Closed-Loop Object Recognition Using Reinforcement Learning

Jing Peng and Bir Bhanu

College of Engineering
University of California, Riverside, CA 92521
Email: {jp,bhanu}@constitution.ucr.edu

Abstract

This paper describes a closed-loop model-based object recognition system that determines its criteria for segmentation by using the recognition algorithm as part of the evaluation function. The confidence level of the recognition process serves as a reinforcement signal for a team of learning automata to generate segmentation parameters. This results in the performance improvement of the recognition system and generation of recognition strategies automatically. The system is verified through computer simulations on a sequence of indoor color images with varying lighting conditions.

1 Introduction

In order to develop real-world object recognition systems that will work under changing environmental conditions, it is essential to combine the interaction between segmentation and recognition components of an image understanding system. The fixed set of parameters used in segmentation, feature and recognition algorithms lead to undesired, ungraceful degradation of performance. No algorithms, no matter how sophisticated they may be, will lead to good performance for practical applications if they cannot adapt to the changing environmental conditions.

There are several main problems with current object recognition systems that are generally open loop systems. First, the segmentation and feature selection are done as preprocessing steps. They totally ignore the effects of the earlier results on the performance of the recognition algorithm. The usefulness of segmented image or selected features can only be determined by the final outcome of the recognition process. Second, generally the criteria used for segmentation and feature extraction require elaborate human design. When the condition for which they are designed is changed slightly, they may fail gracelessly. Furthermore, the criteria themselves can be a subject of debate. Finally, object recognition is a process of making sequences of decisions, and often the consequences of a decision can only be determined at the end of the recognition process. Object recognition systems whose decision criteria or strategies are developed autonomously from a reinforcement signal of the final recognition might transcend all these problems.

In this paper, we introduce a reinforcement learning-based closed-loop object recognition system, shown in Figure 1, that determines its criteria for segmentation and feature selection by taking into account the biases of the recognition algorithm. By using the recognition algorithm as part of the evaluation function, the system is able to develop autonomously its recognition strategies, and thus improves its recognition performance.

2 Reinforcement Learning

Reinforcement learning is a framework for learning to make sequences of decisions in an environment. In this framework, a system is given at each time step inputs describing its environment, the system makes a decision based on these inputs, thereby causing the environment to deliver to the system a reinforcement whose value depends on the environmental state, the system’s decision, and possibly random disturbances. Note that, in general, reinforcement measuring the consequences of a decision can emerge at a multitude of times after the decision is made. The objective of the system is to select sequences of decisions to maximize the sum of future reinforcement over time.

As noted above, a complication to reinforcement learning is the timing of reinforcement. In simple tasks, the system receives, after each decision, reinforcement indicating the goodness of that decision. The best understood method in immediate reinforcement learning is the REINFORCE algorithms of Williams [4], a class of connectionist reinforcement learning algorithms, that perform stochastic hillclimbing.

In more complex tasks, however, reinforcement is often temporally delayed, occurring only after the execution of a sequence of decisions. For example, in object recognition, the goodness of segmentation is not known until the recognition decision has been made. Delayed reinforcement learning is attractive and plays important
role in machine vision.

3 Learning Image Segmentation

Our initial investigation into reinforcement learning-based object recognition is focused on learning image segmentation that takes into account the biases of the recognition algorithm in order to select a set of parameters that will result in high recognition accuracy on newly acquired images.

We begin with image segmentation because it is an extremely important and difficult low-level task. All subsequent interpretation tasks including object detection, feature extraction, object recognition, and classification rely heavily on the quality of the segmentation process. The difficulty arises when segmentation performance needs to be adapted to the changes in image quality, which is affected by variations in environmental conditions, imaging devices, time of day, etc.

Figure 2 depicts the conceptual diagram of our reinforcement learning-based object recognition system that addresses the parameter selection problem encountered in image segmentation tasks by using the recognition algorithm itself as part of the evaluation function. The PHOENIX algorithm [3] was chosen as the image segmentation component in our system because it is a well-known method for the segmentation of color images with a number of adjustable parameters and fits our purpose here well.

The feature extraction consists of finding polygon approximation tokens for each of the regions obtained after image segmentation. The polygon approximation is obtained using a split and merge technique [2] that has a fixed set of parameters.

