Emergence of movement sensitive neurons' properties by learning a sparse code for natural moving images

Rafal Bogacz
Dept. of Computer Science
University of Bristol
Bristol BS8 1UB, U.K.
R.Bogacz@bristol.ac.uk

Malcolm W. Brown
Dept. of Anatomy
University of Bristol
Bristol BS8 1TD, U.K.
M.W.Brown@bristol.ac.uk

Christophe Giraud-Carrier
Dept. of Computer Science
University of Bristol
Bristol BS8 1UB, U.K.
cgc@cs.bris.ac.uk

Abstract

Olshausen & Field demonstrated that a learning algorithm that attempts to generate a sparse code for natural scenes develops a complete family of localised, oriented, bandpass receptive fields, similar to those of 'simple cells' in V1. This paper describes an algorithm which finds a sparse code for sequences of images that preserves information about the input. This algorithm when trained on natural video sequences develops bases representing the movement in particular directions with particular speeds, similar to the receptive fields of the movement-sensitive cells observed in cortical visual areas. Furthermore, in contrast to previous approaches to learning direction selectivity, the timing of neuronal activity encodes the phase of the movement, so the precise timing of spikes is crucially important to the information encoding.

1 Introduction

It was suggested by Barlow [3] that the goal of early sensory processing is to reduce redundancy in sensory information and the activity of sensory neurons encodes independent features. Neural modelling can give some insight into how these neural nets may learn and operate. Atick & Redlich [1] showed that training a neural network on patches of natural images, aiming to remove pair-wise correlation between neuronal responses, results in neurons having centre-surround receptive fields resembling those of retinal ganglion neurons. Olshausen & Field [11,12] demonstrated that a learning algorithm that attempts to generate a sparse code for natural scenes while preserving information about the visual input, develops a complete family of localised, oriented, bandpass receptive fields, similar to those of simple-cells in V1. The activities of the neurons implementing this coding signal the presence of edges, which are basic components of natural images. Olshausen & Field chose their algorithm to create a sparse representation because it possesses a higher degree of statistical independence among its outputs [11]. Similar receptive fields were also obtained by training a neural net so as to make the responses of neurons as independent as possible [4]. Other authors [14,16,5] have shown that direction selectivity of the simple-cells may also emerge from unsupervised

learning. However, there is no agreed way of how the receptive fields of neurons that encode movements are created.

This paper describes an algorithm which finds a sparse code for sequences of images that preserves the critical information about the input. This algorithm, trained on natural video images, develops bases representing movements in particular directions at particular speeds, similar to the receptive fields of the movement-sensitive cells observed in early visual areas [9,2]. The activities of the neurons implementing this encoding signal the presence of edges moving with certain speeds in certain directions, with each neuron having its preferred speed and direction. Furthermore, in contrast to all the previous approaches, the timing of neural activity encodes the movement's phase, so the precise timing of spikes is crucially important for information coding.

The proposed algorithm is an extension of the one proposed by Olshausen & Field. Hence it is a high level algorithm, which cannot be directly implemented in a biologically plausible neural network. However, a plausible neural network performing a similar task can be developed. The proposed algorithm is described in Section 2. Sections 3 and 4 show the methods and the results of simulations. Finally, Section 5 discusses how the algorithm differs from the previous approaches, and the implications of the presented results.

2 Description of the algorithm

Since the proposed algorithm is an extension of the one described by Olshausen & Field [11,12], this section starts with a brief introduction of the main ideas of their algorithm. They assume that an image $\mathbf{x}$ can be represented in terms of a linear superposition of basis functions $\mathbf{A}_i$. For clarity of notation, let us represent both images and bases as vectors created by concatenating rows of pixels as shown in Figure 1, and let each number in the vector describe the brightness of the corresponding pixel. Let the basis functions $\mathbf{A}_i$ form the columns of a matrix $\mathbf{A}$. Let the weighting of the above mentioned linear superposition (which changes from one image to the next) be given by a vector $\mathbf{s}$:

$$\mathbf{x} = \mathbf{A}\,\mathbf{s} \tag{1}$$

The image $\mathbf{x}$ may be encoded, for example using the inverted transformation where it exists. Hence, the image code $\mathbf{s}$ is determined by the choice of basis functions $\mathbf{A}_i$. Olshausen & Field [11,12] try to find bases that result in a code $\mathbf{s}$ that preserves information about the original image $\mathbf{x}$ and that is sparse. Therefore, they minimise the following cost function with respect to $\mathbf{A}$, where λ denotes a constant determining the importance of sparseness [11]:

$$E = -[\text{preserved information in } \mathbf{s} \text{ about } \mathbf{x}] - \lambda[\text{sparseness of } \mathbf{s}] \tag{2}$$

