INTERNATIONAL COMPUTER SCIENCE INSTITUTE

1947 Center Street • Suite 600 • Berkeley, California 94704 • 1-510-642-4274 • FAX 1-510-643-7684

Exploitation of Structured Gating Connections for the Normalization of a Visual Pattern*

Alessandro Sperduti

TR-93-038

Abstract

Structured gating connections can be useful to reduce the complexity of networks with a high number of inputs. An example of their application to the normalization of a visual pattern with respect to scale and position is presented. The use of gating connections allows us to have a linear number of connections in the number of pixels. The connections are also very localized.

*Paper to be presented at the *International Joint Conference on Neural Networks*, Oct. 1993, Nagoya, Japan.

1 Introduction

Usually, networks facing tasks involving high dimensional data need a large amount of connections and long training periods. In this paper we present an example of a neural network exploiting structured gating connections (see [1].) In particular, we show how gating connections of enabling type can reduce the number of connections in a network, provided that the desired function can be realized or approximated by the composition of simplier basic functions. The idea is to merge in a single recurrent network a set of feed-forward networks, each implementing a basic function. The gating connections are used to select one basic function at a time, and their composition is decided by another module according to the current state of the network. Here we give an example of how structured gating connections can be used in the normalization of a visual pattern with respect to scale and position. The structure of several neural networks for visual pattern recognition is inspired after the Neocognitron network [2], where several layers of neurons are connected in cascade by relatively localized but dense connections (see [5] for a shift-invariant network.) In this paper, we show how gating connections can lead to high performance in generalization since they allow the embedding of a priori knowledge in a neural network with a few units and sparse connections.

2 The Task

The task we have chosen consists in scale and position normalization of the representation of a single object in an image. Here the normalization is intended with respect to the image. Some examples of the images (128×128) we used are shown in Figure 1. The images were taken from several different points of view, and under different lighting conditions.

The images were preprocessed by a recurrent edge detector network able to perform a multi-scale analysis of them. The range of filters used by this network can be controlled by a single parameter. This allows to run the first iterations of the edge detector with a set of low-pass filters and then to turn on a set of filters with a higher cut-off frequency. In this way the shift and scale normalization network can use the output of the network at the first iterations to get information about the size of the object (the noise is removed) and the successive iterations to build the normalized representation of it. The second step is needed since low-pass filters tend to cut off several features of the image (see Fig. 2.)

3 Shift and Scale Normalization

More formally, our shift and scale problem can be stated as follows. Given an object o_j, we call its current input representation $R_{o_j}^{(i)}$ and its normalized representation R_{o_j}. The problem is to find the right values for the parameters α and d, such that:

$$R_{o_j} = \alpha R_{o_j}^{(i)} + d, \tag{1}$$

where α is a rescaling parameter and d the displacement one.

The strategy we propose to solve the problem is to use a recurrent network which transforms the object in the input image so that its final representation fits in a bounded

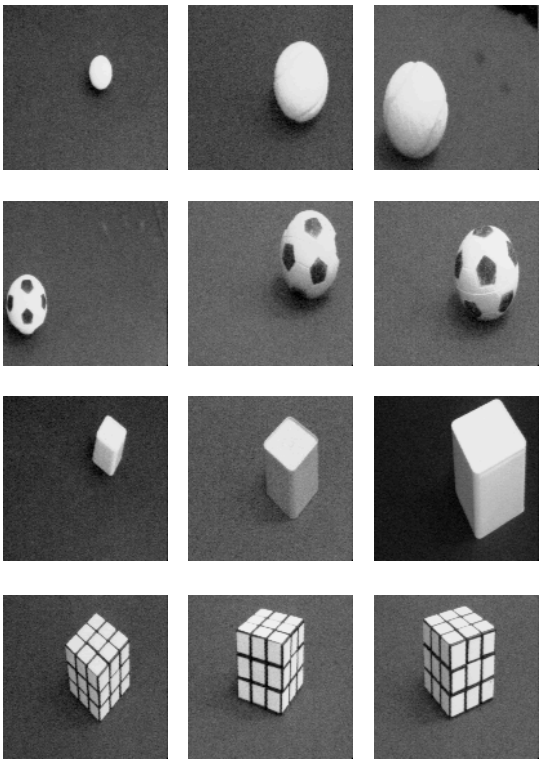

Figure 1: Some exemplars of the images.

