Learning Control Units for Invariant Recognition

Junmei Zhu and Christoph von der Malsburg*

Computer Science Department, University of Southern California

Los Angeles, CA 90089-2520

{junmeizh, malsburg}@organic.usc.edu

Abstract

A control unit is a group of synapses that are consistent with each other in terms of transformation parameters. We showed previously that these high-order interactions led to very fast Dynamic Link Matching (DLM) for self-organization of mappings between two patterns, as needed in invariant recognition and many other visual tasks. We present here a model to learn from examples control units and the connection between them, by generalized Hebbian plasticity. Examples for learning are consistent mappings formed, for instance, by conventional DLM. Simulation results for 1D patterns showed that the learned results resemble those designed by hand, and thus were very encouraging.

Keywords: Dynamic Link Matching, invariant recognition, control unit, Hebbian learning.

1 Introduction

We are interested in object recognition in the presence of variations such as transformations (translation, scale, orientation), illumination, and deformations. Although this task is solved fast and effortlessly by humans, it is very computationally difficult because of the enormous pictorial changes due to the variations.

*Also at Institute for Neuroinfomatik, Ruhr-University Bochum, Germany

The most important process underlying invariant recognition, as well as other visual tasks such as motion estimation and stereo correspondence, we believe, is to create a mapping between two patterns, such that points projected from the same physical spot are connected.

This is the well-known correspondence problem in computer vision, where a lot of algorithms have been developed. That most of the algorithms do not work well outside of well-controlled laboratory settings, unfortunately, suggests the drawbacks of this algorithmic schema. In general, the two aspects of solving a problem, algorithm and implementation, are handled by humans and computers respectively. Human creativity produces algorithms as well as the corresponding codes to be fed into the computer, whereas computers follow the codes blindly. In consequence, computers on their own cannot respond to situations that are not programmed explicitly. Human intervention is required to get a new code whenever a new problem comes up. To reduce this bottleneck, a new computing paradigm should be developed to shift the creativity aspect to the computer. The inspiration of this new paradigm comes from biological systems that provide ingenious solutions to many of the problems we are interested in, such as invariant recognition, and is called organic computing.

Our work is an attempt in the direction of organic computing, the aim of which is, by using only basic organization principles, to build a system that can learn and develop the equivalent of human created algorithms within itself, to perform whatever task it may encounter.

Dynamic link matching (DLM) is such an example system that creates efficient solutions based on only basic self-organizing rules. The basic principle it uses is rapid synaptic plasticity, in which a mapping between two patterns is controlled by signal correlations, which are shaped by neighboring connections and by the mapping itself. DLM is intrinsically invariant to translation and is robust against illumination and deformation, and was proven to be very successful. Although not especially designed for a particular task, its performance in face recognition outperformed all other human created algorithms [1].

However, a difficulty for DLM as a biological model, although based on biologically plausible principles, is its relative slowness in comparison to the split-second recognition times in adults. We previously presented an extension to DLM that has significantly shorter convergence time[3, 4]. The extension is based on *synaptic control units*, each of which stands for a group of synapses that are consistent with each other in terms of transformation parameters. Control units and their connections represent stored knowledge of transformations that leads to the speedup of DLM convergence: significant local connectivity patterns can be retrieved by control units and be propagated rapidly through the interconnections between control units.

In the previous studies, control units and the connections between them were all designed manually, which was possible because the variations we considered then were scale, translation, and in-plane rotation. This kind of analytical solution, nonetheless, not only is impossible when deforming patterns become complicated, but also belongs to the algorithmic schema that should be replaced. Therefore we present here a model to learn control units and the connections between them from examples.

This paper is organized as follows. Section 2 provides the necessary background in DLM and control units, making clear what are expected to be learned. Section 3 gives the learning rules, and an overview of the learning system. Section 4 provides some simulation results. Finally, section 5 is the conclusion.

2 Background: control units in Dynamic Link Matching

In this section, we provide some background on DLM and the control units, serving as a basis for what we are going to learn.

