Learning Active Vision

Hilbert J. Kappen, Marcel J. Nijman, Tonnie van Moorsel
RWCP Novel Functions SNN Laboratory
Dept. of Medical Physics and Biophysics, University of Nijmegen
Geert Grooteplein 21, NL 6525 EZ Nijmegen, The Netherlands
Tel: +31 80 614241
Fax: +31 80 541435
email bert@mbfys.kun.nl

Abstract

In this paper we introduce a new type of problem which we call active decision. It consists of finding the optimal subsequent action, based both on partial observation and on previously learned knowledge. We propose a method for solution, based on Boltzmann Machine learning of joint input-output probabilities and on an entropy minimization criterion. We show how the method provides a basic mechanism for active vision tasks such as saccadic eye movements.

Keywords: Boltzmann Machines, Active Vision, Rule extraction
To be presented at ICANN'95, October 1995, Paris
FTP-host: ftp.mbfys.kun.nl
FTP-file: snn/pub/reports/Kappen.Active.Vision.ps.Z

Learning Active Vision

Hilbert J. Kappen, Marcel J. Nijman, Tonnie van Moorsel
RWCP[1] Novel Functions SNN[2] Laboratory
Dept. of Medical Physics and Biophysics, University of Nijmegen
Geert Grooteplein 21, NL 6525 EZ Nijmegen, The Netherlands

Abstract

In this paper we introduce a new type of problem which we call active decision. It consists of finding the optimal subsequent action, based both on partial observation and on previously learned knowledge. We propose a method for solution, based on Boltzmann Machine learning of joint input-output probabilities and on an entropy minimization criterion. We show how the method provides a basic mechanism for active vision tasks such as saccadic eye movements.

1 Introduction

The problem of active vision can be generally formulated as follows. Given some partial observation of an unknown object, which additional partial observations are needed to allow recognition. The partial observations lead to one or more hypotheses or expectations about the identity of the object. This is done by assigning a probability for the object to belong to one of a number of known classes. These probabilities are based on 'experience' about what type of objects exist, and on what partial observations of these objects look like. Therefore active vision is controlled by both the sensory data as well as top-down expectations.

For autonomous vehicles and movable camera systems which are involved in object recognition, an efficient strategy for active vision is of great importance. This is because the physical movements of robots are time consuming, thereby limiting the total number of accessible 'views'.

Special cases of active vision are feature selection and next-view planning. Feature selection is the problem to find the set of features that best identify a given object. Next-view planning deals with the problem to identify the 3D structure of an object from a number of 2D views. In both cases, based on partial information, the problem is which additional view or feature will maximally improve recognition. For instance, in [9] a method is introduced to identify the parameters of a "model" (i.e. the size of a sphere) by a gradient strategy from a sequence of 2D views. In [2] next views are selected by explicitly using a probabilistic search through feature observations. The problem with these and most other models is that a significant amount of knowledge about the relation between features and objects must be explicitly given for the method to work.

In this paper we propose a novel learning method for active vision. The method is based on a Boltzmann Machine (BM) neural network [4, 3]. The network consists of several layers: an input layer encodes direct sensory data or elementary features calculated from these data. The output layer encodes the possible objects. Between these two layers, there may be several hidden layers, connected by adaptive weights. The BM is trained to learn the relation between features and objects.

A distinguishing feature of the BM is that not only the conditional probabilities of the different classes given the feature values is learned, but also the probabilities of the feature values themselves. Thus the probability in the joint input-output space is learned. Previously, we have used this to develop learning rules for data containing missing values. For each training pattern, the neurons with missing values are treated as if they were hidden neurons and are integrated over [5, 6]. After training, the network contains a model of both the probabilities of classes given the feature values, and of the features themselves.

The active vision paradigm works as follows. Initially, no feature values are known, i.e. all values are missing. The probabilities of the different objects in the output layer and of the missing features in the input layer are given by the BM. Subsequently, the missing feature is selected whose value will, in expectation, yield the most information. Information is measured in terms of the entropy over the space of object probabilities in the output layer. Then, the value of this feature is measured. This may require a camera or robot movement, but will be left unspecified here. With this value, the probabilities of the different objects in the output layer and of the missing features in the input layer are recalculated. The process is repeated until sufficient certainty is obtained about the identity of the object, or until all features have been measured.

