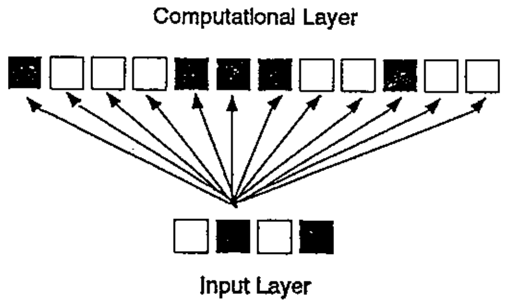


Figure 1: System Architecture for the Computational Engine.

Use of Gibbs Free Energy (GFE) minimization for this engine is analogous to using a Lyapunov function. The GFE is a function defined using statistical mechanics considerations, and by treating the ensemble of domains as a spin glass, which is a common modeling approach in statistical physics. The GFE is not a time-dependent function nor is it a potential energy function in the sense that most Lyapunov functions are. Adopting the language of physicists, we say that a real physical process will move towards equilibrium, which is where the GFE is at a minimum. Thus, during real physical processes, the laws of nature work so as to minimize the GFE, driving the system towards equilibrium. We adopt this as the basis for our computational system, and establish an algorithm which changes the configuration of active domains in the computational layer so as to minimize the GFE for the ensemble of domains.

The Gibbs Free Energy for a system represents the interaction of two competing, driving forces in nature. The first is the tendency of objects to go to their lowest energy state. We describe the energy of the individual elements and their interactions as the "enthalpy" of a system. The competing tendency, called "entropy," is that when a system is composed of elements which can be in more than one state, there is a tendency to distribute the elements evenly among all the possible states. (Our system uses bistate elements). This works against the force driving the system to the lowest energy state. The basic formalism for the Gibbs Free Energy is

$$G = H - TS, \qquad (1)$$

where G is the Gibbs Free Energy, H is the enthalpy, T is the temperature, and S is the entropy. We can express (1) in reduced form, by dividing through by temperature, Boltzmann's constant (k), and the total number of units in the system (N). (Both the latter terms are involved in the expression for entropy). This yields

$$\underline{G} = \underline{H} - S/(Nk), \qquad (2)$$

where $\underline{G}$ and $\underline{H}$ are reduced Gibbs Free Energy and enthalpy, respectively.

The Gibbs Free Energy function we use has a simple interaction enthalpy between neighboring domains, and a complex entropy term involving "cluster variation" variables [2]. This formulation for the entropy term is unusual; it takes into account the distribution of domains into local spatial relationships relative to each other. The variables used to describe these relationships are the nearest-neighbor configurations variables, y_i, and the "triples", z_i, both shown in Fig. 2.

$$\begin{aligned}\underline{G} = G/NkT &= 2\beta e(-z_1+z_3+z_4-z_6) \\ &- 2\sum_{i=1}^{3}\beta_i Lf(y_i) + 2\sum_{i=1}^{6}\gamma_i Lf(z_i) \\ &+ \mu\beta\Big(1-\sum_{i=1}^{6}\gamma_i z_i\Big) + 4\lambda(z_3+z_5-z_2-z_4)\end{aligned} \qquad (3)$$

The first term on the right hand side is the interaction enthalpy, where the z_i are cluster variables for different "triples" of combinations of on/off elements. The second two terms give the entropy, where $Lf(x) = x\ln(x)-x$. The last two terms are Lagrangian multipliers, and the y_i are nearest-neighbor pair cluster variables.

The engine operates by passing weighted input from the input layer to the computational layer. High or low total activations reaching a given domain can "fix" the domain in an on or off state; the activations of the remaining domains are adjusted to minimize Gibbs Free Energy for the ensemble. This is done by randomly selecting an "unfixed" domain in the computational layer, reversing its state, and recalculating the GFE. If the GFE is minimized as a result of the change, the new configuration is kept. If it is increased, the domain state is reversed to its previous state, and a new domain is selected for change.

