**CIS6930/4930 Intro to Computational Neuroscience Fall 2008**
**Home Work Assignment 4**
**Steven Buss 9598-4342**

# 1 Asymptotic Equipartition

Consider the alphabet {A,B} with probabilities $p(A) = .27$ and $p(B) = .73$. The entropy of this distribution is $H = 0.841464$.

(1)
$$\begin{aligned} H &= -\Sigma p_i log_2(p_i) \\ &= -(.27 * log_2(.27) + .73 * log_2(.73)) \\ &= 0.841464 \end{aligned}$$

For a small $\epsilon = 0.001$, the number of sequences that have probabilities between $2^{-20*(H+\epsilon)}$ and $2^{-20*(H-\epsilon)}$ (when the sequences are generated i.i.d.) is, for sufficiently large n,:

(2)
$$\begin{aligned} (1-\epsilon)2^{n(H(X)-\epsilon)} \leq \; & |A_\epsilon^{(n)}| \leq 2^{n(H(X)+\epsilon)} \\ 114727 \leq \; & |A_\epsilon^{(n)}| \leq 118071 \end{aligned}$$

However, $n = 20$ is not sufficiently large, and it turns out that the size of the typical set with $\epsilon = 0.001$ is $0$. To expound, there are only 21[1] distinct probabilities for the $1048576$ sequences of length $20$ generated by the alphabet. This is because the probability of any given sequence, $S$, is $p(S) = p(A)^a * p(B)^b$ where a and b are the count of A and B in the sequence, respectively. Clearly, the order of appearance of A and B in the sequence does not matter, and $p(ABBA) = p(BBAA)$, thus yielding only $21$ different probabilities. Because the number of possible probabilities is so small, the weak law of large numbers does not come into effect and $\epsilon$ cannot be arbitrarily small. The smallest $\epsilon$ that yields a typical set with size $> 0$ is $\epsilon \approx .0287$.

Had $n = 20$ been sufficiently large, one would need $\lceil log_2(2^{n*H})\rceil$, or $\lceil n * H\rceil = 17$ bits to encode for any one of the sequences (assuming a simple index table for the sequences). By the law of large numbers, the sum of the probabilities for the sequences in the typical set is $P\{A_\epsilon^n\} = (1-\epsilon) = .999$. Assuming 20 bits to encode the remaining sequences, $\lceil 20 * (H+\epsilon)\rceil = 17$ bits on average to code for sequences of length $20$.

However, as stated before, because the typical set actually contains 0 values, 0 bits are needed to code for the typical set, and the sum of the probabilities of the sequences in the typical set is 0. This means that on average, 20 bits are necessary to encode the remaining sequences of length 20.

[1] There are 21 probabilities because, for example there are three probabilities for the sequences of length 2 = {AA,AB,BB}

# 2 Feed Forward Neural Network

The feed forward neural network did not produce results consistent with my expectations. The hidden nodes failed to properly learn the rules for a circle and were unable to correctly classify the circle:

