USENIX
THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

The following paper was originally published in the

Proceedings of the Embedded Systems Workshop

Cambridge, Massachusetts, USA, March 29–31, 1999

Learning in Intelligent Embedded Systems

Daniel D. Lee
Bell Laboratories - Lucent Technologies

H. Sebastian Seung
Massachusetts Institute of Technology

For more information about the USENIX Association:
Phone: 1 510 528 8649 FAX: 1 510 548 5738
Email: office@usenix.org WWW: http://www.usenix.org

Learning in Intelligent Embedded Systems

Daniel D. Lee
Bell Laboratories
Lucent Technologies
Murray Hill, NJ 07974
email: ddlee@lucent.com

H. Sebastian Seung
Dept. of Brain and Cognitive Sciences
Massachusetts Institute of Technology
Cambridge, MA 02138
email: seung@mit.edu

Abstract

Information processing capabilities of embedded systems presently lack the robustness and rich complexity found in biological systems. Endowing artificial systems with the ability to adapt to changing conditions requires algorithms that can rapidly learn from examples. We demonstrate the application of one such learning algorithm on an inexpensive robot constructed to perform simple sensorimotor tasks. The robot learns to track a particular object by discovering the salient visual and auditory cues unique to that object. The system uses a convolutional neural network to combine color, luminance, motion, and auditory information. The weights of the networks are adjusted using feedback from a teacher to reflect the reliability of the various input channels in the surrounding environment. We also discuss how unsupervised learning can discover features in data without external interaction. An unsupervised algorithm based upon nonnegative matrix factorization is able to automatically learn the different parts of objects. Such a parts-based representation of data is crucial for robust object recognition.

1 Introduction

The information processing capabilities embedded in systems today are extremely unreliable when operated in conditions that fall outside of their narrow design specifications. For example, the best present-day speech recognition systems will fail if a different microphone is substituted, or if the speaker has a sore throat [Rabiner]. On the other hand, biological systems are extremely robust to environmental changes when compared with artificial systems. The success of biological information processing systems lies in their ability to accommodate changes at all processing levels, from low-level sensor modalities to high-level computational algorithms and architectures. In order for artificial systems to truly become ubiquitous in the future, they will need to incorporate this ability to quickly and robustly adapt to changes.

In these proceedings, we present some of our work on algorithms that allow a system to learn from prior experience and adapt its behavior accordingly. We demonstrate these algorithms on a small quadruped robot that we have constructed to perform various kinds of sensorimotor tasks. In particular, we show how the robot learns to track a novel object by rapidly changing the weights of a convolutional neural network which processes the color, luminance, motion, and audio signals. Based upon the reliability of the different input channels, the system is able to discover the most salient visual and auditory cues unique to that object.

The training of the robot is done online in real time using supervisory signals from a teacher. We also explore how the robot can learn robust cues for object recognition without any supervision. This form of unsupervised learning is essential for adaptation in situations where there is little user interaction. We show how a simple algorithm can automatically learn to segment high-dimensional input data into features that correspond to functionally relevant parts [Palmer]. Our learning al-

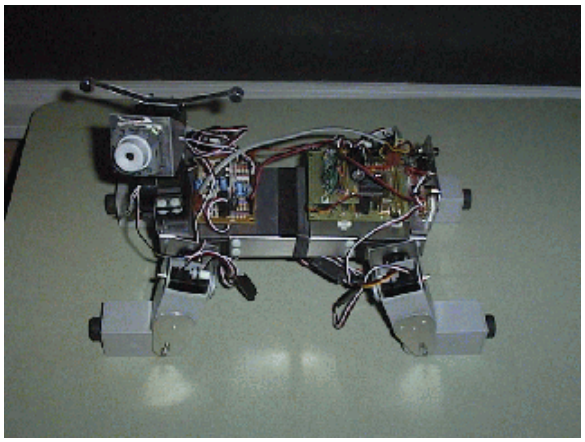

Figure 1: Photograph showing a small quadruped mobile robot that we have constructed to demonstrate our learning algorithms on sensorimotor tasks.

gorithm incorporates nonnegativity constraints that are similar to those found in biological neural networks. This enables our algorithm to learn parts as features by modeling positive coactivation in the inputs. Such a parts-based representation is valuable because it is invariant to perturbations or occlusions that affect localized regions of the input space.