We increase the system's sensitivity to different input patterns by briefly training the connection weights between the input and the computational layer. These weights are initially randomly distributed in the interval[-1, 1]. We use a modified Hebbian rule to increase (or decrease) their values to increase (or decrease) the activation of each of the computational layer nodes in the direction in which the activation is already tending (positive or negative, respectively).

Configuration	Fraction	α_i
A	x_1	1
B	x_2	1

Configuration	Fraction	β_i
A – A	y_1	1
A – B	y_2	2
B – B	y_3	1

Configuration	Fraction	γ_i
$A\searrow_{A}\swarrow A$	z_1	1
$A\searrow_{A}\swarrow B$	z_2	2
$A\searrow_{B}\swarrow A$	z_3	1
$B\searrow_{A}\swarrow B$	z_4	1
$B\searrow_{B}\swarrow A$	z_5	2
$B\searrow_{B}\swarrow B$	z_6	1

Figure 2: The Fraction Variables for Cluster Variation Theory.

II. EXPERIMENTS

We have performed two sets of experiments on this prototype network. The first set was performed on the computational layer itself, before the input layer was added to the system. The second set of experiments, more pertinent to the pattern recognition abilities of the engine, was performed on the system containing both the input and the computational layers.

For the first set of experiments, we have investigated how well the distribution of cluster variables y_i and z_i that resulted from minimizing the GFE of the computational layer approached the analytically predicted results. We obtained the simulation results with computational layers of 200 or 400 domains (in separate experiments). We established an initial random distribution of activations in the domains. For interaction enthalpies ranging from 0 to 1 in steps of 0.1, and from 1 to 3 in steps of 0.5, we determined the actual fractional values for the different cluster variables. We compared this with the analytically predicted results obtained by taking the derivative of (3) with respect to each of the z_i and setting each of the six equations equal to zero.

This set of six nonlinear equations is solved analytically by invoking distribution constraints ($\Sigma_i \beta_i y_i = 1$; $\Sigma \gamma_i z_i = 1$) and setting the condition that the domains be partitioned equally across the two states; $x_1 = x_2 = 1$). The method is similar to that used in [2]. We found the cluster variable values to be in good agreement with theoretical predictions. This investigation indicated that an interaction strength of about 0.5 was sufficient to cause distinct clustering of like-domains,induced by the interaction enthalpy, without forcing such a great extent of clustering that pattern distinctness would be lost.

Our second set of experiments was oriented towards establishing the patten recall characteristics of our engine, and to identifying the effect of different parameter values. We defined two distance metrics; an "intra-class" distance (to measure distances between patterns in the computational layer induced by the same pattern), and an "inter-class" distance (between prototype versions of the patterns produced in the computational layer in response to different patterns in the input layer). For our experiments, we established an input layer of 20 elements and a computational layer of (typically) 250 elements. Typical experiments were run with 3-5 different input patterns per trial.

For each trial, we determined a "prototype" for each response class. (A "response class" is the set of patterns evoked in response to a given input pattern, when presented at randomly different times.) The prototype for a response class is given as the set of most common bistate activations for the domains in the computational layer, after both initial activations and Gibbs Free Energy minimization have been carried out. The intra-class distance is defined as

$$D_{intra,q} = \frac{1}{NP} \sum_{i=1}^{N} \sum_{p=1}^{P} Abs\ (Activ_{i,p,q} - Proto\_Activ_{i,q}) \quad (4)$$

where $D_{intra,q}$ is the intra-class distance metric pattern q, N is the total number of domains, P is the total number of response pattern occurrences for a specific input pattern, $Activ_{i,p,q}$ is the bistate activation for domain i of response p for pattern q and $Proto\_Active_{i,q}$ is the activation of the ith domain of the prototype response for that input pattern.

Next, we calculate the inter-class distance metric, $D_{inter,j,k}$

$$D_{inter,j,k} = \frac{1}{N} \sum_{i=1}^{N} Abs(Proto\_Activ_{i,j} - Proto\_Activ_{i,k}) \quad (5)$$

where $Proto\_Activ_{i,j}$ is the activation of the ith domain in prototype pattern j, and $Proto\_Activ_{i,k}$ is the activation of the ith domain in prototype k.