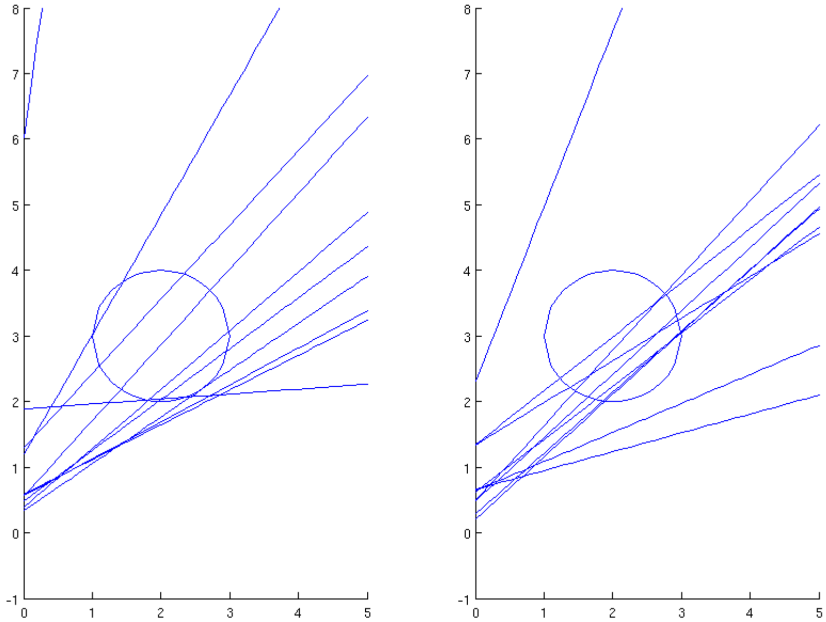
$$(x-2)^2 + (y-3)^2 < 1 \tag{3}$$

The network was trained on differing amounts of random inputs in the range $[0, 5]$, starting with 100 inputs and moving up to 10,000 in an attempt to get the network to learn the correct rules through sheer quantity of experience. Weights were both generated at random as well as hand picked for a variety of values, with no weights proving to be better than others. The hyperplanes of the hidden nodes learned with various numbers of training sets and initial weights are shown in Figures 1,2,3, and 4. Results of the network trying to classify a circle are shown in 5. Green pluses indicate the network has determined that coordinate is in the circle, and red dots indicates the network has determined that coordinate is outside the circle.

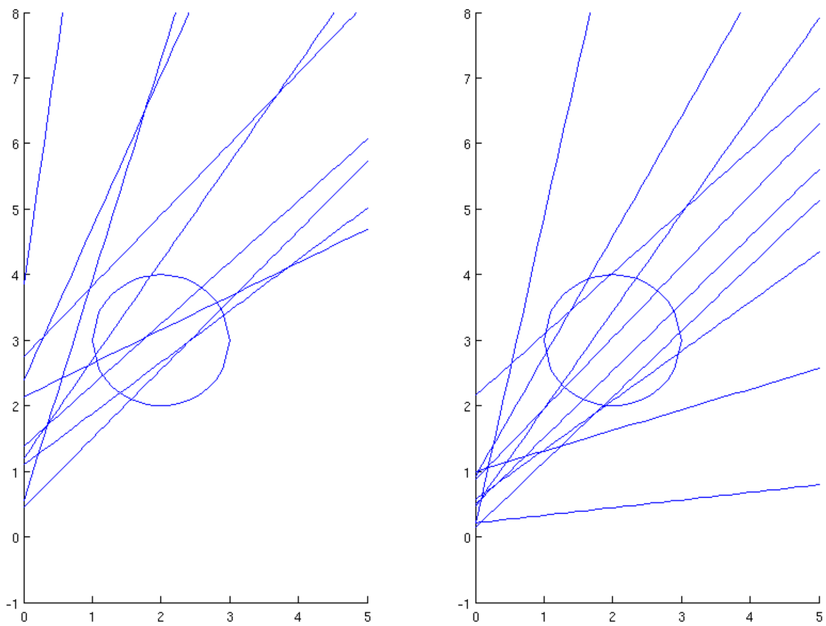
# 3 Hopfield Net

The Hopfield net also did not perform as expected. Instead of converging to one of the training images, the network often converged to a slight variation of one strong global minimum, perhaps local minima very near a global minimum. The set of training images and the output of the Hopfield net when the training images were fed back in is shown in Figure 8. The training images consisted of 1 bit dithered images of 13 faces, three abstract designs, one photograph of the moon's surface, one house, one bell peppper, and one gradient. Most of the inputs to the network after training converged on a vaguely face-like output image consisting of a thin band of hair around a featureless face, with a dark background to the left of the face and a light background to the right of the face. In some cases, the original image was mostly retrieved, as shown in Figure 6.

Repeated trials with different training images and different numbers of training images showed better performance when a smaller set of training images was used. When only the first five faces were trained against, the Hopfield net actually performed rather well. Results of introducing noise and removing data from input images, then feeding them into the Hopfield net are shown in Figure 7. It is interesting to note that all of these images converged to the correct training image after one iteration through all the pixels in the image. Introducing random data to the network also converged in one step to an image similar to the $3^{rd}$ training image.

