Faster Learning with Overlapping Neural Assemblies

Andrei Kursin[1], Dušan Húsek[2], and Roman Neruda[3]

[1] Kharkiv Polytechnic Institute, Information Systems Department,
21 Frunze st., 61002, Kharkiv, Ukraine
ak@kpi.kharkov.ua
[2] Institute of Computer Science, Neural Networks and Nonlinear Systems Department
2 Pod Vodarenskou veži st., 18207, Prague, Czech Republic
dusan@cs.cas.cz
[3] Institute of Computer Science, Neural Theoretical Computer Science Department
2 Pod Vodarenskou veži st., 18207, Prague, Czech Republic
roman@cs.cas.cz

Abstract. Cell assemblies in neural network are often assumed as overlapping, i.e. a neuron may belong to several of them simultaneously. We argue that network structures with overlapping cell assemblies can exhibit faster learning comparing to non-overlapping ones. In such structures newly trained assemblies take advantage of their overlaps with the already trained neighbors. The assemblies learned in such manner nevertheless preserve the ability for subsequent separate firing. We discuss the implications it may have for intensification of neural network training methods and we also propose to view this learning speed-up in a broader context of inter-assembly cooperation useful for modeling concept formation in human thinking.

1 Introduction

Neural assembly is among the main concepts of connection science. It describes a set of cells which are distinguished from the rest of neuron mass due to their higher connectivity. The chief part in assembly formation is devoted to Hebbian learning process which strengthens the links among neurons that fire simultaneously.

It goes hand in hand with the very definition of neural assembly [1], [2] that the assemblies should be *overlapping*, i.e. one neuron may belong to several of them. It is evident that overlapping structure of assemblies increases storage capacity of a network but on the other hand it rises a problem of organizing connection matrix in a way to avoid palimpsest effects and to ensure separate firing of assemblies in spite of the overlaps. So far the studies of overlapping neural assemblies have concerned primarily this problem [3], [4], [5]. We want to draw the reader's attention to the fact that overlapping assembly structures may have other interesting properties. In particular, we argue here that overlapping assemblies may exhibit faster learning comparing to non-overlapping ones. We describe simulations that support our assumption and, in concluding section, we discuss what consequences such assembly cooperation may have either for neural network training methods or for modeling cognitive functions like concept formation.

S. Kollias et al. (Eds.): ICANN 2006, Part I, LNCS 4131, pp. 226 – 233, 2006.

2 Overview

To verify our assumption we chose to study a simple Hopfield-type neural network [6], [7], [8]. From time to time it is exposed to certain input and when input is released, the network converges to some attractor state. The configuration of the state is determined by the input pattern and the assembly structure of the network. In fact, the network tends to select a neural assembly, which is the closest to the input pattern. Assemblies are distinguished by denser connectivity of their neurons while the network as a whole is sparsely connected. Each assembly may be either trained or not. In the trained state, the most of the links among the assembly neurons are potentiated. An untrained assembly has only initially weak inter-neuron connections.

We studied the process of potentiation of intra-assembly links and found that under certain conditions, the assemblies, overlapping with already trained ones, learn their input patterns faster than assemblies without such overlaps. This can be explained by the fact that portion of their internal linkage is already potentiated. We consider this fact as the simplest form of inter-assembly interactions we hope to simulate later.

3 Network Model

3.1 Neurons

Our network consists of N excitatory neurons and an inhibitory subsystem. Each excitatory neuron *i* (*i*=1,…,N) is a simple 2-state neuron whose activation at time *t* is denoted as $A_i(t)$, $A_i(t) \in \{0, 1\}$. The activation is calculated as a function of the neuron input:

$$A_i(t) = \Theta(I_i(t)), \tag{1}$$

where function Θ returns 1 for positive values of the argument and 0 otherwise.

Neuron input is a sum of the signals incoming from other excitatory neurons, inhibitory signal T_i, and external input H_i if any:

$$\tau_n \dot{I}_i(t) = -I_i + \sum_{j=1}^{k} J_{ij} A_j - T_i + H_i, \tag{2}$$

where J_{ij} is efficacy of a link from neuron *j* to neuron *i, k* is the number of input links per neuron and τ_n is an integration time constant for excitatory neurons.