2 Artificial sensorimotor system

In order to demonstrate how learning algorithms can be applied to improve the information processing capabilities of systems in real environments, we have constructed a small quadruped robot that is able to perform various simple sensorimotor tasks. Figure 1 shows a picture of our robot, which is approximately 25 cm in length and about 1 kg in weight, powered by a 9.6V rechargeable battery pack. It contains 14 hobby servo motors to provide three degrees of freedom for each of the four legs, as well as two degrees of freedom for the head. Attached to the head assembly is a small board level CCD camera that acts as a single eye, as well as two directional electret microphones for ears. The two head servo motors allow the camera to rapidly pan and tilt over a wide range of viewing angles. In addition to the visual and auditory inputs, a two-axis gyroscopic rate sensor is used to provide vestibular input for stabilization tasks.

An onboard Motorola 68HC11 microprocessor converts and processes the gyroscopic inputs, and provides auditory feedback by producing sounds in a small speaker. It also generates the timing signals for the 14 motors,

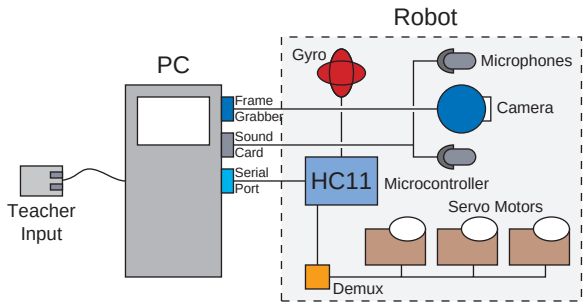

Figure 2: Schematic diagram showing the various hardware components of the system.

which are demultiplexed to the separate motors using shift registers. Because the microprocessor is not powerful enough to process the input signals from the CCD camera and microphones, the video and audio signals are currently sent to an offboard computer for processing as shown in Figure 2. The computer is then able to direct the onboard microprocessor to generate the appropriate motor behavior via a serial link.

The neural network algorithms for visual and auditory processing and control are implemented using Matlab for rapid development purposes. Real-time acquisition of the video and audio signals into Matlab variables is accomplished with custom-written drivers running under either the Microsoft Windows or the Linux operating system. Because the hardware was built using cheap, commercially available components (total cost of the system $< \$700$), the control software needs to be robust against the large sources of noise and drift in the hardware components. The following sections explain how online learning algorithms can compensate for limitations in the hardware and changes in environmental conditions.

3 Tracking application

Let us consider the problem of trying to program the robot to watch and follow with its head as someone walks around the room. This type of active perception is critical for proper functioning of the human visual system [Yarbus]. Because human retinas have small, high resolution foveal regions surrounded by visual fields of relatively much lower resolution, the eyes need to be making constant movements in order to keep objects focused on the foveas. Thus, the role of the human oculomotor system is to direct relevant objects in the visual world onto these high resolution areas. To accomplish

this task, biology has evolved many complex neural circuits to control eye movements. Some examples include the saccadic system which rapidly acquires new objects, and the smooth pursuit system which tracks slowly moving objects [Horiuchi, Rao].

In analogy with the biological system, our robot also needs to first determine which feature in its visual field is most relevant and then direct its gaze towards that object. As the target person moves about, the robot's tracking system will attempt to stabilize the person's image in the center of the field of view. Unfortunately, the physical performance of our hardware system is very lacking compared to biology. A fixed lens is used on the camera which gives it roughly a 65° horizontal and 48° vertical field of view. With the video digitized at frames of 120×160 pixels, this corresponds to a little less than half a degree of angular resolution, which is 25 times worse than human foveal acuity. Also, the maximum saccadic velocity of the servo motors is about 300 deg/sec, which is twice as slow as human eye movements. The tracking algorithm needs to be able to overcome these physical limitations.

A naive heuristic algorithm for the robot to track someone is to simply have it follow the largest moving object in the image. However, such a system is easily fooled if there are multiple objects moving in the room. More sophisticated algorithms [Darrell, Petajan] have proposed locating human heads by employing rules such as flesh color detection [Yang], or matching ellipses to head outlines [Eleftheriadis]. Another approach is to use the directional microphones to locate any person who is speaking using audio cues [Bregman]. But all these algorithms will fail in situations for which they were not designed. Such predetermined tracking algorithms tend to be very brittle and break when the surrounding environment is highly variable.

Recently, a potentially more robust method has been demonstrated for head detection using neural networks to learn the appropriate grayscale features of a face [Sung, Nowlan, Rowley]. Although these networks can very accurately detect faces in images, they need to be trained on a very large set of labelled face and non-face images. Since these networks are typically trained in batch mode on a preset ensemble of faces, they also do not learn to discriminate one person from another.

