УДК 004.392, 004.93'1

K.A. Kermani, F. Hamker, V.G. Spitsyn

MAXIMIZING COMPETITION IN A BIOLOGICALLY PLAUSIBLE NEURAL NETWORK

The goal of this work is increasing competition among the neurons of a one layer feed-forward neural network. The feed-forward weights from input to the cells are learnt using a Hebbian based learning rule and the weights between the neurons are learnt with an anti-Hebbian rule. A dynamic learning rate is introduced and it is claimed that this rule improves the competition.

Keywords: *neural networks*, *Hebbian learning*, *anti-Hebbian learning*, *Oja rule*, *covariance learning*, *competition.*

In order to study the black box of the brain it is necessary to have a model simulating its functionality. The visual system, occupying a big part of the brain and a considerable part of the brain involved in processing of facial images [1] are good candidates for studying the brain. The receptive fields [2] in the visual cortex seems to be result of long term unsupervised learning during the evolution of animals and human. This unsupervised learning has some difficulties to be simulated; beginning with the primary visual cortex, the first problem encounters is that cells receiving the same input signals from the retina and then Lateral Geniculate Nucleus (LGN) should learn different pattern from this unique input. This is what is also called efficient coding in brain [3]. Meaning that brain learns and memorizes the information with the lowest redundancy and with as less resources as possible. Having several cells doing exactly the same function would be meaningless and waste of resources. This may remind one, of principle or independent component analysis in which we try to find functions or components describing the data with low redundancy and at the same time reduce the dimensionality. The goal of this work is enhancing competition among the neurons of a biologically plausible neural network and learning maximum possible number of components from the same input signal. The learning of features is based on the Hebb's method [4] and the main job in competition is done by lateral weights among the neurons of the output layer of the neural network which are learnt with an anti-Hebbian [5] learning rule.

1. Learning rule

1.1. Long term potentiating and long term depression

In this paper we introduce an improvement in the learning rule originally used in [3] which learns the Gabor like filters [6],[7] from natural images and was inspired from [4],[8] and Covariance [2],[9] learning methods:

$$\frac{\tau(\partial W_{ij})}{\partial t} = (u_i - thr_i)^+[(v_j - thr_j) - \alpha(v_j - thr_j)^+ W_{ij}],$$

$$thr_j = mean(v_j;\ j = 1...n)(n:\ \text{number of output cells}),$$

$$thr_i = mean(u_i;\ i = 1...m)(m:\ \text{number of input cells}).$$

Here the + means positive rectification, u_i is the value of the i'th input (i'th pre-synaptic cell's activity), v_j is the activity of the j'th output(i'th post-synaptic cell's activity), τ is the time constant for learning (reverse of learning rate) and α is the weight reduction rate which prevents the constant weight increase. Here W_{ij} is the weight between the neuron i and j.

In the current work the learning process is divided in two parts: Long Term Potentiating (LTP) and Long Term Depression (LTD) [10]. This way the periods when the cells have meaningful activity based on an appropriate input signal and when the cells are facing noise are separated. In the LTP part, usually the weights are increased and in the LTD part are decreased. The basic structure for the learning rule can be defined as:

$$thr_i = Tmax_i ,$$

$$\text{if } v_i > thr_i \text{ then LTP; else LTD,}$$

where $Tmax_i$ is the temporal maximum of the activity of the cell number *i* and thr_i is the threshold for switching between LTP and LTD. The mean activity is not subtracted from the activity as here we are using the temporal maximum activity of the cell as the threshold between LTP and LTD and the subtraction of the temporal maximum from the activity will never result in a positive value and thus there would be no LTP. In contrast the sliding maximum is used as the criteria for switching between LTP and LTD:

$$\text{LTP}: \ \tau_w \frac{dw_{ij}}{dt} = [(u_i - \bar{u})(v_j) - \alpha_j (v_j)^2 w_{ij}],$$

$$\text{LTD}: \ \tau_w \frac{dw_{ij}}{dt} = -[(u_i - \bar{u})(v_j) - \alpha (v_j)^2 \ w_{ij}].$$