The algorithm proposed in this paper is similar, but it takes into consideration the temporal order of images. Let us divide time into intervals (to be able to treat it as discrete) and denote the image observed at time t and the code generated by $\mathbf{x}^t$ and $\mathbf{s}^t$, respectively. The Olshausen & Field algorithm assumes that image $\mathbf{x}$ is a linear superposition (mixture) of $\mathbf{s}$. By contrast, our algorithm assumes that images are convolved mixtures of $\mathbf{s}$, i.e., $\mathbf{s}^t$ depends not only on $\mathbf{x}^t$ but also on $\mathbf{x}^{t-1}$, $\mathbf{x}^{t-2}$,…, $\mathbf{x}^{t-(T-1)}$ (i.e. $\mathbf{s}^t$ depends on T preceding $\mathbf{x}^t$). Therefore, each basis function may also be

Figure 1: Representing images as vectors.

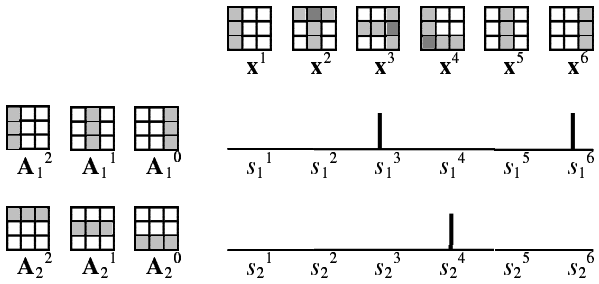

Figure 2: Encoding of an image sequence. In the example, there are two basis functions, each described by $T = 3$ vectors. The first basis encodes movement to the right, the second encodes movement down. A sequence $\mathbf{x}$ of 6 images is shown on the top and the corresponding code $\mathbf{s}$ below. A "spike" over a coefficient s_i^t denotes that $s_i^t = 1$, the absence of a "spike" denotes $s_i^t = 0$.

represented as a sequence of vectors $\mathbf{A}_i^0$, $\mathbf{A}_i^1$, … , $\mathbf{A}_i^{T-1}$ (corresponding to a sequence of images). These vectors create columns of the mixing matrices $\mathbf{A}^0$, $\mathbf{A}^1$, … , $\mathbf{A}^{T-1}$. Each coefficient s_i^t describes how strongly the basis function $\mathbf{A}_i$ is present in the last T images. This relationship is illustrated in Figure 2 and is expressed by Equation 3.

$$\mathbf{x}^t = \sum_{f=0}^{T-1} \mathbf{A}^f \mathbf{s}^{f+t} \tag{3}$$

In the proposed algorithm, the basis functions $\mathbf{A}$ are also found by optimising the cost function of Equation 2. The detailed method of this minimisation is described below, and this paragraph gives its overview. In each optimisation step, a sequence $\mathbf{x}$ of P image patches is selected from a random position in the video sequence ($P \geq 2T$). Each of the optimisation steps consists of two operations. Firstly, the sequence of coefficient vectors $\mathbf{s}$ which minimises the cost function E for the images $\mathbf{x}$ is found. Secondly, the basis matrices $\mathbf{A}$ are modified in the direction opposite to the gradient of E over $\mathbf{A}$, thus minimising the cost function. These two operations are repeated for different sequences of image patches.

In Equation 2, the term "preserved information in $\mathbf{s}$ about $\mathbf{x}$" expresses how well $\mathbf{x}$ may be reconstructed on the basis of $\mathbf{s}$. In particular, it is defined as the negative of the square of the reconstruction error. The reconstruction error is the difference between the original image sequence $\mathbf{x}$ and the sequence of images $\mathbf{r}$ reconstructed from $\mathbf{s}$. The sequence $\mathbf{r}$ may be reconstructed from $\mathbf{s}$ in the following way:

$$\mathbf{r}^t = \sum_{f=0}^{T-1} \mathbf{A}^f \mathbf{s}^{f+t} \tag{4}$$

The precise definition of the cost function is then given by:

$$E = \sum_{t=T}^{P-T+1} \sum_j \left(x_j^t - r_j^t\right)^2 + \lambda \sum_{t=T}^{P} \sum_i C\left(\frac{s_i^t}{\sigma}\right) \tag{5}$$