3.2 Inhibition

The inhibitory subsystem can be viewed as a single giant neuron that takes the sum of all excitatory neuron activations $I = \sum_{i=1}^{N} A_i$ as input and calculates inhibitory signal according to the transduction function

$$\varphi(I) = \begin{cases} \varphi_{\min} & I < I_c \\ \eta I & I \geq I_c \end{cases},$$

where and $\varphi_{\min} > 0$, $\eta \geq 1$ are constant values. The inhibitory signal $T_i = \varphi(I)$ is equal for every excitatory neuron. The goal of the inhibitory subsystem is to keep the number of active excitatory neurons in certain limits. Inhibitory signal is calculated each τ_h time units.

3.3 Input

The network input is described by a set E which comprises all neurons of the network activated at this time. The same value H^{ext} of external input current ensuring high activation probability is injected to all neurons in E while for all other neurons $H_i=0$ in (2). The input set E can be described as follows:

$$E = (A_q \setminus C) \cup B, \qquad (3)$$

where A_q is a set of neurons belonging assemble q while C and B are the sets providing input noise. $C \subset A_q$ contains the neurons from A_q absent in current pattern and B forms "added noise" ($B \cap A_q = \emptyset$).

3.4 Link Modification Rule

As was stated above, our network is sparsely connected. Each neuron receives input links from $k= rN$ neurons, where $r \ll 1$. In the described simulations r was about 10%. A link is characterized by its conductivity J, which can be either modifiable or constant depending on the type of simulation. We make all intra-assembly links constant and potentiated when we test discriminating ability of overlapping assemblies. In learning experiments all links are modifiable.

The links are modified according to Hebbian principle, i.e. a link connecting simultaneously firing neuron is strengthened, and a link connecting a firing and a silent neuron is depressed. We use stochastic link modification rule proposed in [7], [8]. It is notable because it supports either long term potentiation (LTP) or long term depression (LTD) and has analog short term dynamics producing short term memory in attractors. The rule provides robust learning in our experiments though we believe that the main experiment results, i.e. inter-assembly interactions, may be reproduced with other learning rules as well.

A link, according to this model, has two stable conductivity values J_0 and J_1 for LTD and LTP states correspondingly. Suppose, a link is in LTD state and both neurons are active, then current conductivity gradually increases up to threshold w_{ij}, which fluctuates in some boundaries J_0' and J_1', $J_0 < J_0' < J_1' < J_1$. If the threshold is reached, the conductivity jumps to another stable value J_1. If the threshold is not reached, the conductivity quickly returns to the previous stable value in the absence of activation in the connected neurons. Depression of an LTP link occurs reversely. The rule can be formalized as follows:

$$\tau_c \dot{J}_{ij}(t) = -J_{ij}(t) + J_0 + c_{ij}(t) + (J_1 - J_0)\Theta(J_{ij}(t) - w_{ij}(t)). \tag{4}$$

It is an integrator with time constant τ_c. The term $c_{ij}(t)$ represents Hebbian learning source; it is specified in terms of mean activation of the two neurons:

$$c_{ij}(t) = \lambda_+ \overline{A}_i(t)\overline{A}_j(t) - \lambda_-[\overline{A}_i(t) + \overline{A}_j(t)] \ ,$$

where $\overline{A}_i$ is the mean activation of neuron i. λ_+ and λ_- are constant coefficients selected so that transition between the stable states occur only when both values $\overline{A}$ are high or one is high and one is low. w_{ij} is the fluctuating threshold, Θ is the same as in (1) and the whole last term of (4) is the refresh source that indefinitely keeps J_i at one of the stable values in absence of activation. τ_c is taken to be sufficiently long to ensure slow stable learning. Mean activation of neurons is calculated as

$$\overline{A}_i(t) = \frac{\overline{A}_i(t - \tau_n) + A_i(t)}{2} \ .$$

When a neuron is active for several time steps, $\overline{A}_i$ reaches 1. In inactive state $\overline{A}_i$ quickly approaches zero.