The most attractive feature of this approach is that the knowledge on which the active vision is based is learned, and not preprogrammed, as in many other approaches. Also, as we will show, the method performs quite well in the presence of noise, which is not the case for other feature selection methods such as ID3 [8].

[1] Real World Computing Partnership
[2] Foundation for Neural Networks

2 Boltzmann Machines

Boltzmann Machines are stochastic networks. The neurons can be in two states $\sigma_i = \pm 1$, but also continuous neuron values are possible [5, 7]. Using Glauber dynamics, neurons are randomly selected and updated one at the time at discrete time steps. After long times, the probability to observe the network in a state $\vec{s}$ becomes independent of time. When the weights of the network are chosen symmetrically, this time independent equilibrium distribution is the Boltzmann distribution and is given by

$$p(\vec{s}) = \frac{1}{Z} \exp\{\beta \sum_{i,j}^{n} w_{ij} s_i s_j\}$$

$$Z = \sum_{\vec{s}} \exp\{\beta \sum_{i,j}^{n} w_{ij} s_i s_j\}$$

w_{ij} are the weights in the network, w_{0j} are neuron thresholds and n is the number of neurons. Since only the equilibrium distribution is needed, it could be calculated in principle using the above formulae. However, this is generally not computationally feasible because of the sum over an exponential number of terms in Z.

For joint probability estimation, Boltzmann Machine learning consists of minimization of the Kullback divergence between the Boltzmann distribution and the probability distribution over the training data $q(x, y)$. x and y refer to input and output neurons/data, respectively. The learning rules are given by gradient descent on the Kullback divergence [1]. Calculation of the gradient involves the calculation of Z, which involves a number of terms which grows exponential with n. Therefore, the general Boltzmann Machine architecture requires too much time for training for practical problems [3]. However, by introducing lateral inhibition in the network, the number of states $\vec{s}$ that contributes to Z can be reduced to a polynomial number in n, thereby yielding learning rules that can be used for practical problems.

The proposed Boltzmann Machine architectures make use of this fact and are given in Fig. 1. For the network without hidden units the optimal weights, and thus also the probability to observe a state, can be computed directly from the training data, as is shown in the Appendix. The learning rules for the network with hidden units can be accelerated with EM-learning and are given in [7].

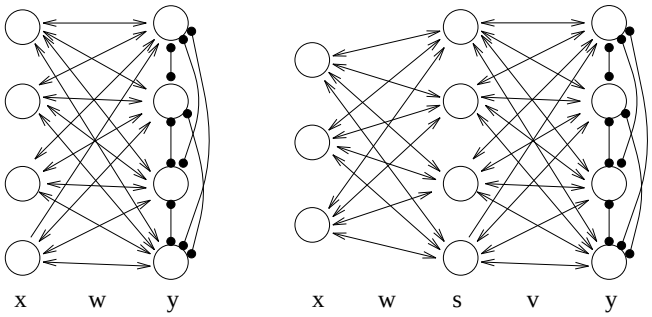

Figure 1: The architecture of Boltzmann machines used for learning active vision. Inputs x_i are continuous or binary valued. Hidden units s_j and output units y_k are 0 or 1. Lateral inhibition in the output layer ensures that only states with $\sum_k y_k = 1$ have finite probability of occurrence. a) Network without hidden units. Optimal network weights can be obtained without iterative learning. b) Network with hidden units. Lateral inhibition in the hidden layer (not shown) ensures that only states with $\sum_j s_j = 1$ have finite probability of occurrence.

3 Active Decision

We use training data (possibly with missing values) to learn the joint probability $p(x, k)$ with the BM. From this joint probability we can compute any conditional probability such as the class probabilities or the individual feature probabilities given some data x_c: $p(k|x_c)$ and $p(x_i|x_c)$, respectively.