In Equation 5, C is a nonlinear function, and σ is a scaling constant. Images at the start and end of the sequence (e.g., $\mathbf{x}^1$, $\mathbf{x}^P$) may share some bases with images not in the sequence (e.g., $\mathbf{x}^0$, $\mathbf{x}^{-1}$, $\mathbf{x}^{P+1}$). To avoid this problem, only the middle images are reconstructed and only for them is the reconstruction error computed in the cost function. In particular, only images from T to P-T+1 are reconstructed – since the assumed length of the bases is T, those images contain only the bases whose other

parts are also contained in the sequence. Since only images from T to P-T+1 are reconstructed, it is clear from Equation 4, that only coefficients $\mathbf{s}^T$ to $\mathbf{s}^P$ need to be found. These considerations explain the limits of the outer summations in both terms of Equation 5.

For each image sequence, in the first operation, the coefficients $\mathbf{s}^T$, $\mathbf{s}^{T+1}$, ... , $\mathbf{s}^P$ minimising E are found using an optimisation method. Minus the gradient of E over $\mathbf{s}$ is given by:

$$-\frac{\partial E}{\partial s_i^f} = 2\sum_t \sum_j \left(x_j^t - r_j^t\right) A_{ij}^{t-f} - \frac{\lambda}{\sigma} C'\left(\frac{s_i^f}{\sigma}\right) \tag{6}$$

In the second operation, the bases $\mathbf{A}$ are modified so as to minimise E:

$$\Delta A_{ij}^f = -\frac{\eta}{2}\frac{\partial E}{\partial A_{ij}^f} = \eta \sum_t \left(x_j^t - r_j^t\right) s_i^{t+f} \tag{7}$$

In equation 7, η denotes the learning rate. The vector length of each basis function $\mathbf{A}_i$ is adapted over time so as to maintain equal variance on each coefficient $\mathbf{s}$, in exactly the same way as described in [12].

3 Methods of simulations

The proposed algorithm was implemented in Matlab except for finding $\mathbf{s}$ minimising E, which was implemented in C++, using the conjugate gradient method for the sake of speed. In the implementation, the original codes of Olshausen & Field were used and modified (downloaded from http://redwood.ucdavis.edu/bruno/sparsenet.html).

Many parameters of the proposed algorithm were taken from [11]. In particular, $C(x) = \ln(1+x^2)$, σ is the standard deviation of pixels' colours in the images, λ is set up such that $\lambda/\sigma = 0.14$, and $\eta = 1$. $\Delta\mathbf{A}$ is averaged over 100 image sequences, and hence the bases $\mathbf{A}$ are updated with the average of $\Delta\mathbf{A}$ every 100 optimisation steps. The length of an image sequence P is set up such that $P = 3T$.

The proposed algorithm was tested on two types of video sequences: 'toy' problems and natural video sequences. Each of the toy sequences consisted of 10 frames – 100×100 pixels. In the sequence, there were 20 moving lines. Each line was either horizontal or vertical and 1 pixel thick. Each line was either black or white, which corresponded to positive or negative values of the elements of $\mathbf{x}$ vectors (the grey background corresponded to zero). Each horizontal line moved up or down, each vertical – left or right, with the speed of one pixel per frame.

Then the algorithm was tested on five natural video sequences showing moving people or animals. In each optimisation step, a sequence of image patches was selected from a randomly chosen video. The video sequences were preprocessed. First, to remove the static aspect of the images, from each frame the previous one was subtracted, i.e., each image encoded the difference between two successive frames of the video. This simple operation reduces redundancy in data since the corresponding pixels in the successive frames tend to have similar colours. An analogous operation may be performed by the retina, since the ganglion cells typically respond to the changes in light intensity [10].

Then, to remove the pair-wise correlation between pixels of the same frame, Zero-phase Component Analysis (ZCA) [4] was applied to each of the patches from the selected sequence, i.e., $\mathbf{x}^t := \mathbf{W}\,\mathbf{x}^t$, where $\mathbf{W} = \langle \mathbf{x}^t(\mathbf{x}^t)^{\mathrm{T}} \rangle^{-½}$ i.e., $\mathbf{W}$ is equal to the inverted square root of the covariance matrix of $\mathbf{x}$. The filters in $\mathbf{W}$ have centre-surround receptive fields resembling those of retinal ganglion neurons [4].