Object recognition employs a cluster-structure matching algorithm [2] that takes as input two sets of tokens, one of which represents the stored model and the other represents the input region to be classified. It then performs topological matching between the two token sets and computes a real number that indicates the confidence level of the matching process. This confidence level is then used as a reinforcement signal to drive the reinforcement learning algorithm, REINFORCE, which is described in the following section.

Note that Bhanu and Lee [1] describe a system that uses genetic algorithms for learning PHOENIX’s parameters. However, the recognition algorithm is not part of the evaluation function in their system. The genetic algorithms simply search for a set of parameters that optimize a prespecified evaluation function.

4 REINFORCE Algorithms

REINFORCE is a class of connectionist reinforcement learning algorithms developed by Williams [4], where units in such a network are Bernoulli logistic units [4].

In the general reinforcement learning paradigm, the network generates an output pattern and the environment responds by providing the reinforcement r as its evaluation of that output pattern, which is then used to drive the weight changes according to the particular reinforcement learning algorithm being used by the network. For the Bernoulli logistic units used in this research, the REINFORCE algorithm we use prescribes weight increments equal to

$$\Delta w_{ij} = \alpha(r - b)(y_i - p_i)x_j$$

(1)

where $\alpha$ is a positive learning rate (possibly different for each weight), $b$ serves as a reinforcement baseline (which can also be different for each weight), $y_i$ is the output of the $i$th Bernoulli unit, $p_i$ is an input parameter to a Bernoulli random number generator and computed according to (3), and $x_j$ is the input to the Bernoulli unit.
It can be shown [4] that, regardless of how \( b \) is computed, whenever it does not depend on the immediately received reinforcement value \( r \), such an algorithm satisfies

\[
E\{\Delta W|W\} = \alpha \nabla_W E\{r|W\}
\]  

(2)

where \( E \) denotes the expectation operator. A reinforcement learning algorithm satisfying (2) can be loosely described as having the property that it statistically climbs the gradient of expected reinforcement in weight space. For extensive discussions of these algorithms, see [4, 5]. Next two subsections describe the particular network and the REINFORCE algorithm used in the experiments reported here.

4.1 The Team Network

In this work, we used a very simple form of trial generating network in which all of the units are output units and there are no interconnections between them. This degenerate class of network corresponds to what is called a team of automata in the literature on stochastic learning automata. We thus call these networks teams of Bernoulli logistic units.

For any Bernoulli logistic unit, the probability that it produces a 1 on any particular trial given the value of the weight matrix \( W \) is

\[
Pr\{y_i = 1|W\} = p_i = f(s_i) = \frac{1}{1 + e^{-s_i}},
\]  

(3)

where \( s_i = \sum_j w_{ij} x_j \). The weights \( w_{ij} \) are adjusted according to the particular learning algorithm used. We note that when \( s_i = 0 \), the unit is equally likely to pick either 0 or 1, while increasing \( s_i \) makes a 1 more likely. Adjusting the weights in a team of Bernoulli logistic units is thus tantamount to adjusting the probabilities for the individual components.

Note that, except bias terms, there are no input connections in the team networks experimented in [5]. In contrast, the team network described here does have input weights which play the role of long-term memory in associative learning tasks. Specifically, each unit in the team network has a total 48 input weights, each of which takes input on a 40 by 40 neighborhood on the input plane. This “weight sharing” can be interpreted as imposing equality constraints among the connection strengths. It greatly reduces the number of free parameters in the network. For weights that are adjacent in a unit, their receptive fields are 20 pixels apart in the input image. Thus, the input image is undersampled. The motivation is that variations in lighting need not be determined with high resolution.

4.2 The Team Algorithm Used

The algorithm we used with the team architecture has the following general form: At the \( t \)th time step, after generating output, \( y(t) \), and receiving reinforcement \( r(t) \), i.e., the confidence level indicating the matching result, increment each weight \( w_{ij} \) by

\[
\Delta w_{ij}(t) = \alpha (r(t) - \bar{r}(t-1))(y_i(t) - \bar{y}_i(t-1)) - \delta w_{ij}(t)
\]  

(4)

where \( \alpha \), the learning rate, and \( \delta \), the weight decay rate, are parameters of the algorithm. \( \bar{r}(t) \) is computed according to \( \bar{r}(t) = \gamma \bar{r}(t - 1) + (1 - \gamma) r(t) \). The trace parameter \( \gamma \) was set equal to 0.9 for all the experiments reported here. \( \bar{y}_i(t) \) is similarly computed by the same exponential weighting scheme used for \( r \). The size of the weight rate \( \delta \) was chosen to be 0.01 in all our experiments.