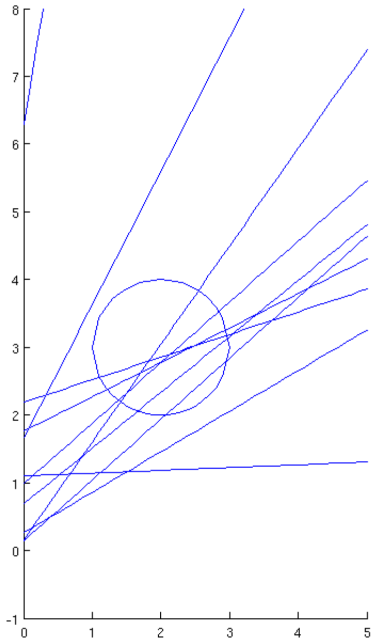



(a) 100 sets, random weights #1 (b) 100 sets, random weights #2

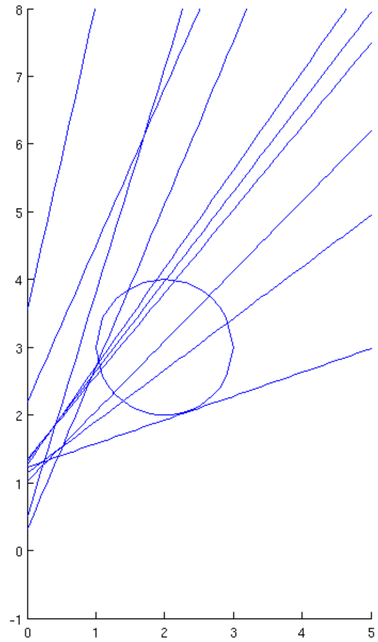



(c) 100 sets, weights=.2 #1 (d) 100 sets, weights=.2 #2

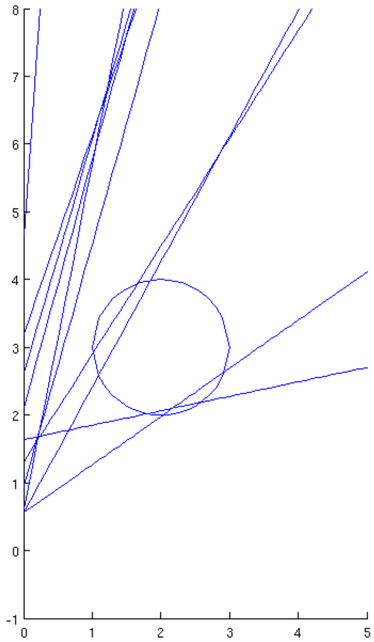
Figure 1: Neural Net, 100 training sets



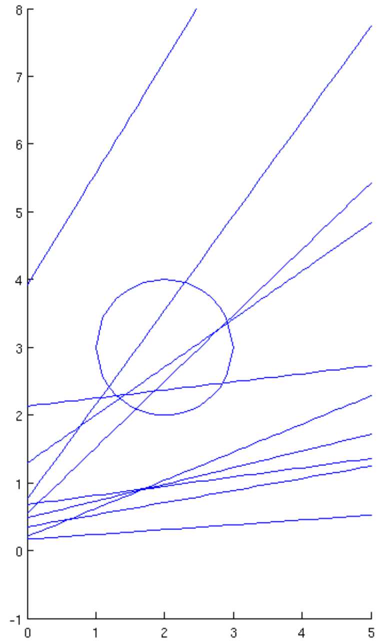
(a) 100 sets, weights=.5



(b) 100 sets, weights=.1

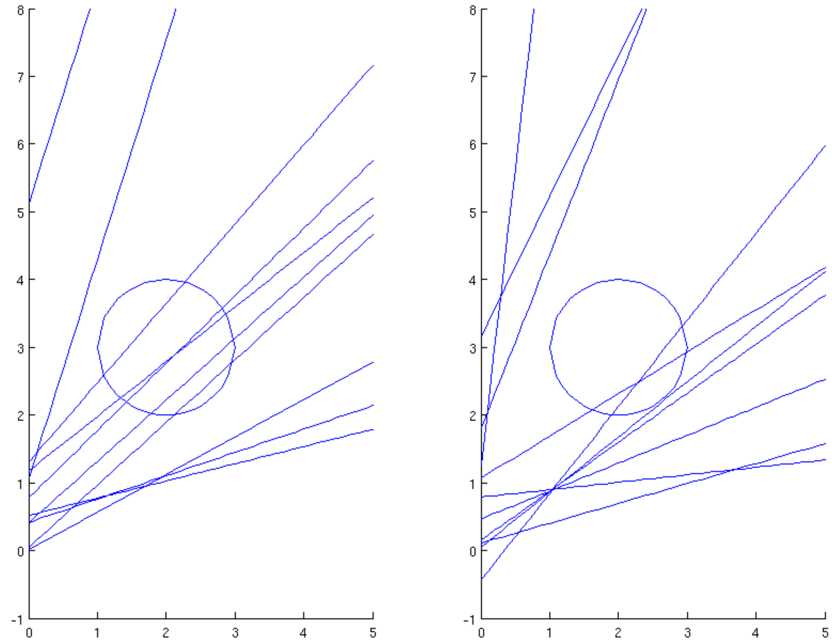


(c) 100 sets, weights=.1 to 1