III. RESULTS

Typically, intra-class distances are on the order of 0.05, and inter-class distances are about 0.3. The inter-class distance for two randomly-generated patterns would be 0.5. Thus, the prototypes are not as distinct as they would be if they were completely independent of each other. Nevertheless, the distance between prototypes is about an order of magnitude greater than the distance between intra-class patterns.

IV. CONCLUSIONS

We have found that the new computational engine creates distinct patterns in the computational layer when presented with distinct inputs. There is high similarity between the patterns which form in the computational layer in response to different presentations of the same input; the variance is due to the random way in which domains are selected for Gibbs Free Energy minimization. We anticipate that the clustering of like-domains in the computational layer, induced by minimizing the GFE, can lead to greater robustness of pattern recall when input patterns are corroded by noise.

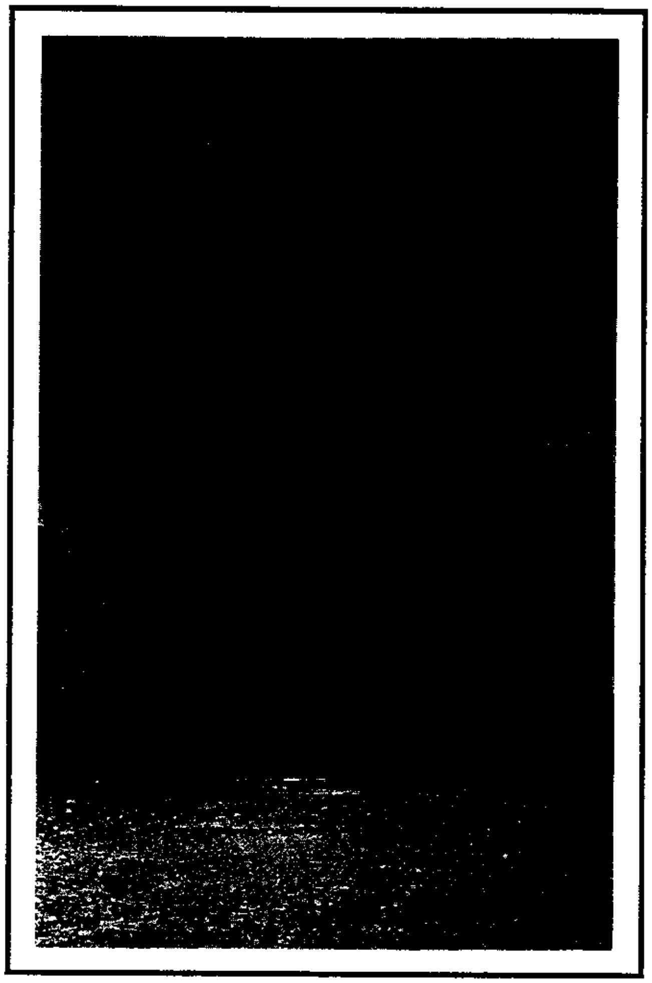
As we refine the system to have temporal dynamics, induced by noise and domain activation decay, we anticipate that GFE minimization acting as ongoing dynamic driver will lead to interesting temporal properties of pattern free association. By creating long-range connection between different domains in the computational layer, we anticipate being able to induce preferential orders recall in temporal activation of associated patterns. This will be a new feature in the design of computational engines, and can lead to applications in modeling complex systems.

REFERENCES

[1] Maren,A.J., Harston,C. and Pap R.M., *Handbook of Neural Computing Applications*, Academic Press, San Diego, 1990.

[2] Kikuchi, R. and Brush,S.G., "Improvement of the cluster-variation method", *J. Chem. Phys.*, vol. 47, 1967, pp 195-203.

1992 IEEE International Conference on Systems, Man, and Cybernetics



Emergent Innovations
in
Information Transfer
Processing
and Decision Making

The Knickerbocker Hotel
Chicago, Illinois

October 18 - 21, 1992

Volume 1 of 2
92CH3176-5

IEEE Systems, Man
and Cybernetics Society