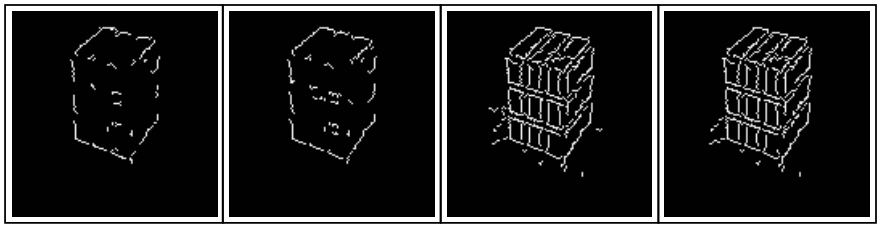

Figure 2: The output of the edge detector network as it evolves along time. The first picture from the left shows the output after one iteration. The second picture is the output at the fourth iteration, the last one with a range of low-pass filters. The last two pictures show instead the output of the network after five and seven iterations, respectively the first and third iterations which use a range of filters with an higher cut-off frequency.

region around the center of the output image (see Figure 3.) The scaling and the shifting of the image I are performed by two functions $shift_{d_x,d_y}(I_{x,y})$ and $scale_{\alpha,\sigma}(I_{x,y})$, where $\forall x, y$:

$$shift_{d_x,d_y}(I_{x,y}) = I_{x+d_x,y+d_y} \tag{2}$$

is a simple shift of the image and

$$scale_{\alpha,\sigma}(I_{x,y}) = H(1.5 \sum_{t,k} I_{t,k} e^{(-\frac{(t-\alpha x)^2+(k-\alpha y)^2}{2\sigma^2})}) \tag{3}$$

is a gaussian interpolating scaling function passed through the hard limiter $H()$ ($H(x) = 1$ if $x \geq 1, 0$ otherwise.) Since a direct implementation of these functions in a network would be resulted in an enormous amount of connections, we decided to restrict d_x and d_y to assume values in the set $\{-8, 0, 8\}$, while α and σ can assume only the joint values ($\alpha = 0.9, \sigma = 0.5$) (scale down) and ($\alpha = 1.1, \sigma = 0.538$) (scale up.) In total, we have 10 basic transformations (eight translations and two scalings) which can be composed to obtain an approximation of the desired transformation of the input image. The decision of how to compose them is taken by generating a scaled version of the image (128×128) in a second layer, layer B, of 16×16 neurons (see Figure 4.) This layer, according to the resulting activity of its neurons, selects the basic transformation to apply to layer A. Thus, for the example shown in Figure 4, it selects a translation towards the center ($\forall x, y,\ shift_{8,8}(I_{x,y})$.) When the input image is transformed, a new scaled version of it is computed in the layer B and the process is repeated until the activity of the neurons of that layer fits the 8×8 central region of B corresponding to the 64×64 central region of layer A.

3.1 Realization of the Basic Transformations

In this section we discuss how the basic transformations can be realized. Both translations and scalings can be implemented introducing the same connections among neurons into layer A. Translations can be realized by connecting each unit in position (x, y) to all the neurons in position $(x+d_x, y+d_y)$, $d_x, d_y \in \{-8, 0, 8\}$. Each connection has a weight equal to 1 and is enabled when the corresponding translation is selected. The introduction of connections realizing the scale functions is just a bit more complex. The localization and the weights on the connections change with the position of the unit under consideration, according to the definition of the parameters α and σ. In fact, the weights on the connections are defined by the following equation:

$$w_{(t-\alpha x,k-\alpha y)(x,y)} = 1.5 e^{(-\frac{(t-\alpha x)^2+(k-\alpha y)^2}{2\sigma^2})}, \tag{4}$$

where $w_{(t-\alpha x,k-\alpha y)(x,y)}$ is the weight on the connection from location $(t - \alpha x, k - \alpha y)$ to location (x, y). A high connectivity would be required to implement the general form of it, but since we allow only two couples of values for α and σ, we need only a few and very localized connections. In fact, since σ is small, the resulting gaussian function is narrow and decreases rapidly to zero. Moreover, the connections are very localized because α assumes values around 1. It results that all the connections needed to implement the basic transformations are localized within a 17×17 box around each neuron. To give an idea