Let us assume that during the process of decision making we know some of the input features. Given this vector x_c we can compute the current entropy on the classes:

$$E_{x_c} = -\sum_{k} p(k|x_c) \log p(k|x_c).$$

For every still unknown feature x_i we can compute the expected value of the entropy on the classes. The expectation is the average over the different values that this feature can take upon measurement, weighted with the probabilities given by the BM:

$$\langle E \rangle_i = \int p(x_i|x_c) E_{x_c,x_i} \, dx_i .$$

The feature i^* which gives the smallest expected entropy is the feature with the highest expected information, and it will thus be most advantageous to measure *that* feature[3] This process may be repeated until the current entropy has, for instance, dropped below a threshold, giving a sequence of most informative features.

4 Numerical results

The method is demonstrated on a simple recognition task where local features can be measured across the visual field. The data set consisted of images (12 by 10 pixels) of letters. The model is trained to represent the relation between local feature values (in this case binary pixels) and a number of objects. We used the network in Fig. 1 without hidden units. After training, it is demonstrated how local sensor information is combined with learned representation about possible objects to generate a sequence of 'saccadic' movements, such that the total number of measurements to identify the object is minimized.

In the first numerical simulation, we used one image of each letter as a training set. The data are shown in Fig. 2a. Subsequently, for each of the images, we let the network determine the sequence of saccadic movements until full recognition. In Fig. 3a, we show the entropy as a function of the number of known features, averaged over the 26 images. In this case the network performs binary search, thus requiring either 4 or 5 steps. For comparison, the average entropy for random bit selection is shown.

In the second simulation, we used 10 distorted images for each letter as shown in Fig. 2b. Half of the data was used for training the network. The results are shown in Fig. 3b.

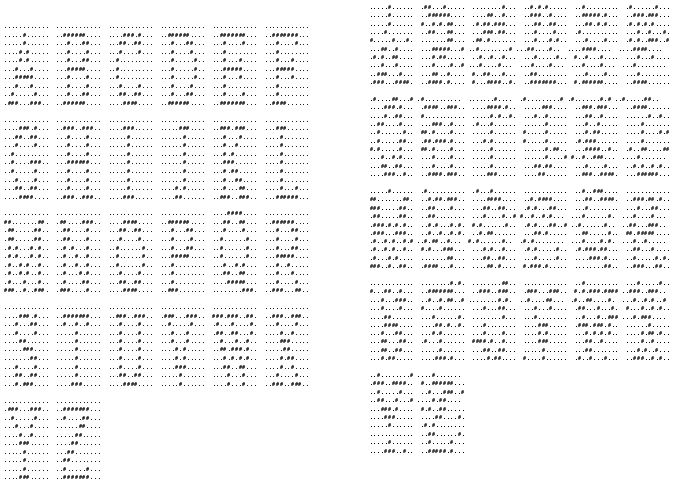

Figure 2: Letters used for active vision. a) Training set without noise. b) From each letter, 10 noisy versions were generated. Each bit was flipped with probability 0.1.

5 Discussion

In this paper we have introduced a new type of problem which we call active decision. It consists of finding the optimal subsequent action, based both on partial observation and on previously learned knowledge. We have proposed a method for solution, based on Boltzmann Machine learning of joint input-output probabilities and on an entropy minimization criterion.

This method is of relevance for modeling the cognitive aspects underlying saccades. It is well known, that saccadic eye movements during recognition of human faces concentrate on certain 'informative' areas such as eyes and mouth. Clearly, our method could be directly applied to this case. After training on a number of

[3]One can also define i^* such as to maximize the Kullback divergence between $p(k|x_c)$ and $p(k|x_c, x_i)$. When the Kullback divergence is defined as $K = \sum_k p(k|x_c, x_i) \log \left(\frac{p(k|x_c,x_i)}{p(k|x_c)} \right)$ this yields an identical definition for i^*. When the Kullback divergence is defined as $K = \sum_k p(k|x_c) \log \left(\frac{p(k|x_c)}{p(k|x_c,x_i)} \right)$ the definition of i^* is different.