In order to prevent appearance of negative weights as a result of weight reduction we push up the weights after each update (1)

$$w_{ij} = w_{ij} + (-\min(w_i))^+ . \tag{1}$$

1.2. Dynamic α

As was introduced in [8], α is used to prevent the weights from constant increase. Based on an idea introduced by [11] a dynamic α was used which lets the weights to freely increase when the neural activity is too low and decreases them when the activity is too high. In other words, the α constrain is regulated by the neural activity and regulates the weight changes. A faster increase in α prevents the networks from having too fast weight increase in the cases a neuron faces a familiar input pattern or when the input itself is strong (2)

$$AlphaInc = (v_i - thresh)^2 - (-(v_i - thresh)^3)^+ , \tag{2}$$

$$AlphaInc = AlphaInc^+ ,$$

$$\tau \frac{\partial \alpha_i}{\partial t} = (AlphaInc - AlphaDec) \ ,$$

$$\tau = 1000; \ \partial t = 1; \ thresh = 0.4 .$$

2. Activation and completion

2.1. Activation

The membrane potential is slowly following a nonlinear function (equation (3)) of the input signals multiplied by the synaptic weights. At the same time the inhibition (h_i) from the other neurons is considered to decrease the neural activity in the case of correlation with other neurons:

$$\tau_p \frac{\partial p_i}{\partial t} = ((uSum_i) - h_i - v_i), \tag{3}$$

$$\tau_p = 10,\ uSum_i = \sum_{j=1}^{m} w_{ij} u_j,$$

$$\text{if } (uSum_i > 1) \text{ then } uSum_i = 0.5 + \frac{1}{(1 + \exp(-3.5(uSum_i - 1)))}.$$

As negative neural activity is not allowed, the neuron's output activity is actually the half-rectification of the membrane potential:

$$v_i = p_i^{+}.$$

The inhibition from other neurons to each neuron is calculated by multiplying the activities of the other neurons to the inhibitory weights and summing the result up:

$$h_i = \sum_{j=1}^{n} hc_i * Wa_{ij} v_j \ ,$$

where

$$\tau_h \ \frac{(\partial hc_i)}{\partial t} = p_i, \tag{4}$$

$$\tau_h = 100,$$

$$\text{if } (mean_i > 0.1) \text{ then } 10000 \ \frac{\partial hc_i}{\partial t} = 0.7 - mean_i, \tag{5}$$

$$\text{else } 100000 \ \frac{\partial hc_i}{\partial t} = mean_i - 0.1,$$

$$hc_i = hc_i^{+},$$

where Wa_{ij} is the lateral weight between the two neurons *i* and *j* of the output layer.

The equations (4) and (5) take care of the strength of the inhibition not to get bigger than the activity of the neuron itself. Without this, the activity of the neuron will be suppressed, the dynamic α will not reduce the weights and as a result the weights will constantly rise. As it will be mentioned later (6), the learning rate slows down when the neural activity increases. This, together with the equations (4) and (5) give also stability to the network.

2.2. Anti-Hebbian learning

The inhibitory weights are learnt using an anti-Hebbian learning rule, meaning that the weights change the same way as in the Hebbian learning, but at the end, the activity is calculated by considering the negative weights. Following is the update rule for the

lateral anti-Hebbian connections between the neurons of the output layer:

$$\tau_h \frac{\partial Wa_{ij}}{\partial t} = (v_i - \gamma)^+ (v_j - \bar{v}) - \alpha_c (v_i - \gamma)^2 Wa_{ij},$$

$$\gamma = 0.3;\ \alpha_c = 0.1.$$

Here γ=0.3 guarantees that just the neurons with an activity higher that 0.3 are considered for competition. This prevents noise to effect the competition. $\bar{v}$ is the mean of the activities of all neurons.