For our robot, we use a convolutional neural network that rapidly learns to locate and track a particular person's head. The system learns to do this task in real time with online supervisory signals. The network architecture integrates multimodal information in a natural way,

Figure 3: Preprocessing of the video images. Luminance, chromatic and motion information are separately represented in the Y, U, V, D channels at multiple resolutions.

and the fast online adaptation of the network's weights allows it to adjust the relative importance of the inputs depending upon the current environment. This enables the system to robustly track a person even in a cluttered background in the presence of other moving objects.

3.1 Video preprocessing

The raw video signal from the robot first needs to be preprocessed before it can be input to the convolutional neural network. The composite video signal from the CCD camera is digitized with a video capture board into a time series of raw 120×160 RGB images as shown in Figure 3. Each RGB color image is then converted into its YUV representation, and a difference (D) image is also computed from the absolute value of the difference between consecutive frames. The Y component represents the luminance or grayscale information in the image, while the U and V channels contain the chromatic or color data. Motion information in the video stream is represented in the D image where moving objects appear highlighted.

At each time step, the four YUVD images are then subsampled successively to yield representations at lower and lower resolutions. The resulting "image pyramids" allow the network to achieve recognition invariance across many different image scales without having to train separate neural networks for each resolution. Instead, a single neural network with the same set of weights is simultaneously run across the different resolutions, and the maximally active resolution and position

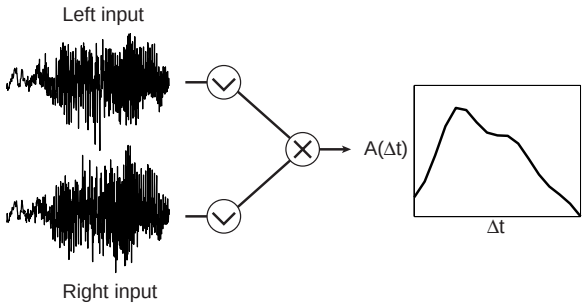

Figure 4: Processing of the audio waveforms to yield a topological map of audio sources as a function of time delay.

is selected.

3.2 Audio preprocessing

Animals are able to robustly localize audio sources using a variety of auditory cues [Bregman]. Experiments have shown that one of the most important cues for azimuthal position determination is interaural time difference, corresponding to the slight delay it takes for a sound to reach the different ears [Konishi]. Similarly, the two microphones on the head of our robot are separated horizontally in space to optimize the interaural difference between their audio signals. This information is then combined with visual cues by the convolutional neural network to determine the overall saliency of the different locations in its field of view. In order for the neural network to process the auditory information in the same manner as it does visual information, the raw audio data has to first be converted into an auditory space map as depicted in Figure 4.

The audio signal from the left and right microphones are first digitized at 16000 Hz in the sound card. These waveforms are then passed through a rectification nonlinearity to emphasize the envelope of the sound waveforms present in the recordings. The resulting signals are then cross-correlated, giving the following time correlation function:

$$A(\Delta t) = \sum_t |x_L(t)||x_R(t+\Delta t)| \qquad (1)$$

where $x_L(t)$ and $x_R(t)$ are the audio inputs from the left and right microphones. Ambient background noises are attenuated by subtracting out the mean of this correlation function. Different values of the time delay Δt correspond to varying azimuthal positions of auditory sources, assuming their elevation angle is near the horizontal plane of the microphones. Thus, the cross-correlation indicates the likelihood that an auditory source is located at the various azimuthal positions. This correlation function can be considered a one-dimensional auditory space map of the environment.

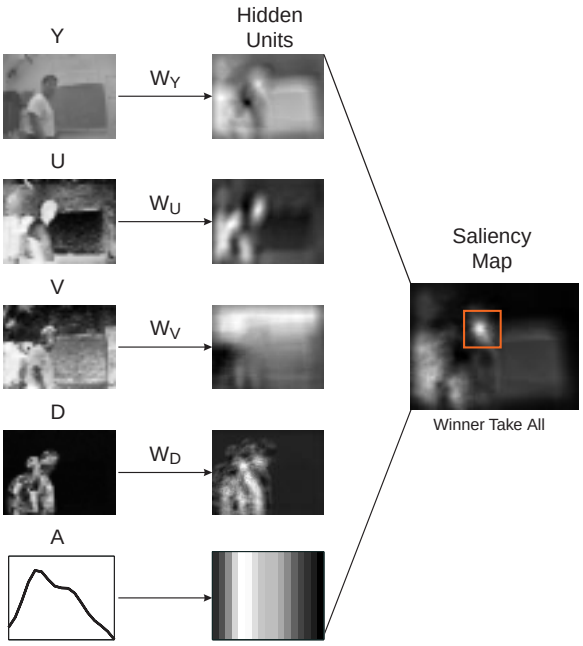

Figure 5: Neural network uses a convolutional architecture to integrate the different sources of information and determine the maximally salient object.