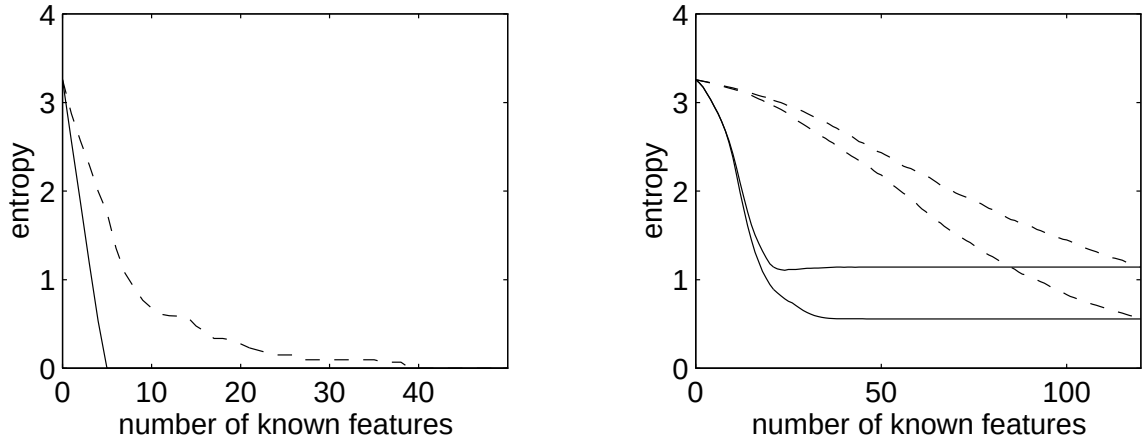

Figure 3: The entropy as a function of the number of measured features. The continuous line corresponds to active decision. The dashed line corresponds to the random strategy. a) Without noise. With active decision only 4 or 5 features are needed for certain classification. With random decision about 40 (of the 120) features are needed. b) With noise ($P_{\text{flip}} = 0.1$). The bottom of each set of lines refers to the average entropy on the training set. The top line refers to the test set. Because of the noise classification is never certain, not even with all features available. However, the smart strategy has reached its final level of uncertainty with approximately 25 known features (test set).

faces, it should be expected that the eyes and mouth areas contain the most discriminative features between people. Therefore, active decision will select features in these regions first. In a similar manner, the proposed method could be used for moving camera systems, providing a basic mechanism for artificial attention.

The proposed method yields a sequence of features. If we assume that the complete data vector is known this sequence gets a different interpretation; it is the order in which features should be examined to obtain reliable classification in as few steps as possible. In other words, our network represents a decision tree. For binary (or nominal) vectors this decision tree is analogous to the decision tree generated by ID3; a binary tree with in every node a boolean test on one of the features. As opposed to ID3 our decision tree is represented in a distributed way by a neural network. The main advantage of our method is that it can still perform sensible feature selection on noisy data. For continuous-valued vectors the decision process can not be visualized as a tree any more and the similarity with ID3 no longer holds.

The proposed method relies heavily on having a good model of the joint probability. This requires a good model of the conditional probability $p(k|x)$ and of $p(x)$. The former is much easier to obtain than the latter, because the output space is usually much smaller than the input space. So far, we have only used Boltzmann Machines without hidden units, which give quite poor representations of the input probability distribution. This can be easily understood because the class conditional probabilities $p(x|k) = \Pi_i p(x_i|k)$, i.e. for each class, the different inputs are independent. Nevertheless, quite good results can be obtained with these simple networks. In general, however, hidden units will be required. This has no consequence for the method, but clearly affects the computation time for the learning phase. This can be somewhat remedied by training networks for each class separately. For each class k, $p_k(x)$ will be the distribution of the input patterns given by the Boltzmann Machine. The total model will be $p(x,k) = B_k p_k(x)$, with B_k as defined in the Appendix.

An advantage for using the Boltzmann Machine is that it can be trained with incomplete data, i.e., training data with missing features. We have not used this aspect in this paper, but one could easily imagine situations where this would be advantageous. One example is the case of medical diagnosis. Given some patient data, a physician must assess the probability of several diseases. To rule out some of these hypotheses, the physician must request several lab tests. The question is which tests will be most effective for the decision process. Here learning with missing values becomes crucial, because for each patient with completed diagnosis not all possible test results are known.