2.3. Temporal mean and standard deviation

The sliding characteristics of the network are used to automatically regulate its function. The sliding standard deviation is simplified to the mean distance from the mean activity. The sliding maximum is calculated by adding the sliding standard deviation to the temporal mean. The mean itself is slowly following the neural activity by time constant τ_t :

$$\tau_t \frac{\partial mean_i}{\partial t} = v_i - mean_i,$$

$$\tau_t \frac{\partial SD(r_i)}{\partial t} = | v_i - mean_i | - SD(v_i),$$

$$Tmax_i = \max(r_i, mean_i + SD(v_i)).$$

2.4. Dynamic time window for temporal mean and standard deviation

An important role in completion among neurons is played by dynamically changing the learning rates τ_l and τ_h and the time constant for following the temporal (sliding) activities τ_t :

$$Tmax_i = \max(r_i, mean_i + SD(r_i)),$$

$$\tau_l = ((Tmax_i)^+)^3 5000 + base, \qquad (6)$$

$$\tau_h = \tau_l \cdot 0.1,$$

$$\tau_t = ((Tmax_i)^+)^2 + b,$$

$$b = 2.$$

The learning time constant is a combination of a linear part; «base» and a strongly non-linear part both rising by the increase in the temporal maximum of neural activities which prevent fast growth in weights when the temporal maximum of the activity is high. Two different «bases» are discussed in Results.

The fact that the learning rate is decreased when the neuron is off for long time, gives the neurons the ability to learn their initial values directly from the input data. The fast learning rate for the lateral (inhibitory) connections $(\tau_h = \tau_l \times 0.1)$ prevents neurons to save similar patterns for a long time as the strong inhibition which is also guided by hc_i de-correlates the neurons in the very early stages of learning.

3. Evaluation criteria

The performance of leaning was measured by various criteria. Two main goals were a high sparseness on the output and a high discrimination ability between different patches.

3.1. Sparseness

One of the characteristics of natural neural networks is their ability in effectively coding information. That is, the combination of activation of just a few cells is used to code amount of information. The fewer number of cells code the information, the more efficiency the network has. That is, high sparseness of the active cells on the output can be a criterion for the efficiency of the network. We have used the sparseness measurement introduced by [12]:

$$S(r) = \frac{\sqrt{n} - (\sum_i |v_i|) / \sqrt{\sum_i v_i^2}}{\sqrt{n} - 1},$$

where n is the number of dimensions. The idea here is that if a vector has a value just in one of its components, the length of the vector and the value it has in the mentioned dimension are the same and their division is one. So in the case in a vector we have just one value, $S(r)$ is one, and in other cases this value is less than one as:

$$x + y + ... > \sqrt{x^2 + y^2 + ...}$$

3.2. Similarity

We considered the cosine of angle between norms of two vectors as a criterion for similarity [3]. This value is one for parallel or similar vectors. The more two image vectors are different; the similarity factor for them is more far from 1 and closer to zero. For measuring the similarity, we divided the dot product of two vectors by the multiplication of their lengths:

$$Similarity(A, B) = A.B / |A||B| = \cos(\theta),$$

$$A.B = |A||B| \cos(\theta).$$

3.2.1. Similarity as a tool for counting learned patterns

This similarity was used as preliminary criteria for the efficiency of learning. The weight vectors of the neurons were compared one by one and the ones with high similarities were grouped together. The number of groups was considered as the number of different patterns the network has learnt.

3.3. Independence

In order to measure the independence, we used «variance of the conditional distributions» introduced in [3]. The idea was that if the variance of the output of one neuron changes depending on the output of the other neuron, these two neurons are not independent. We defined 9 ranges of activities beginning with the first range from 0 to 0.1 of the maximum neural activity and the 9th range between the 0.9 of the maximum activity and the maximum activity. Then supposing the activity of one cell being in one of the ranges, calculated the variance of the activity of another arbitrary neuron. If this variance does not change (has the variance of zero) in all other ranges, then the two se-

lected neurons are independent. We called the mean of the variances of variance of the neural activities in different rages, «Dependency». The algorithm is as follows:

for each two v_c and v_k cell activities $(c < k)$

{

for i = 1 to Nranges

{

for j = 1 to Nranges

{

$lb_i = i$*range;

$ub_i = (i+1)$* range;

$ld_j = j$*range;

$ud_j = (j+1)$* range;

$record_{i,j}$ = number of cases where $(lb_i > v_c >= ub_i) \wedge (ld_j > v_k >= ud_j)$

}

$record_i = (record_i) / \max(record_i)$

}

$\text{var}_{c,k}$ = variance(variance($record$));

}

Dependence = mean(var);

Where Nranges is the number of ranges, *lb* and *ub* are the lower and upper bounds for ranges.

