FIGURE CAPTIONS

FIGURE 1: a) A generative model for 4×4 images. The top level decides whether to use vertical or horizontal bars. The next level decides whether or not each possible bar of the chosen orientation should be present in the image. b) A sample of the images produced by the model in (a) with the ambiguous all-white images removed. A neural net with 16 input units, 8 units in the first hidden layer, and 1 hidden unit in the second hidden layer was trained on 2×10^6 random examples produced by the generative model. After training [15], the probability distribution produced in the sleep phase was almost exactly correct. c) The generative weights to and from the 8 units in the first hidden layer. Positive weights are white, negative weights are black, and the area is proportional to the magnitude. The largest weight shown is 14.1. The generative bias of the unit is shown on the top right of each block and its generative weight from the single unit in the layer above is shown on the top left. The rightmost block shows the generative biases of the input units. To encourage an easily interpretable solution, the generative weights to the input units were constrained to be positive. If they are allowed to go negative, the algorithm finds solutions that produce the correct distribution but in a much more complicated way, and it requires more units in the second hidden layer.

FIGURE2: Handwritten digits were normalized and quantized to produce 8×8 binary images. 24 examples of each digit are shown on the left. A separate network was trained on each digit class and 24 fantasies from each network are shown on the right. The variations within each digit class are modelled quite well. The error rate was 4.8% when new test images were classified by choosing the network that minimized the description length of the image. On the same data, nearest neighbor classification gave 6.7% errors and backpropagation training of a single supervised net with 10 output units and one hidden layer gave a minimum of 5.6% errors even when we used the test data to optimize the number of hidden units, the training time, and the amount of weight decay [16].