References

[1] D. Ackley, G. Hinton, and T. Sejnowski. A learning algorithm for Boltzmann machines. *Cognitive Science*, 9:147–169, 1985.

[2] S.J. Dickinson, H.I. Christensen, J. Tsotsos, and G. Olofsson. Active object recognition integrating attention and viewpoint control. In *Third European Conference on Computer Vision*, volume II, pages 3–14. Springer-Verlag, 1994.

[3] H.J. Kappen. Using boltzmann machines for probability estimation: A general framework for neural network learning. In E.S. Gelsema and L.N. Kanal, editors, *Proc. Pattern Recognition in Practice IV*, pages 299–312, Amsterdam, 1994. Elsevier.

[4] H.J. Kappen. Deterministic learning rules for boltzmann machines. *Neural Networks*, 8:537–548, 1995.

[5] H.J. Kappen and M.J. Nijman. Radial Basis Boltzmann Machines and learning with missing values. In J.G. Taylor, editor, *World Congress on Neural Networks*, Washington, 1995. INNS. Submitted.

[6] M.J. Nijman and H.J. Kappen. Radial Basis Boltzmann Machines and incomplete data. In *ICANN-95*, 1995. Submitted.

[7] M.J. Nijman and H.J. Kappen. Symmetry breaking and training from incomplete data with Radial Basis Boltzmann Machines. *Neural Computation*, 1995. Submitted.

[8] J.R. Quinlan. Learning efficient classification procedures. In R.S. Michalski, J.G. Carbonell, and T.M. Mitchell, editors, *Machine Learning: an artificial intelligence appraoch*, pages 463–482. Palo Alto: Tioga, 1983.

[9] P. Whaite and F.P. Ferrie. Autonomous exploration: Driven by uncertainty. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 10:331–336, 1994. Submitted.

Appendix

For the Boltzmann Machine architecture without hidden units the probability to observe binary pattern x ($x_i \pm 1$) from class k is

$$p(x,k) = \frac{1}{Z} \exp\left(\sum_{i=1}^{n} x_i u_{ik} + \theta_k\right)$$

with

$$\begin{aligned} Z &= \sum_k \sum_x \exp\left(\sum_{i=1}^{n} x_i u_{ik} + \theta_k\right) \\ &= \sum_k e^{\theta_k} \prod_i 2\cosh(u_{ik}) \equiv \sum_k Z_k. \end{aligned}$$

Training a network on P patterns means maximizing the log likelihood of these patterns

$$L = \frac{1}{P} \sum_\mu \ln p(x^\mu, k^\mu).$$

Define $B_{ik} = \frac{1}{P}\sum_\mu x_i^\mu \delta_{kk^\mu}$ and $B_k = \frac{1}{P}\sum_\mu \delta_{kk^\mu}$. It can be shown that the optimal weight configuration (i.e., where $\partial L/\partial u_{ik} = 0$ and $\partial L/\partial \theta_k = 0$) satisfies $Z_k = B_k$ and $\tanh(u_{ik}) = \frac{B_i k}{B_k}$. From this it follows that $p(x,k)$ can be written directly in terms of the data, leading to

$$p(x,k) = \frac{B_k}{2^n} \prod_i \left(1 + x_i \frac{B_{ik}}{B_k}\right).$$

Similarly, for the Boltzmann Machine architecture without hidden units the probability to observe continuous valued pattern x from class k is [3]

$$p(x,k) = \frac{1}{Z} \exp\left(-\beta \|\vec{x} - \vec{w}_k\|^2 + \theta_k\right)$$

with

$$Z = \left(\frac{\pi}{\beta}\right)^{\frac{n}{2}} \sum_k \exp(\theta_k).$$

After training, the equilibrium distribution becomes

$$p(x,k) = \left(\frac{\beta}{\pi}\right)^{\frac{n}{2}} B_k \exp\left(-\beta \|\vec{x} - \vec{w}_k\|^2\right)$$

with $w_{ik} = \frac{B_{ik}}{B_k}$.