3.5 Connection Matrix

One can study overlapping neural assemblies in several ways. For example, it is possible to choose a fully connected network and study how assemblies are formed according to correlations in input data [4], [5]. Or one may arrange a set of "innate" overlapping assemblies and train their connections from the input layer of the network [3]. We chose a structure that would probably help us to test our assumption. It is namely a sparsely connected network where cell assemblies are distinguished by denser connectivity of their neurons. To ensure existence of cell assemblies in such network the connection matrix should be organized according to proximity principle, i.e. when the probability of two neurons to be connected depends on the distance between them. It is about 1 for nearby neurons and gradually comes to zero as the distance increases. We used a square metrics where neurons are located at cross-sections of a square grid. Both dimensions of the network are assumed cycled to avoid undesirable edge effects. The assembly structure in such network is determined according to structured principle formulated in [3]: “The *minint* (minimum internal connectivity) of a set of neurons is the minimum number of innate links that any neuron in the set receives from other neurons in the set. The *maxext* (maximum external connectivity) of a set of neurons is the maximum number of innate links that any neuron outside the set receives from neurons in the set. A web [neural assembly] is a set of neurons whose minint is greater than its maxext.” It was reported that there do exist overlapping assemblies in such networks and their number is usually about the number of neurons in the network [3]. Actually the number is lower if we take into account activation dynamics [2] so that each portion of an assembly can ignite the whole or be necessary for firing of the whole. Otherwise, the set of assemblies, distinguished purely structurally, contains also weakly coupled unions of smaller assemblies.

Actually, reaching maximum number of assemblies was not our purpose here, and we adopted an approximation of proximity principle to simplify control over the network during experiments. We designed our assemblies to occupy certain simple-form geometric areas on network “surface”. Each neuron gets input links from all neurons of the areas to which it belongs, excluding itself. The remainder of its connections is randomly distributed over the rest of the network.

Each arrangement of assemblies achieved in such way was tested first for discrimination properties, i.e. the ability of each assembly to fire independently in the fully trained state. We arranged the connection matrix with fully potentiated intra-assembly links and tested if assemblies can fire independently in response to corresponding data. The matrix versions with insufficient discrimination characteristics did not participate in training tests.

4 Simulations and Results

The simulations were performed with a network of $N = 1024$ neurons. Mean assembly size $M = 16$. The number of input links per neuron $k = 96$. Inhibition parameters were $I_c = 14$ — 16, $\varphi_{\min} = 2$ and $\eta = 3$. $H^{\text{ext}} = 16$. Learning parameters had the following values: $J_0 = 0$, $J_1 = 1$, $J'_0 = 0.4$, $J'_1 = 0.6$, $\lambda_+ = 1.23$ and $\lambda_- = 0.41$. τ_h was taken as a minimum time unit and $\tau_n = 32\tau_h$, which means for discretely calculated network that between two successive updates of inhibitory signal T, activation states of $N/32$ randomly chosen neurons are recalculated.

During simulation, input pattern created as specified above were presented to the network for time period $t_p = 30\tau_n$, then input current was removed and the network was allowed to move to an attractor state during delay period $t_d = 60\tau_n$ whose length was chosen to be sufficient to reach an attractor in any experimental situation. In a fully trained network the attractors usually coincide with corresponding assemblies. During learning tests, the situation is not always the same. The coincidence between the reached attractor and the intended assembly is calculated as a portion of the assembly neurons present in the attractor. This number L was used as a measure of learning. As more intra-assembly links become potentiated, this number approaches 100% and remains close to this in the trained network.

The training sets contained 4-8 assemblies each. Actually, the training set sizes are not of much importance here since assemblies in a set are uncorrelated and learn independently. Sets of any size would be learned in about the same number of cycles.

Training experiments were performed according to the following scheme. The preparation stage started from unlearned connection matrix ($A_{ij} = J_0$ for every i, j). A set X of assemblies in the network were trained using corresponding sequence of input patterns. Each member of the sequence was built on the basis of certain assembly from the set X according to (3). Assemblies in X did not overlap and input patterns were uncorrelated. The sequence was repeated a number of times (S), every time with different random noise portions till L reached 100% for every assembly in the set.