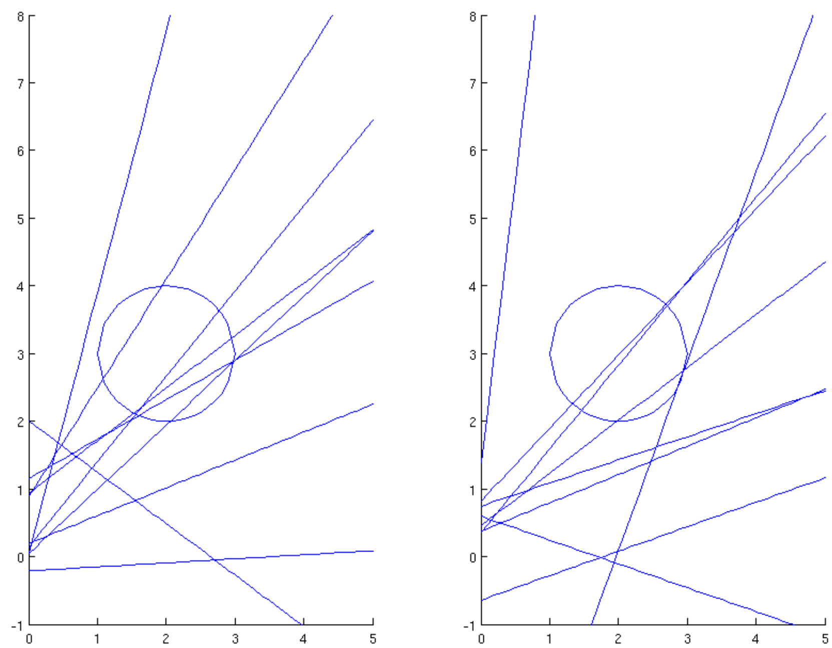


(d) 100 sets, weights=1 to .1

Figure 2: Neural Net, 100 training sets

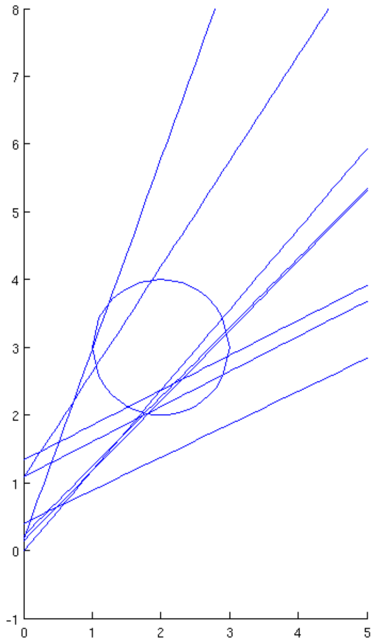



(a) 500 sets, random weights #1 (b) 500 sets, random weights #2

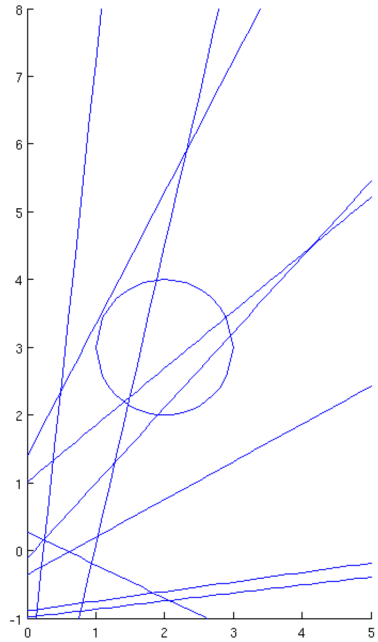



(c) 1000 sets, random weights (d) 2000 sets, random weights

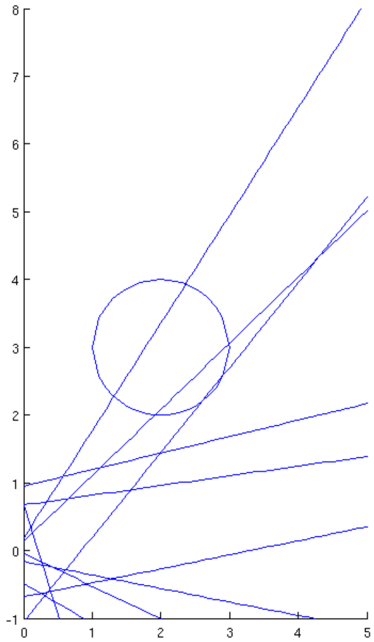
Figure 3: Neural Net, 500 to 2000 training sets with random weights