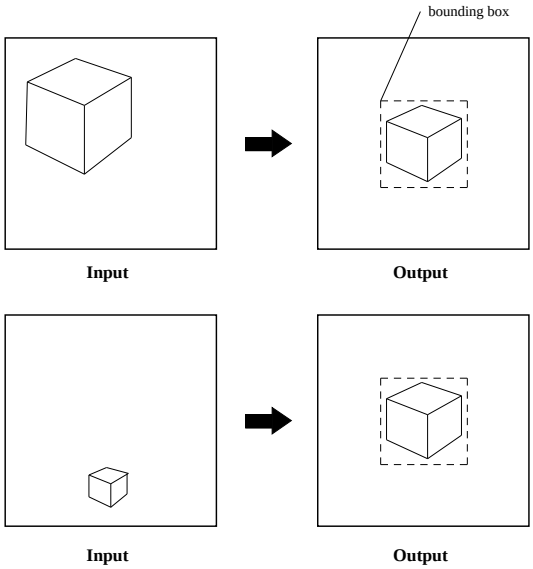

Figure 3: Normalization of the input image. The representation of the object in the input image is scaled and translated so to fit in the region bounded by the dashed square in the center of the output image.

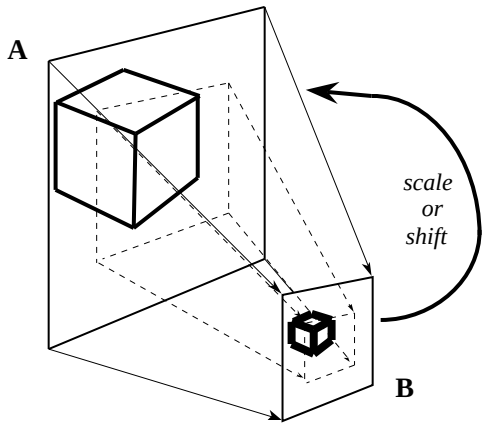

Figure 4: A schema of the recurrent network.

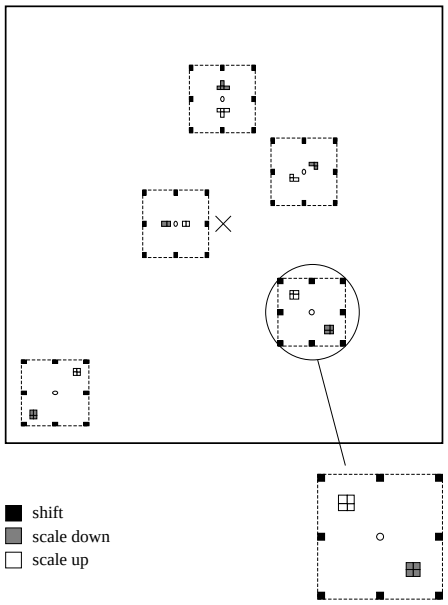

Figure 5: Realization of the basic transformations: examples of connections.

of how the localization of the connections changes, in Figure 5 we report a map of the connections, where only a few sparse neurons are in evidence. A unit receiving connections is represented by a small circle and its receptive field delimited by a dashed box (17×17.) The center of the layer A is marked with a cross. Small boxes represent neurons from which a connection starts and the color of the boxes represents the type of connection between them and the receiving unit. Translation connections are represented by black boxes, scale up connections by white boxes and scale down connections by gray boxes. Only relevant connections for the scale up and down transformations are represented. Note how scale up connections come from the inner part of the image, since the image must be enlarged, while scale down connections come from the external part, since in this case the image must be reduced. It can be seen that the number of significative weights is very small: each neuron in layer A has no more of 23 entering connections. The weights on the connections from layer A to layer B are not gated. They are defined according to the same gaussian interpolating function used in eqn.(4), with parameters $\alpha = 0.125$ and $\sigma = 0.225$.

3.2 Selection of the Basic Transformations

In this section we discuss how layer B selects the basic transformations. In order a scaling transformation to be selected, some constraints on the activation of B must be satisfied. A simple constraint consists in the activation of one or more neurons in a receptive field on B. The constraints which must be satisfied to apply the $scale()$ function are shown in Figure 6. The layer B is represented by a big square where each neuron is represented by a small square. A receptive field on the layer B is represented by a group of contiguous gray neurons. In the case of a scale down, at least one couple of receptive fields must be simultaneously active to satisfy the constraint (the first level neurons perform an AND,