4 Supervised learning

4.1 Neural Network Architecture

Our tracking algorithm uses the convolutional neural network architecture shown in Figure 5 to locate the salient objects in its visual and auditory fields. The YUVD input images are filtered with separate 16×16 kernels, denoted by W_Y, W_U, W_V, and W_D respectively. This results in the filtered images $\bar{Y^s}$, $\bar{U^s}$, $\bar{V^s}$, $\bar{D^s}$:

$$\begin{aligned} \bar{Z^s}(i,j) &= W_Z \circ Z^s \\ &= \sum_{i',j'} W_Z(i',j')\, Z^s(i+i', j+j') \end{aligned} \qquad (2)$$

where s denotes the scale resolution of the inputs, and Z represents any one of the Y, U, V, or D channels. These filtered images correspond to a single layer of hidden units in the neural network. The hidden units are then combined with the one-dimenional auditory correlation function $A(j)$ to form the saliency map X^s in the following manner:

$$\begin{aligned} X^s(i,j) &= c_Y\, g[\bar{Y}^s(i,j)] + c_U\, g[\bar{U}^s(i,j)] + \\ &\quad c_V\, g[\bar{V}^s(i,j)] + c_D\, g[\bar{D}^s(i,j)] + \\ &\quad c_A\, g[A(j)] + c_0 \end{aligned} \tag{3}$$

where the sigmoidal nonlinearity is given by $g(x) = \tanh(x)$. Thus, the saliency X^s is computed on a pixel-by-pixel basis using a nonlinear combination of hidden units. The relative importance of the different luminance, chromatic, motion, and auditory channels in the overall saliency of an object is given by the scalar variables c_Y, c_U, c_V, c_D, and c_A.

With the bias term c_0, the function $g[X^s(i,j)]$ may be interpreted as the relative probability that the tracked object appears in location (i,j) at input resolution s. The final output of the neural network is then determined in a competitive manner by finding the location (i_m, j_m) and scale s_m of the best possible match:

$$g[X_m] = g[X^{s_m}(i_m, j_m)] = \max_{i,j,s} g[X^s(i,j)]. \tag{4}$$

After processing the visual and auditory inputs in this manner, head movements are generated in order to keep the maximally salient object located near the center of the field of view.

4.2 Adaptation

Our robot learns to track a specific target using supervisory feedback from a teacher. During training, the teacher watches the robot's video input and corrects its behavior whenever it makes a mistake. The teacher corrects the robot by indicating where the target is located in the visual field using a graphical user interface. This action prompts the learning algorithm to adjust the weights of the convolutional neural network in order to better track the object at that location.

The algorithm uses the supervisory signals to adjust the kernels W_Z and scalar weights c_Z of the neural network. The neural network is updated whenever the maximally salient location of the neural network (i_m, j_m) does not correspond to the desired object's true position (i_n, j_n) as identified by the teacher. Objective functions proportional to the sum squared error terms at the maximal location and at the new desired location are used for training the network:

$$e_m^2 = |g_m - g[X^{s_m}(i_m, j_m)|^2, \tag{5}$$

$$e_n^2 = \min_s |g_n - g[X^s(i_n, j_n)|^2. \tag{6}$$

For each correction that is provided by the teacher, the algorithm tries to minimize the errors in these objective functions. First, the gradients of Eqs. 5–6 are computed. These gradient terms are then backpropagated through the convolutional network [Nowlan, LeCun], resulting in the following rules for adaptation:

$$\begin{aligned} \Delta c_Z &= \eta\, e_m g'(X_m) g[\bar{Z}(i_m, j_m)] + \\ &\quad \eta\, e_n g'(X_n) g[\bar{Z}(i_n, j_n)], \end{aligned} \tag{7}$$

$$\begin{aligned} \Delta W_Z &= \eta\, e_m g'(X_m) g'(\bar{Z}_m) c_Z Z_m + \\ &\quad \eta\, e_n g'(X_n) g'(\bar{Z}_n) c_Z Z_n. \end{aligned} \tag{8}$$

In typical applications of neural networks where a large set of the training examples can be considered simultaneously, the learning rate η is set to some small positive number. However in our case, it is desirable for the robot to learn to track an object in a new environment as quickly as possible. Thus, rapid adaptation of the weights during even a single training example is needed. A natural way of doing this is to use a fairly large learning rate, and to repeatedly apply the update rules in Eqs. 7–8 until the calculated maximally salient location is very close to the actual desired position.