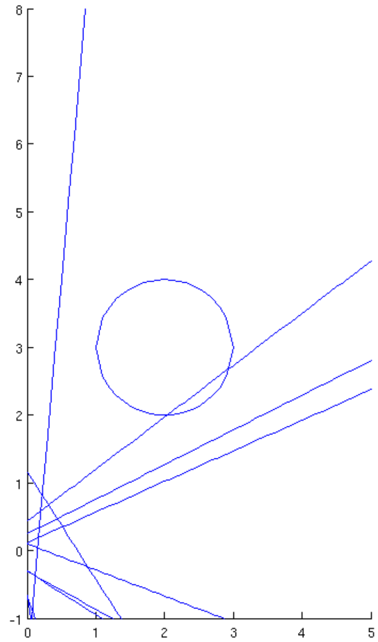

(a) 2000 sets, weights=.1




(b) 5000 sets, random weights

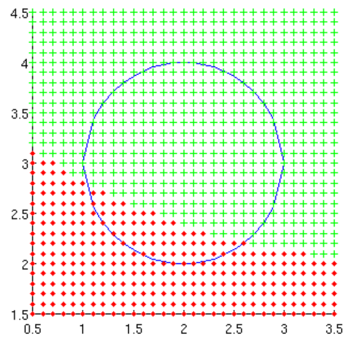



(c) 5000 sets, random weights

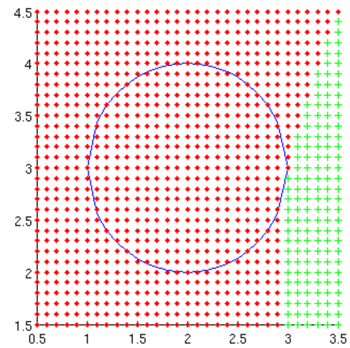



(d) 10000 sets, random weights

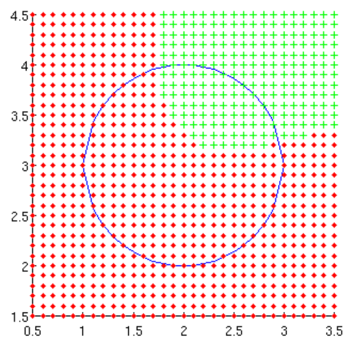
Figure 4: Neural Net, 2000 to 10000 training sets