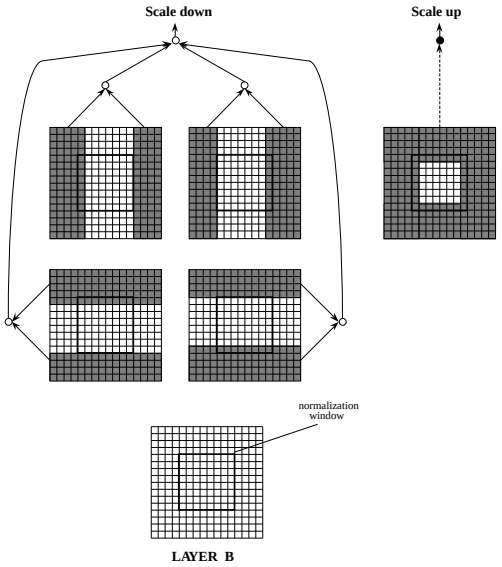

Figure 6: Selection of scaling transformations.

while the neuron at the second level performs an OR.) A scale up is instead selected if the representation of the object is contained in, but does not fill, the central region of the image (thick square.) This constraint has been implemented by a spontaneously active neuron (black neuron) which is inhibited by the receptive field shown on the right side of the picture. Note that, the scale up and scale down constraints are mutually exclusive, so that only one between them is selected at each time.

The translations along the vertical and horizontal axes are independently selected, so that the final global translation is given by the combination of them. Both the translations are selected using a compensatory mechanism which detects if the neural activity is unbalanced on one side. If the output of a neuron in the layer B at position (i,j) is named b_{ij}, then the activities of a column c_j and of a row r_i of layer B are defined as:

$$c_j = H(\sum_{i=1}^{16} b_{ij}), \quad r_i = H(\sum_{j=1}^{16} b_{ij}). \tag{5}$$

Using c_i, we are able to define how further the neural activity is displaced on the left (Δ_l) or right (Δ_r) side of B:

$$\Delta_l = \sum_{k=1}^{16} H(\sum_{i=1}^{k} c_i), \quad \Delta_r = \sum_{k=1}^{16} H(\sum_{i=k}^{16} c_i). \tag{6}$$

The top (Δ_t) and bottom (Δ_b) displacements can be defined in the same manner using the activity in the rows r_i. On the basis of these displacements, it is possible to select the translations ($tr_{x(y)}$) which reduce the unbalance in the neural activity of layer B:

$$tr_x = \begin{cases} shift_{8,*}() & \text{if } (\Delta_l - \Delta_r) > 1 \\ shift_{-8,*}() & \text{if } (\Delta_l - \Delta_r) < -1 \\ shift_{0,*}() & \text{otherwise} \end{cases} \tag{7}$$

$$tr_y = \begin{cases} shift_{*,8}() & \text{if } (\Delta_b - \Delta_t) > 1 \\ shift_{*,-8}() & \text{if } (\Delta_b - \Delta_t) < -1 \\ shift_{*,0}() & \text{otherwise} \end{cases} \quad (8)$$

Since, at a given time, it may happen that both a scaling and a translation are selected, and our network does not allow this, we have given highest priority to the scaling functions.

To avoid that the noise could disturb the selection of the transformations, the layer A is duplicated in layers A_1 and A_2. One layer (A_1) is loaded with the output of the edge detector at the end of the first iteration phase (low-pass filters), the second one (A_2) with the final output of the network (high-pass filters.) Layer B is connected only with layer A_1. The transformations decided by layer B are applied to both A_1 and A_2.

3.3 Execution of the Basic Transformations

In order to execute a basic transformation, the output of layer B must be transformed in control signals for the gating connections. For this reason, we define the output of a neuron in layers A_1 and A_2 as:

$$o_{x,y}(t+1) = H(\sum_{i,j} w_{(i,j)(x,y)} g_{(i,j)(x,y)}(t) o_{i,j}(t)), \quad (9)$$

where $g_{(i,j)(x,y)}(t)$ is a dynamical binary control signal gating the connection from (i,j) to (x,y) at time t. Since each transformation uses different lines of input, we can use a control signal for each translation connection and one control signal for each receptive field implementing the scale up or scale down transformations. In total, we need only 10 control signals and 11 control patterns (one for each transformation plus the null control pattern used to implement the null transformation.) The control patterns can be distributed directly or in an encoded form (using an encoder network) to layers A_1 and A_2. The distribution can be organized in a hierarchical fashion in such a way to localize the connections. A very simple network can be trained to learn the correspondence between the output of layer B and the desired control pattern. When the input to the network is null (the image is normalized) the null control pattern is activated so that the normalized image is no more transformed.