2.1 DLM

DLM self-organizes a mapping between two patterns, an image and a model [2]. The mapping is represented by a connection matrix $X = x(r,t)$, where r is the position in the image domain, and t in the model domain. r and t are 2D vectors if the patterns are 2D. Each $x(r,t)$ is called a dynamic link, or a synapse, and they are the system variables that can have rapid changes. Synaptic growth is controlled by cooperation and competition. For each $x(r,t)$ the cooperation term is obtained by gathering support from all synapses that are its spatial neighbors in the image domain and the model domain.

2.2 Control units

Control units were introduced to speedup DLM convergence and to deal with transformations, especially scale and in-plane rotation, more efficiently [3, 4]. Control units stand for groups of synapses, which represent stored significant patterns. They control synapses by regulating how synaptic cooperation is computed. The speedup of DLM is achieved by the specific long-range synaptic cooperative interactions carried out by the interaction between control units, where the specificity comes from the fact that any single synapse is ambiguous as to the transformation parameters, but a group of synapses, as represented by control units, are unambiguous.

There are two aspects of a control unit we need to learn: a control function, specifying what synapses are under its control, and its connection to other control units. Detailed description of control units can be found in [4]. In short, a control function has the same domain as the connection matrix, and is represented by a real function $K(r, t)$. It is specific to transformation parameters, and is localized in variables (r, t), with the center of its compact support called the control center. As an example, for 1D patterns, a control function may have the form of a 2D Gaussian, whose center is controlled by the translation parameter, and the axes by the scale parameter. The connections between control units are also intuitive. In theory, two control units have strong connections between them if their control centers are close, and they are neighbors in the transformation space, i.e., similar translation, scale, and orientation.

3 Learning control unit connections

This section presents the learning rules and the learning system. Both control units and their connections are learned by generalized Hebbian plasticity.

3.1 Learning examples

The idea of this learning system is as follows. It is reasonable for a baby to stare at an object for several minutes before it can recognize it. But once a connectivity pattern is formed, it should be stored into the control units to speed up the convergence of future map creation. Therefore, examples used for learning control units are consistent mappings formed, for instance, by slow conventional DLM. Each mapping is represented by a synaptic matrix $x(r,t)$. An advantage of learning mappings, instead of learning patterns as in most learning systems, is that the weight of a connection in a mapping has a physical meaning, while the intensity level of a pixel in a pattern does not.

3.2 System structure

In reality, the distribution of sizes or centers of control units may be random as a result of competitive learning, but here we consider a simplified and regular structure.

The domain of mappings is divided into small regions called control regions, indexed by s, with membership function $A_s(r,t)$. Different regions can overlap with one another. For each s there is a set of control units, with control functions $K_{si}(r,t)$. The response of unit $K_{si}(r,t)$ to input $x(r,t)$, y_{si}, is:

$$y_{si} = \sum_{r,t} x(r,t) A_s(r,t) K_{si}(r,t) + \sum_{s',i'} c(s,i,s',i') y_{s'i'}. \tag{1}$$

where $c(s,i,s',i')$ is the connection strength between control units K_{si} and $K_{s'i'}$.

3.3 Learning rules

3.3.1 Control functions

For each input mapping $x(r,t)$, the learning rule for control functions is:

$$\triangle K_{si}(r,t) = \alpha A_s(r,t) x(r,t) y^*_{si}, \tag{2}$$

where

$$y^*_{si} = \begin{cases} y_{si} & \text{if } argmax_j(y_{sj}) = i \\ 0 & \text{otherwise} \end{cases}$$

and α is the learning rate.

In order to avoid uncontrolled growth, this update is followed by a normalization:

$$\sum_{r,t} K_{si}(r,t) = \text{const.} \tag{3}$$

3.3.2 Connection between control units

The connection between control units K_{si} and $K_{s'i'}$, $c(s,i,s',i')$, is also learned by Hebbian rule:

$$\triangle c(s,i,s',i') = \beta y^*_{si} y^*_{s'i'}, \tag{4}$$

where β is the learning rate. This is also followed by normalization:

$$\sum_{s',i'} c(s,i,s',i') + \sum_{s,i} c(s,i,s',i') = \text{const.} \tag{5}$$

3.4 The learning system

The learning system is summarized as follows:

- Initialization
  - control function: random values within its control region, 0 otherwise
  - control unit connections: 0
- For each training example
  - compute control unit responses (equation (1))
  - update control functions (equations (2) and (3))
  - update connections (equations (4) and (5))

4 Experiments

The development of control units and their connections is simulated in an example in which the goal is to learn translation invariance for 1D patterns.