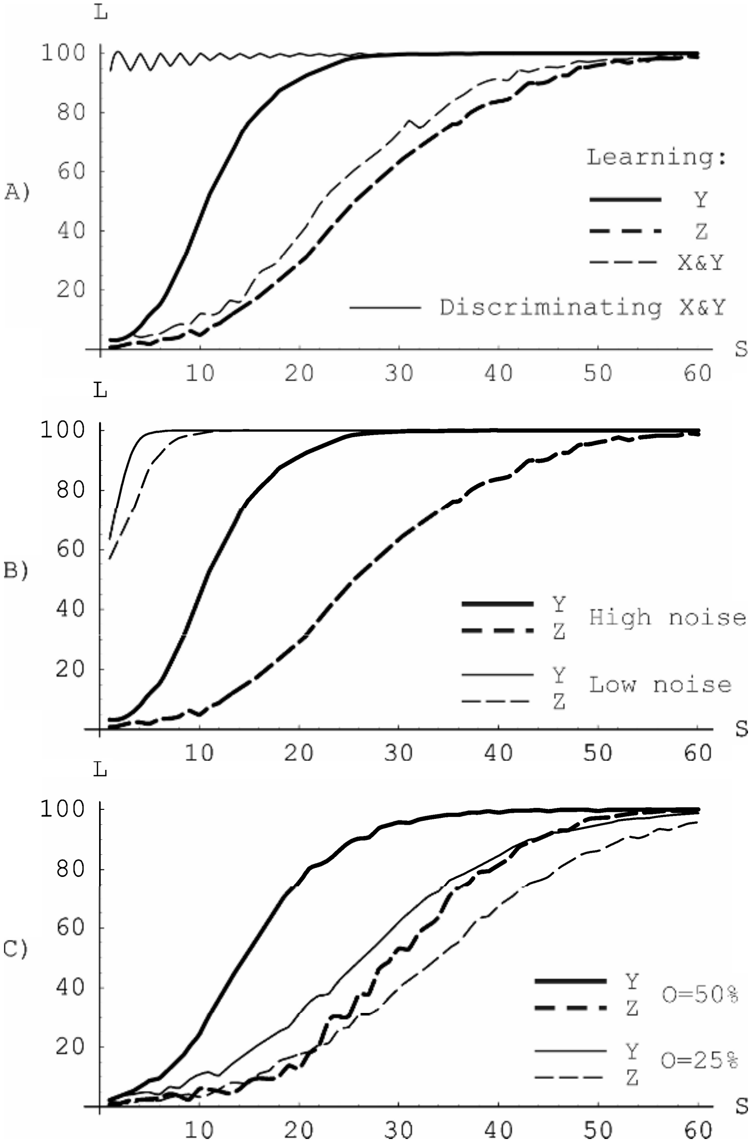

Fig. 1. Assembly-attractor matching measure L (%) versus number of input cycles S. (A) illustrates learning different pattern sets. Thin solid line corresponds to the test for X and Y discriminability after learning both sets. $|C| = 0.0625M$, $|B| = M$, overlap size O = 50%. (B) shows influence of noise: thick lines correspond to $|B| = M$, $|C| = 0.125M$, thin – to $|B| = 0.5M$, $|C| = 0.0625M$; $O = 50\%$. (C) illustrates influence of overlap size; $|C| = 0.25M$, $|B| = M$. The data were averaged over 4–8 assemblies and 50 trials for each test.

Then sets Y and Z of assemblies were trained in the same manner in separate copies of the network obtained on the preparation stage. Members of these sets don't overlap with each other but every assembly in Y overlaps with one or more assemblies in X while assemblies of *Z* don't. It was found that set *Y* is learned faster than *Z*, especially on early stages. The data describing learning progress are presented in Fig. 1.

After the training stage, the networks were tested for sufficient discriminability between sets *X* and *Y* and showed good responses for either set of patterns (the thin solid line in the Fig. 1A).

A copy of each network obtained after the preparation stage was also exposed to an input sequence comprising patterns corresponding to members of both *X* and *Y*. This was done to test a network's ability to learn set *Y* while pertaining memory of set *X*. The learning rates here were slower but usually still higher than the rates for *Z* (the thin dashed line in Fig. 1A). However this test depends much on the ratio between quantities of *X* and *Y* members in the training set. If assemblies from *X* prevail, the Y learning curve after $L = 40$—50% comes close to or even below the curve for set *Z*.

Two factors were found to influence learning advantages of *Y* assemblies. First, they more significantly win in "difficult conditions", i.e. when there is sufficient noise (especially "added noise", $| B | > 0.75\, M$). For illustration see Fig. 1B. We consider it positive since sufficient noise should be expected in real tasks. Tests were performed for $| C |$ between 0 and $0.25M$, and $| B |$ between 0 and M.