Some examples of the evolution of the pattern of activation of layer B are shown in Figure 7. Note that, the first activation pattern of layer B can be used to localize the object, and thus it solves the classical *where* problem ([3, 4].) Moreover, the sequence of control patterns selected to normalize the image contains information about the size of the object. Even if this information is dependent on the point of view, it can be used to give a qualitative estimation of the size of the object. Some examples of the output of the normalization network are shown in Figure 8.

4 Conclusions

In this paper we have presented an example of a network exploiting gating connections. The task faced by the network was scale and position normalization of the representation of a single object in an image. The use of structured gating connections has allowed us

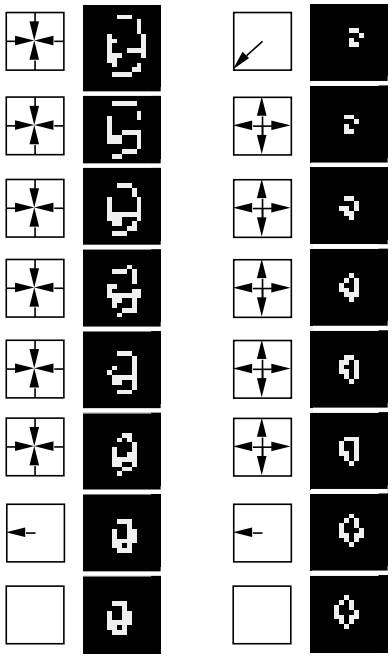

Figure 7: Two examples of evolution of the activation pattern of layer B. The left sequence refers to a cube, the right to a sphere. On the left side of each activation pattern is shown the transformation selected for the successive iteration.

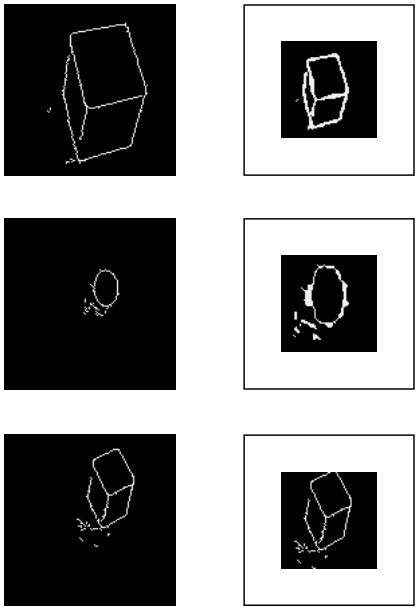

Figure 8: Some examples of the normalization performed by the network.

to build a network with localized and not too many connections. In fact, the number of connections is linear with respect to the number of pixels in the image. Moreover, a priori knowledge on the problem has been embedded to such an extent that only a small part of the network needed learning. However, learning on the whole network can be performed in order to improve the output of the network. In fact, because of the approximations on the sampling of the scaling and translation basic functions, and of the approximations introduced by the discrete interpolation of the scaling functions, the output of the network is sometimes blurred. The advantage to have very localized connections is paid in terms of response speed. However, this should not be a problem, since the response of the network is still fast enough and bounded in time. Moreover, the sequence of control patterns generated by the layer B can be used to extract qualitative information on the size of the object. The main advantage of this approach is that generalization is guaranteed a priori.

References

[1] R. Durbin and D. E. Rumelhart. Product units: a computationally powerful and biologically plausible extension to backpropagation networks. *Neural Computation*, 1:133, 1989.

[2] K. Fukushima, S. Miyake, and I. Takayuki. Noecognitron: a neural network model for a mechanism of visual pattern recognition. *IEEE Transaction on Systems, Man, and Cybernetics*, 13:826–834, 1983.

[3] R. A. Jacobs, M. I. Jordan, and A. G. Barto. Task decomposition through competition in a modular connectionist architecture: The what and where vision tasks. Technical Report COINS 90-27, Dept. Comput. and Inform. Sci., Univ. Mass., 1990.

[4] B. Olshausen, C. Anderson, and D. Van Essen. A neural model of visual attention and invariant pattern recognition. Technical Report CNS Memo 18, California Institute of Technology Computation and Neural Systems Program, 1992.

[5] W. Zhang, A. Hasegawa, K. Itoh, and Y. Ichioka. Error back propagation with minimum-entropy weights: A technique for better generalization of 2-d shift-invariant nns. In *International Joint Conference on Neural Networks*, pages 219–224, 1991. Seattle.