4. Results

In this work the network has 100 cells each of which receiving 800 input signals. The input signals are the values of the positive and negative parts of 20×20 patches of the whitened face images reshaped to two 1×400 vectors in a row. Each input sample is shown to the network for 100 times to have the dynamics of the network slowly adapted. The following is the measured characteristics of the network after more than 1 000 000 cycles of learning.

Table I

Network Characteristics

Algorithm	Base	Number of patterns	Dependency	Sparseness
1	$4000 + ((\bar{v} - 0.7)^+ / 0.3) \times 5000$	97	$2.5 \cdot 10^{-9}$	0.70
2	$2000 + ((\bar{v} - 0.7)^+ / 0.3) \times 10000$	70	$2.7 \cdot 10^{-10}$	0.99

Here the preferred algorithm is the Algorithm 1. As it can be seen in the Table 1, the highest number of patterns is achieved by the dynamic learning rate having a «base» with a smaller linear part (Algorithm1). The high sparseness in the second case can be the result of strong lateral (inhibitory) connections which is not much satisfying. In Addition, a very high sparseness, close to one, is not desired as in fact not much information is on the output to be feed to the next layers of the neural hierarchy. The higher dependency in the

algorithm 1 may come from the fact that the number of components is too high and they can not be highly independent. But in both algorithms the dependency is satisfyingly small. The constant 4000 in the algorithm 1 is more than in algorithm 2 in order to prevent too fast learning. $((\bar{v}-0.7)^{+}/0.3)$ has the minimum value of 0 if the mean activity of the cell is less than 0.7. That is, the linear part of the time constant is used just if the mean activity is higher than 0.7. Else the time constant will have its default value (2000 or 4000). An example of the receptive fields or weight matrices of the cells is represented on Fig. 1. Here one can observe how the learnt components look like.

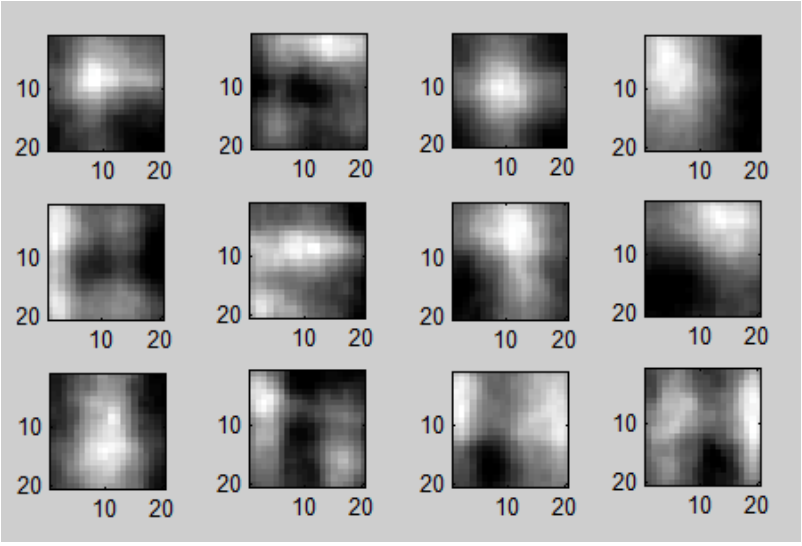

Fig. 1. Visualization of the learnt components (receptive fields) of the neural network. The images function of Matlab was used to visualize the weight matrices of the output layer's neurons

Conclusion

We observed that using different methods together could increase competition, reduce the dependence of the different cells and lead to learning different components. The dynamic learning rate helped the competition by giving new and meaningful initial values to the cells with low activities which with a high probability are the losers of the competition with other cells. This way, if a cell is the looser and its firing rate (activity) reduces, its learning rate increases and it learns the new inputs faster while the winners of the previous completions are tuning their patterns with a low leaning rate and more precise details. We observed (Table 1) that the lowest base we use for the learning time constant the better competition we have. But this is limited as a too much low base would lead to very fast increase in the weights. A high learning rate for the lateral weights which inhibits the activity of the neurons could prevent the neurons from correlation and learning the same patterns at very early stages of learning. A dynamic coefficient for inhibition maximizes the inhibition to the maximum value which doesn't kill the neural activity. A strong inhibition reduces the activity of the neurons, preventing the dynamic weight reduction factor α from increasing and leads to constant weight increase which is not desired.