4.1 Examples for learning

Each learning example is a mapping between a pair of relatively shifted image and model. Image and model are identical patterns up to a scale difference. Accordingly the synaptic matrix has value 1 along a straight line, whose slope depends on the scale difference, and 0 elsewhere. Some examples are shown in figure 1. Mappings in the upper row are from identical image and model, while those in the lower row have a size difference of 0.5 octave. The position of the line corresponds to the translation parameter, which is generated randomly and uniformly over all possible translations. Noise in the mapping is simulated by randomly setting elements next to the ideal line value 1.

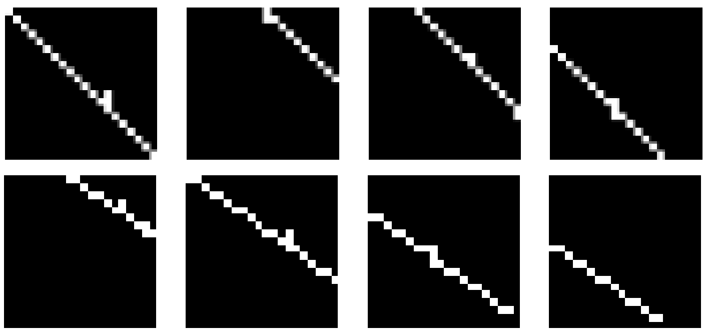

Figure 1: Examples of synaptic matrices used for learning translation invariance: mappings with different translation parameter. Row 1: identical image and model; Row 2: image and model has scale difference of 0.5 octave.

4.2 Results

In this simulation, the synaptic matrix is of size 20×20, and is divided into 4×4 small regions, each of which has size 5×5. In each small region there are 9 control units, to cover all possible translations. In the following simulation, different scales are learned separately, which means in each run all examples have the same and fixed scale. Learning examples are presented sequentially, and the system learns according to the procedure in section 3.4. We use normalization constant$=1$ for control functions, and 0.2 for control unit connections. $\alpha = 0.1$, $\beta = 0.05$.

The initial random control functions are shown in figure 2(a). The learned control functions after 1000 learning examples are shown in figure 2(b) and 2(c). What is plotted is only the control functions within their respective control regions. Figure 2(b) shows the functions learned with examples with an image and model of the same scale, while in 2(c) the scale difference is 0.5 octave. Each small block represents the control function of one control unit. In the horizontal direction runs s, the index of control center position in the synaptic matrix. In the vertical direction is i, the index of control units sharing the same control center. We can see that the learned functions have reasonable shapes, resembling those designed by hand previously [4]. Also it seems that control units which share the same control center learned different translation parameters, which often span the whole range of possible translations. But we also noticed that even after 1000 training examples, it is still possible to have a few uncommitted slots, i.e., control units with unstructured control functions.