4.3 Training example

An example of how quickly the robot is able to adapt is given by the learning curve in Figure 6. In this particular example, the system was trained to track the head of one of the authors as he moved around and talked in his office. The weights were first initialized to small random values and the learning parameters were set to $g_m = 0$, $g_n = 1$, and $\eta = 0.1$. The robot was then corrected in an online fashion using supervisory inputs to follow the author's head.

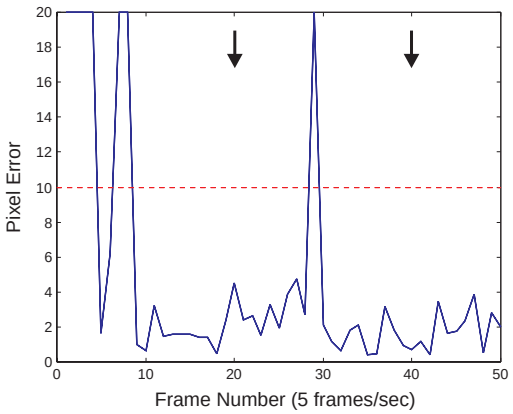

Figure 6: Fast online adaptation of the neural network. The head location error in pixels in a 120×160 image is plotted as a function of frame number (5 frames/sec).

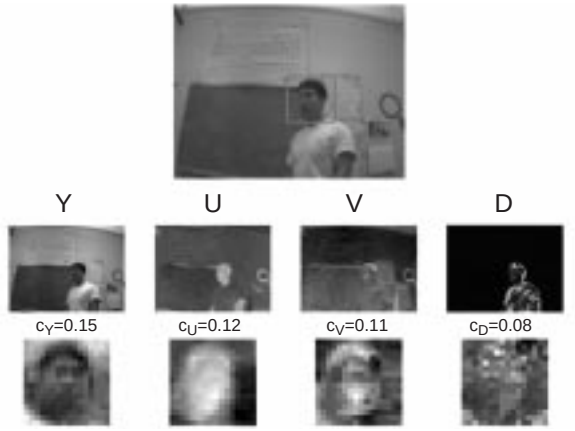

Figure 7: Example showing the inputs and weights used in tracking a head ($c_A = 0.05$). The head position as calculated by the neural network is marked with a box.

After only a few seconds of training at 5 frames/sec (200 ms processing time per image), the system was able to locate the head to within four pixels of accuracy, as determined by hand labelling the video data afterwards. As saccadic eye movements were initiated at the times indicated by the arrows in Figure 6, new environments of the office were sampled and an occasional large error was seen. However, over time these errors were corrected, and the neural network learned to robustly discriminate the head from the office surroundings.

Figure 7 shows the inputs and weights of the network after a minute of training as the author walked around his office. The kernels necessarily appear slightly smeared because they are adapted to be invariant to slight changes in head position, rotation, and scale. But they clearly depict the dark hair, facial features, and skin color of the head. The relative weighting ($c_Y, c_U, c_V > c_D, c_A$) of the different input channels shows that the luminance and color information are the most reliable for tracking the head. This is probably because the presence of other moving body parts and external noise sources in the office made the motion and auditory channels relatively unreliable.

More complicated neural network architectures could be used for combining the different sensory inputs to achieve better tracking performance. However, this example shows how a simple convolutional network architecture can efficiently integrate the different visual and auditory cues in order to learn how to robustly track an object. Moreover, by using fast online adaptation of the neural network weights, the system is able to rapidly accommodate changing environments.

5 Unsupervised learning

The online learning algorithm described above allows our robot to rapidly adapt and correct mistakes when given supervisory feedback by a teacher. But there are many situations in which it is impossible for any teacher to be present. In these situations, would it still be possible for a system to adapt based upon raw sensory stimuli without being told the appropriate thing to do? This is generically quite a difficult problem, but there are some well-established algorithms that allow the system to continue to adapt even without a teacher. Generally, these unsupervised learning algorithms attempt to extract common sets of features present in the input data. The system can then use these features to reconstruct structured patterns from corrupted input data. This invariance of the system to noise allows for more robust processing in performing recognition tasks.