5 Experimental Demonstration

This section describes experimental results evaluating the performance of our system on a set of indoor images. As a comparison, the segmentation results with the PHOENIX algorithm [3] with default parameter setting were also obtained for feature extraction and recognition in Figure 2.

The PHOENIX algorithm has a total of fourteen adjustable parameters. The four most critical ones that affect the overall results of the segmentation process are used in learning. These parameters are \( hsmooth, maxmin, splitmin, \) and \( height \). The REINFORCE algorithm searches for the combination that gives, using PHOENIX, a segmentation that results in the best recognition. The ranges for each of these parameters are the same as those used in [1]. The resulting search space is about one million sample points. Each of the PHOENIX parameters are represented using 5 bits with Gray coding. One reason for using the binary representation is that it offers a combinatorially advantageous way of approaching learning problems having large search space. While the same task could be learned in the original parameter space, for many types of problems the binary representation can be expected to learn much faster.
5.1 The Segmentation Tasks

The segmentation tasks whose experimental results we report here are a sequence of indoor color images (160 by 120 pixels) having simple geometric objects with varying lighting conditions. Figure 3 shows these images, where, from left to right, images are moving away from the camera, and within each column, lighting conditions deteriorate from top to bottom. The objective of the segmentation task is to find a set of parameters that will result in regions which after appropriate feature extraction will match the triangular object in the input images. Note that, unlike previous work on image segmentation, the criteria measuring image segmentation quality here are completely determined by the matching algorithm itself.

5.2 Simulation Results

Figure 4 shows the segmentation performance of the PHOENIX algorithm with learned parameters on the images shown in Figure 3. Note that input to the network is the luminance image of the corresponding color image. The system was trained on the images in the first column. When the resulting system was applied to the images in the second and third columns, no learning took place. This is intended to test system's generalization capability. The results in Figure 4 are obtained after an average (over 10 runs) of 2000 passes through the data. Figure 5 shows the average reward received by the system over time during learning. It should be noted that when the resulting segmented images were used as input to the matching algorithm, a 100% confidence level was achieved on most images. Furthermore, additional experiments show that when learning is allowed to take place on the images where the system has failed, it took much less computational effort (an average of 300 iterations) for the system to find a set of parameters that resulted in a 100% matching.

![Graph](image)

Figure 5: Average reward received over time.

In comparison, the PHOENIX algorithm with default parameter setting was also used on the same images. We omit the details of the experiment, but note that this default parameter setting resulted in a total matching failure.

6 Conclusion

Our investigation into reinforcement learning-based image understanding shows that a robust and adaptive system can be built that determines autonomously the criteria for detection and segmentation of the input images and selects useful features which will result in a system with high recognition accuracy on new images. In our future research, we will address the issue of learning feature selection and apply our approach to real-world problems.

Acknowledgements

This work was supported by ARPA/AFOSR grant F49620-93-1-0624 and ARPA grant MDA972-93-1-0010. The contents of the information does not necessarily reflect the position or the policy of the Government.

References


1994 Image Understanding Workshop

Proceedings
Hyatt Regency Hotel
13 - 16 November
Monterey, California
Volume I

Image Understanding Research Program
University of California at Riverside
Host of 1994 Image Understanding Workshop

Sponsored by:
Advanced Research Projects Agency
Software and Intelligent Systems Technology Office
Image Understanding Workshop

Proceedings of a Workshop
held in
Monterey, California

November 13 - 16, 1994

Volume I

Sponsored by:
Advanced Research Projects Agency
Software and Intelligent Systems Technology Office

This document contains copies of reports prepared for the ARPA Image Understanding Workshop. Included are Principal Investigator reports and technical results from both the basic and strategic computing programs within ARPA/SISTO sponsored projects and certain technical reports from selected scientists from other organizations.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Advanced Research Projects Agency or the United States Government.