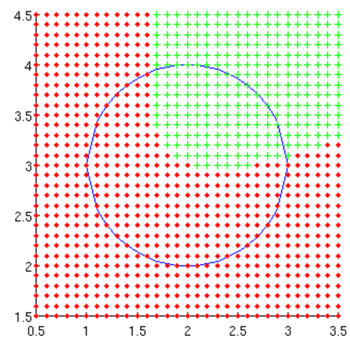

(a) 5000 training sets



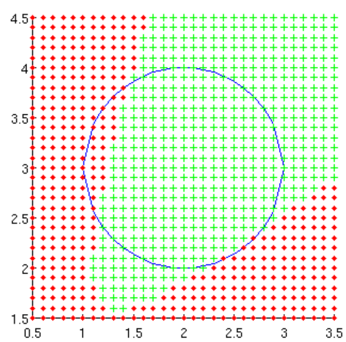

(b) 8000 training sets



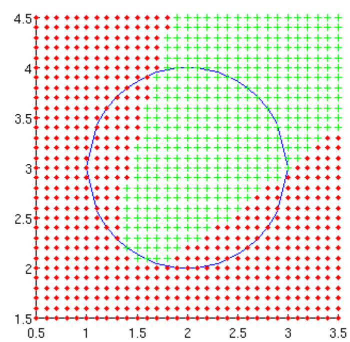

(c) 10000 training sets




(d) 10000 training sets



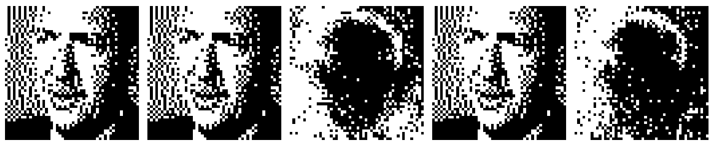

(e) 15000 training sets




(f) 20000 training sets

Figure 5: Classifying a circle

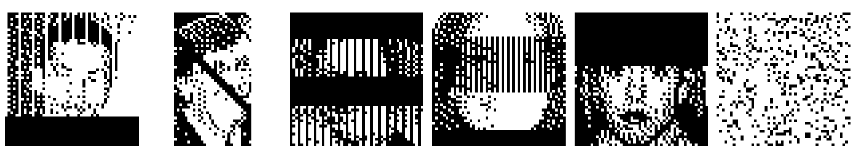
(a) input



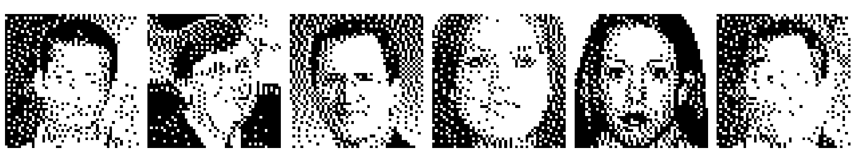
(b) output

Figure 6: Hopfield Net Correcting Errors

(a) training

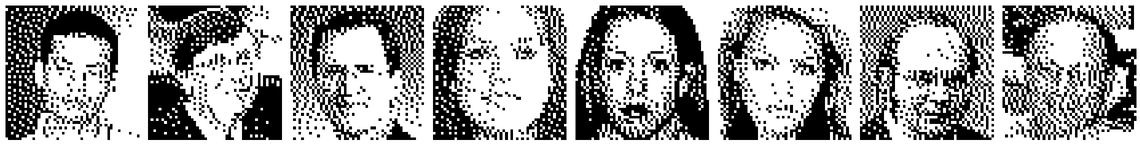


(b) input

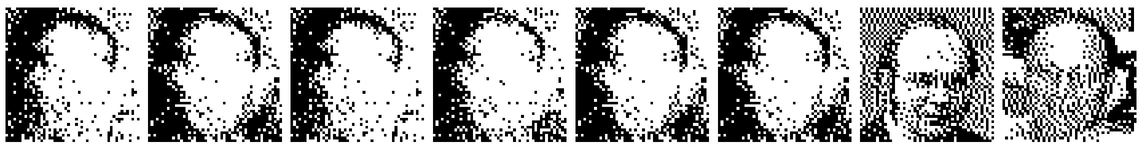


(c) output

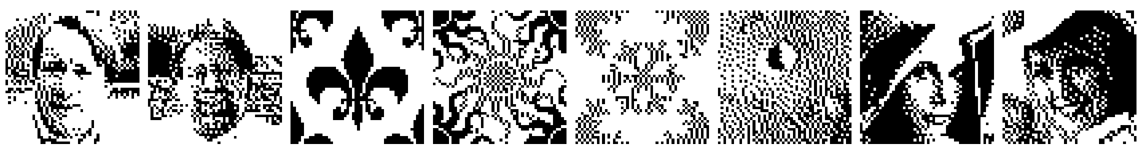
Figure 7: 5-Image Hopfield Net Correcting Errors



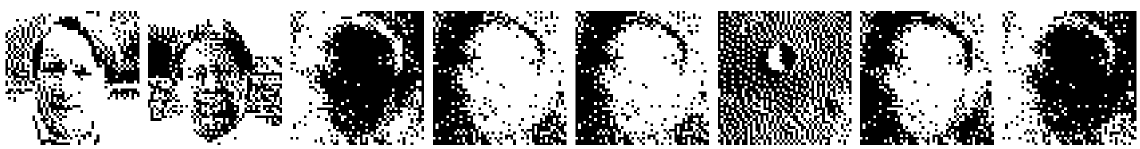
(a) input



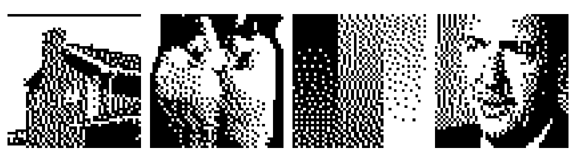
(b) output



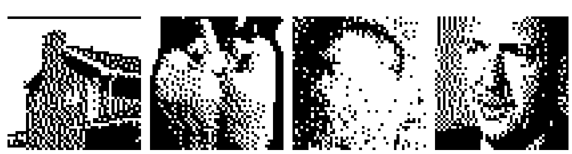
(c) input



(d) output



(e) input



(f) output

Figure 8: Hopfield Net