One commonly used technique for building invariances into systems is to project the input data onto a few representative directions known as the principal components. This procedure called principal components analysis (PCA) has historically found widespread use in data compression, modeling, and classification systems [Jolliffe, Turk]. For an example of its application, consider the bottom portion of Figure 8. Here, a set of images of handwritten "two"'s has been analyzed using PCA. The images are taken from the Buffalo Zip Code database [LeCun], and can formally be described by a $n \times m$ matrix X of pixel values. Each of the $m = 731$ grayscale images is preprocessed to roughly center and align it within a 16×16 grid. The pixels are scaled such that white is equal to zero and black is equal to one. The

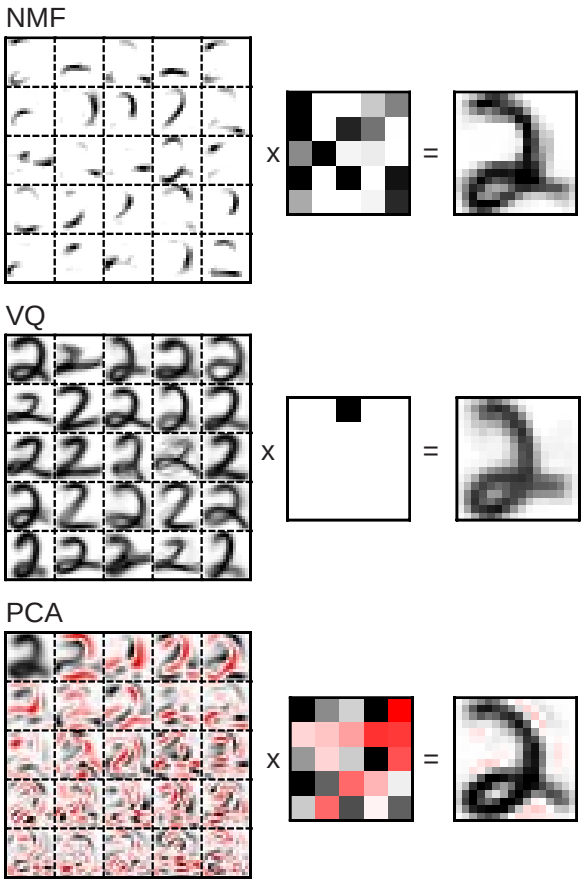

Figure 8: Three different unsupervised learning techniques applied to handwritten images of the digit "two." NMF learns a representation based upon the strokes of the digits, while VQ and PCA learn holistic representations. For each algorithm, the 25 columns of basis W are depicted as a 5×5 montage on the left, with their corresponding activations V in the middle. The reconstruction of the digit given by the product WV is displayed on the right of the figure.

$n = 256$ pixel values are then arranged into a column of X, with each handwritten image forming a separate column. PCA factorizes the matrix X into the approximate form $X \approx WV$ where the matrix factors W and V are $n \times r$ and $r \times m$ in size, respectively. The columns of W correspond to an orthogonal basis set which can be used to reconstruct the original images in X using the coefficients in V. The left side of Figure 8 displays the $r = 25$ different columns of W, which are then multiplied by a particular set of coefficients in V to form the reconstruction image on the right.

PCA is able to efficiently reconstruct the original data X using the limited number r of bases because it uses a representation that is distributed across all of the matrix factors. Some of the bases in W resemble common distortions of "two"'s such as translations and rotations, so the PCA representation is able to capture this global variability in the data. However, this representation utilizes both positive and negative combinations of all the available bases in W. This implies that the reconstruction involves finely tuned cancellations among the different bases, so it is difficult to visualize exactly how most of the columns in W contribute to a robust representation.

In contrast, vector quantization (VQ) is another unsupervised learning technique that categorizes the input data in terms of prototypes rather than orthogonal basis vectors. VQ may be thought of as applying a quantizing error correction to the data in order to remove small perturbations in the input. Formally, it can again be described as an approximate matrix factorization $X \approx WV$ where the columns in matrix V of activations are constrained to be unary (exactly one element in each column is equal to one, all the other elements are zero). In the middle portion of Figure 8, the effect of this constraint can be seen as forcing the VQ factorization to learn various templates of different types of "two"'s. In this case, reconstruction merely involves choosing the most similar prototype, and the representation is extremely sparse since only a single element of V is active.