Acknowledgment. This work was sponsored by Deutscher Akademischer Austausch Dienst (DAAD), Germany.

REFERENCES

1. *Haxby J.V.*, *Hoffmann E.A.*, *and Gobbini M.I.* Human Neural Systems for Face recognition and Social Communication // Society of Biological Psychiatry. 2002. No. 51. P. 59–67.
2. *Dayan P.*, *Abbot L.F.* Theoretical Neuroscience. Cambridge. Massachusetts: MIT Press, 2001. P. 45–122.
3. *Wiltschut J.*, *Hamker F.* Efficient coding correlates with spatial frequency tuning in a model of V1 receptive filed organization // Visual Neuroscience. 2009. No. 26. P. 21–34.
4. *Hebb D.O.* Organization of behavior. New York: Wiley, 1949. 335 p.
5. *Foldiak P.* Forming sparse representations by local anti-Hebbian learning // Biological Cybernetics. 1990. No. 64. P. 165–170.
6. *Hubel D.H. and Wiesel T.N.* Receptive fields, binocular interaction and functional architecture in the cat's visual cortex // J. Physiology. 1961. No. 160. P. 106–154.
7. *Olshausen B.A. and Field D.J.* Emergence of simple-cell receptive field properties by learning a sparse code for natural images // Nature. 1996. No. 381. P. 607–609.
8. *Oja E.* Simplified neuron model as a principal component analyzer // J. Mathematical Biology. November 1982. No. 15 (3). P. 267–273.
9. *Sejnowski T.J.* Storing covariance with nonlinearity interacting neurons // J. Mathematical Biology. 1997. No. 4. P. 303–321.
10. *Robert C.M.*, *Mark F.B.* LTP and LTD: An embarrassment of Riches // Neuron. 2004. No. 44. P. 5–21.
11. *Teichmann M.*, *Wiltschut J.*, *Hamker F.H.*, Learning invariance from natural images inspired by observations in the primary visual cortex // Neural Computation. 2012. No. 24(5). P. 1271–1296.
12. *Hoyer P.O.* Non-negative matrix factorization with sparseness constraints // J. Machine Learning Research. 2004. No. 5. P. 1457–1469.

Kermani Arash K., Technical University of Chemnitz
Spitsyn Vladimir G., Tomsk Polytechnic University
Hamker Fred H., Technical University of Chemnitz
E-mail: arash@hrz.tu-chemnitz.de; spvg@tpu.ru;
fred.hamker@informatik.tu-chemnitz.de

Поступила в редакцию 18 мая 2012 г.

Кермани А.К., (Технический университет Кемниц, Германия) *Спицын В.Г.*, (Томский политехнический университет)**,** *Хамкер Ф.Х.*, (Технический университет Кемниц, Германия).
Максимизация конкуренции в биологически правдоподобной нейронной сети.

Ключевые слова: нейронные сети, правило обучения Хэбба, антихэббовское обучение, правило Ойя, ковариантное обучение, конкуренция.

Целью данной работы является максимизация конкуренции среди биологически правдоподобных нейронов для получения максимального количества рецептивных полей. Биологически правдоподобная нейронная сеть в процессе обучения должна выучить различные независимые компоненты уникального входного сигнала. Сеть содержит 100 выходных нейронов, каждый из которых получает на вход 800 значений яркостей пикселей фрагментов изображений. Метод обучения нейронной сети основан на правиле Хебба. Для решения задачи максимизации конкуренции применяется антихеббовская модель модификации синаптических связей. Показана важная роль динамической скорости обучения, значение которой возрастает в случае низкого максимального выходного значения нейронов и уменьшается при высоком максимальном выходном значении нейронов. Это приводит к повышению конкуренции и созданию большого количества различных образов, характеризуемых соответствующими рецептивными полями.