Second, the overlap of assemblies should not be so great that it tends to frequently ignite a previously learned *X* assembly, otherwise learning abilities of assemblies in *Y* would be very low. Below this ignition level, increasing size of overlaps facilitates learning of set *Y*. Fig. 1C presents test results for overlap sizes (measured as a portion of M) O=25% and O=50%.

5 Discussion

Faster learning rates for the sets of overlapping assemblies observed in the experiments clearly follow from the ability of such assemblies to benefit from the potentiated links that they already have in their structure due to the overlaps with trained assemblies. At the same time such overlaps don't slow down learning much when the network is exposed to the sequences containing either old or new input patterns. This may be only one from the range of interesting properties that overlapping assembly structures may exhibit. But we want to draw your attention to certain consequences the results may have.

In other publications [9, 10] we argue that neural networks of certain architecture are capable of advanced type of learning resembling metaphorizing abilities of human thinking [11], i.e. when a novel concept is formed by analogy and on the basis of some already known concept. Such phenomenon is ubiquitous in human mental practice. Its cognitive gain is evident: it provides faster and better learning than purely inductive method since learning occurs not through exploration of novel concept from scratch but rather through looking for properties of an old concept in the new one.

In terms of neural networks, such learning is characteristic for the process in which an old trained assembly "helps" a new one to learn a novel piece of input data. This help can be provided only in two ways: through overlapping of neuron sets belonging to the assemblies, or via associative links between the assemblies. It seems that the former factor is more powerful since associative links of previously dormant neurons belonging to the freshly recruited assembly are probably too weak to ensure such "help". In this paper we showed that the "help" through assembly overlaps exists.

On the other hand this research has interesting implications for the field of artificial neural network training. Usage of previously obtained knowledge (when learning a new thing benefits from already knowing something similar) is clearly an effective way for intensification of training methods. This evident feature of natural neural networks is still not sufficiently accounted for in training their artificial analogs. We showed here that neural networks with overlapping assemblies have necessary properties to introduce this feature into learning. Further research may consist in application of the proposed principle to some practical task, e.g. categorization or information retrieval, like in [12].

Acknowledgements

This work is supported by INTAS grant Nr. 03-55-1661 and partly by the Institutional Research Plan AVOZ10300504 "Computer Science for the Information Society: Models, Algorithms, Applications" and grant IET100300414.

References

1. Hebb, D. O.: The Organization of Behavior. A Neuropsychological Theory. New York: John Wiley (1949).
2. Palm, G.: Neural Assemblies. Studies of Brain Function. Vol. VII. Springer, Berlin Heidelberg New York (1982).
3. Wickelgren, W.A.: Webs, Cell Assemblies, and Chunking in Neural Nets. In: Canadian Journal of Experimental psychology. Vol. 53. 1 (1999) 118-131
4. Strangman, G.: Detecting Synchronous Cell Assemblies with Limited Data and Overlapping Assemblies. In: Neural Computation. Vol. 9. (1997) 51-76
5. Huyk, C. R.: Overlapping Cell Assemblies from Correlators. In: Neurocomputing. Vol. 56. (2004) 435-439
6. Hopfield, J. Neural Nets and Physical Systems with Emergent Collective Computational Abilities. Proc. of the Nat. Academy of Sciences USA 79 (1982) 2554-2558
7. Amit, D. J., Brunel, N.: Learning Internal Representations in an Attractor Neural Network. In: Network, 6 (1995) 359-388.
8. Brunel, N.: Hebbian Learning of Context in Recurrent Neural Networks. In: Neural Computation, Vol. 8 (1996) 1677-1710
9. Kursin, A.: Neural Network: Input Anticipation May Lead To Advanced Adaptation Properties. In: Kaynak, O. et al. (eds.): Artificial Neural Networks and Neural Information Processing. Springer-Verlag, Berlin-Heidelberg, 779-785 (2003).
10. Kursin, A.: Self-Organization of Anticipatory Neural Network. In Scientific Proceedings Of Riga Technical University, series Computer Science, Information Technology and Management Science, Riga (2004) 51-59.
11. Lakoff, G., Johnson, M.: Metaphors We Live By. Univ. of Chicago Press, Chicago (1980).
12. Huyck, C., Orengo V.: Information Retrieval and Categorisation using a Cell Assembly Network. In: Neural Computing & Applications. Vol. 14. 4 (2005) 282-289.