The top portion of Figure 8 shows our new matrix factorization algorithm that decomposes the images into their representative parts. It achieves this by utilizing nonnegativity as a constraint on the components of the matrix factors W and V [Lee]. As seen in the figure, nonnegative matrix factorization (NMF) learns to decompose the image data into their constituent parts, corresponding to the different strokes of a "two." To approximate a particular two, the appropriate strokes are summed together to form the reconstruction. This is in sharp contrast to the holistic features found by PCA and VQ. Note that NMF finds a decomposition into localized parts even though no prior information about the topology of the images was built into the input data, i.e. the rows of matrix X could have been arbitrarily scrambled and the NMF algorithm would have yielded the same results.

NMF also combines some of the representational advantages of PCA and VQ. Since many of the parts are used to form the reconstruction, NMF has the combinatorial expressiveness of a distributed representation. But because the nonzero elements of W and V are all positive, NMF allows only additive combinations. So unlike PCA, no cancellations can occur. On the other hand, the nonnegativity constraint causes almost half of the coefficients in V to go to zero, resulting in a sparse coding of the input. Thus, NMF learns a parts-based repre-

$$W_{ij} \leftarrow W_{ij} \sum_k \frac{X_{ik}}{(WV)_{ik}} V_{jk}$$

$$W_{ij} \leftarrow \frac{W_{ij}}{\sum_k W_{kj}}$$

$$V_{ij} \leftarrow V_{ij} \sum_k \frac{X_{kj}}{(WV)_{kj}} W_{ki}$$

Figure 9: Multiplicative update rules for nonnegative matrix factorization. The accuracy of the current approximation $X \approx WV$ enters the learning through the quotient $X_{ik}/(WV)_{ik}$. It can be shown that these rules maximize the objective function $\sum_{i=1}^{n} \sum_{k=1}^{m} [X_{ik} \log(WV)_{ik} - (WV)_{ik}]$.

sentation of the data that is sparse as well as distributed [Foldiak, Olshausen].

Analytically, the NMF algorithm can be derived from a probabilistic generative model that incorporates Poisson noise [Hinton]. This model has previously been used in the deconvolution of astronomical images that have been blurred by the atmosphere and the telescope [Richardson, Lucy]. NMF may thus be considered a generalization of this technique to blind deconvolution. By maximizing the likelihood of this probabilistic model, the NMF learning rule for nonnegative matrix factorization is obtained [Dempster, Saul]: Given a data matrix X, the matrix factors W and V are first initialized to random positive values. The factors are then updated using the multiplicative update rules shown in Figure 9, which guarantees that the likelihood of the data X is increased. This procedure is then iterated until an optimal factorization $X \approx WV$ is learned.

6 Discussion

The reason NMF discovers strokes as the functional parts of digits is because the nonnegativity constraints allow it to learn from positive coactivation of image pixels in the data. The NMF algorithm adapts the basis W in order to learn the appropriate coactivations. In constrast, PCA has no constraints on the sign of the activation, and learns simultaneously from both positive and negative activations in the data. It should be noted that biological neural networks may use similar types of constraints to achieve analogous representations. The firing rates of neurons cannot be negative, and the strengths of synapses do not generally change sign. These one-sided constraints could possibly be important in developing the sparsely, distributed coding of sensory input that give rise to robust biological information processing.

The NMF algorithm is also generally applicable to problems outside the domain of image analysis. We have also applied it to the semantic analysis of text documents [Salton, Landauer], as well as to the analysis of routing patterns in data networks. In each of these cases, the algorithm learns to decompose the input data into their constituent parts. This enables any additional processing of the data to be robust against perturbations that can change only a small number of these features [Biederman]. Our current research in this area involves incorporating the NMF parts decomposition of the input data into a hierarchical representation that would be appropriate for high level control of systems such as our robot. Along with future improvements in sensory, motor, communications, and processing hardware, advances in these new learning algorithms will hopefully allow artificial embedded systems to someday exhibit the computational complexity and robustness found in biological systems.

7 Acknowledgments

We acknowledge the support of Bell Laboratories, Lucent Technologies. We also thank M. Fee, A. Jacquin, S. Levinson, E. Petajan, G. Pingali, and L. Saul for helpful discussions.

References

[Biederman] I. Biederman, *Recognition by components: a theory of human image understanding*, Psychological Review 94 (1987) p. 115.

[Bregman] A. Bregman, *Auditory scene analysis: the perceptual organization of sound*, MIT Press, Cambridge, MA (1990).