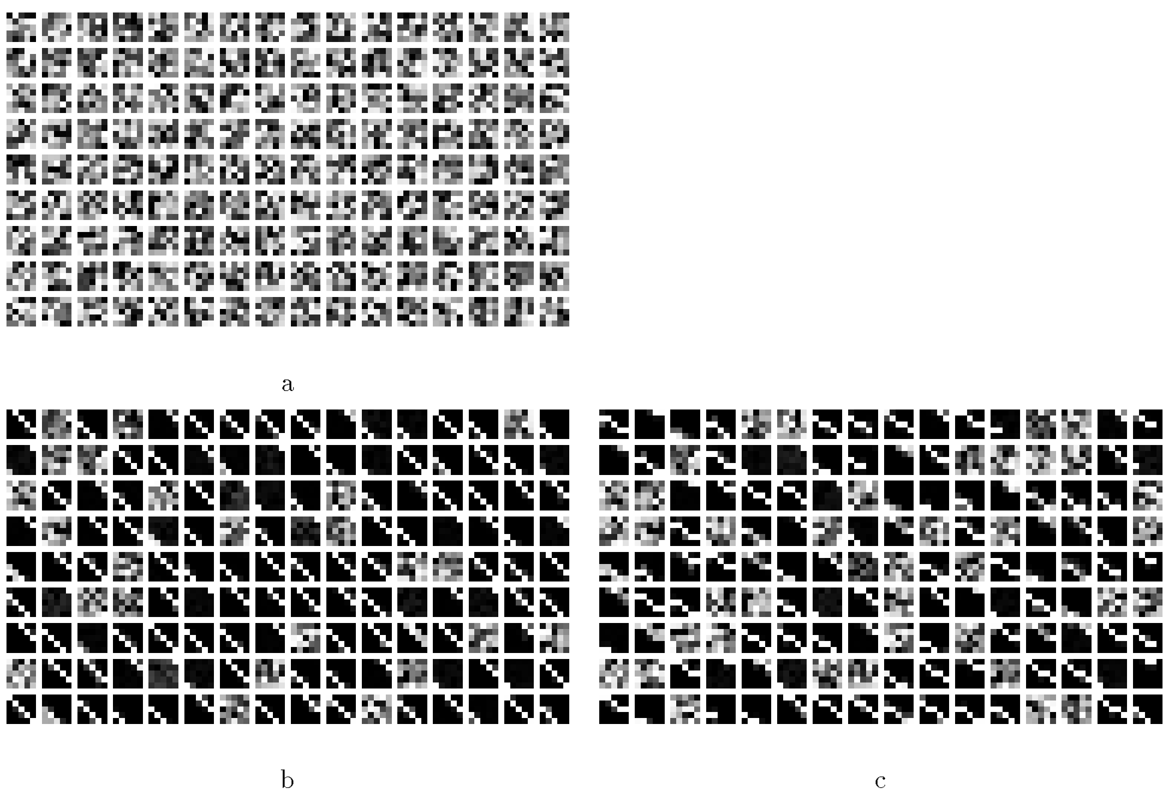

Figure 2: Control functions. (a): initial values. (b) and (c): learned control functions after 1000 iterations. (b): identical image and model; (c): image and model has scale difference of 0.5 octave.

The learned interconnections between units are evaluated in the following way. For any control unit indexed by si, compute the summation of the control functions of all control units, weighted by their connection strength to unit si: $\sum_{s',i'} c(s,i,s',i')A_{s'}(r,t)K_{s'i'}(r,t)$. In most instances, the reconstructed result resembles one of the training examples, i.e., a straight line with appropriate slope, meaning that all control units with the same translation are connected, while those with different translation are not.

We also did simulations on learning simultaneously shift and scale invariance for 1D patterns. The results are very encouraging, but still more work is needed.

5 Conclusion

This work demonstrates the feasibility to learn control unit functions and the connections between control units. Both can be learned by generalized Hebbian plasticity. The same learning rules apply to 2D patterns and to more complicated invariances.

Future work includes learning simultaneously multiple invariances, and eventually using these learned results to replace the hand crafted ones in the previous system.

Acknowledgments:

The authors would like to thank the developers of the FLAVOR software environment, which served as the platform for this work. This work was supported by ARO-WASSP, contract DAAD19-00-1-0356.

References

[1] P.J. Phillips, H. Moon, S.A. Rizvi and P.J. Rauss, The FERET evaluation methodology for face-recognition algorithms, *IEEE Trans. Pattern Analysis and Machine Intelligence* 22 (2000) 1090 -1104.

[2] L. Wiskott and C. von der Malsburg, Face Recognition by Dynamic Link Matching, in *Lateral Interactions in the Cortex: Structure and Function* (Electronic book, 1996).

[3] J. Zhu and C. von der Malsburg, Fast Dynamic Link Matching, *ICCNS*01* (Boston 2001).

[4] J. Zhu and C. von der Malsburg, Synapto-Synaptic Interactions speed up Dynamic Link Matching, *CNS*01*, *Neurocomputing* 44 (2002) 721-728.