[Darrell] T. Darrell, P. Maes, B. Blumberg, and A. P. Pentland, *A novel environment for situated vision and behavior*, Proc. IEEE Workshop for Visual Behaviors (1994) p. 68–72.

[Dempster] A. P. Dempster, N. M. Laird, and D. B. Rubin, *Maximum likelihood from incomplete data via*

the EM algorithm, J. Royal Statistical Society 39 (1977) p. 1-38.

[Eleftheriadis] A. Eleftheriadis and A. Jacquin, *Automatic face location detection and tracking for model-assisted coding of video teleconferencing sequences at low bit-rates* Signal Processing: Image Communication 7 (1995) p. 231.

[Foldiak] P. Foldiak and M. Young, *Sparse coding in the primate cortex*, The Handbook of Brain Theory and Neural Networks, MIT Press, Cambridge, MA (1995) p. 895–898.

[Hinton] G. E. Hinton, P. Dayan, B. J. Frey, and R. M. Neal, *The "wake-sleep" algorithm for unsupervised neural networks*, Science 268 (1995) p. 1158–1161.

[Horiuchi] T. K. Horiuchi, B. Bishofberger, and C. Koch, *An analog VLSI saccadic eye movement system*, Advances in Neural Information Processing Systems 6 (1994) p. 582–589.

[Jolliffe] I. T. Jolliffe, *Principal Component Analysis.* Springer-Verlag, New York, NY (1986).

[Konishi] M. Konishi, *Listening with two ears*, Scientific American, April (1993) p. 66–73.

[Landauer] T. K. Landauer and S. T. Dumais, *The latent semantic analysis theory of knowledge*, Psychological Review 104 (1997) p. 211–240.

[LeCun] Y. Le Cun, et al, *Backpropagation applied to handwritten zip code recognition*, Neural Computation 1 (1989) p. 541.

[Lee] D. D. Lee and H. S. Seung, *Unsupervised learning by convex and conic coding*, Advances in Neural Information Processing Systems 9 (1997) p. 515–521.

[Lucy] L. B. Lucy, *An iterative technique for the rectification of observed distributions*, Astronomical J. 74 (1974) 745.

[Nowlan] S. J. Nowlan, and J. C. Platt, *A convolutional neural network hand tracker*, Advances in Neural Information Processing Systems 7 (1995) p. 901–908.

[Olshausen] B. A. Olshausen and D. J. Field, *Emergence of simple-cell receptive field properties by learning a sparse code for natural images*, Nature 381 (1996) p. 607–609.

[Palmer] S. E. Palmer, *Hierarchical structure in perceptual representation*, Cognitive Psychology 9 (1977) p. 441–474.

[Petajan] E. Petajan and H. P. Graf, *Robust face feature analysis for automatic speechreading and character animation.* Proc. 2nd Int. Conf. Automatic Face and Gesture Recognition (1996) p. 357-362.

[Rabiner] L. R. Rabiner, *Fundamentals of speech recognition*, Prentice Hall, Englewood Cliffs, NJ (1993).

[Rao] R. P. N. Rao, G. J. Zelinsky, M. M. Hayhoe, and D. H. Ballard, *Modeling saccadic targeting in visual search*, Advances in Neural Information Processing Systems 8 (1996) p. 830–836.

[Richardson] W. H. Richardson, *Bayesian-based iterative method of image restoration*, J. Opt. Soc. Am. 62 (1972) 55–59.

[Rowley] H. A. Rowley, S. Baluja, and T. Kanade, *Human face detection in visual scenes*, Advances in Neural Information Processing Systems 8 (1996) 875–881.

[Salton] G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval* McGraw-Hill, New York, NY (1983).

[Saul] L. Saul, and F. Pereira, *Aggregate and mixed-order Markov models for statistical language processing*, In C. Cardie and R. Weischedel (eds), Proceedings of the Second Conference on Empirical Methods in Natural Language Processing (1997) p. 81–89.

[Sung] K. K. Sung and T. Poggio, *Example-based learning for view-based human face detection*, Proc. 23rd Image Understanding Workshop (1994) p. 843–850.

[Turk] M. Turk and A. Pentland, *Eigenfaces for recognition*, J. Cognitive Neuroscience 3 (1991) p. 71–86.

[Yang] J. Yang and A. Waibel, *A real-time face tracker*, Proc. 3rd IEEE Workshop on Application of Computer Vision (1996) p. 142–147.

[Yarbus] A. L. Yarbus, *Eye movements and vision* Plenum Press (1967).