SGN-2556 Pattern Recognition

Computer exercise 3

Stochastic methods

11.04.2006

1 Boltzmann learning

The class of stochastic optimization problems can be viewed in terms of a network of nodes or units, each of which can be the $s_i = +1$ or $s_i = -1$ state. Every pair of nodes i and j is connected by the bidirectional weights w_{ij} ; if a weight between two nodes is zero, then no connection is drawn. The optimization problem is to find a configuration (i.e. assessment of all nodes) that minimizes the energy described by

$$E = -\frac{1}{2}\sum_{i,j=1}^{N} w_{ij}s_i s_j. \tag{1}$$

For pattern recognition the network structure presented in Fig.1 is used (Boltzmann network). It is important to distinguish between two types of visible units - d input units and c output units, which receive external binary feature and category information - as well as the remaining hidden units. During classification the input units are held fixed or clamped to the feature values of the input pattern; the remaining units are annealed to find the lowest-energy, most probable configuration. The category information is then read from the final values of the output units.

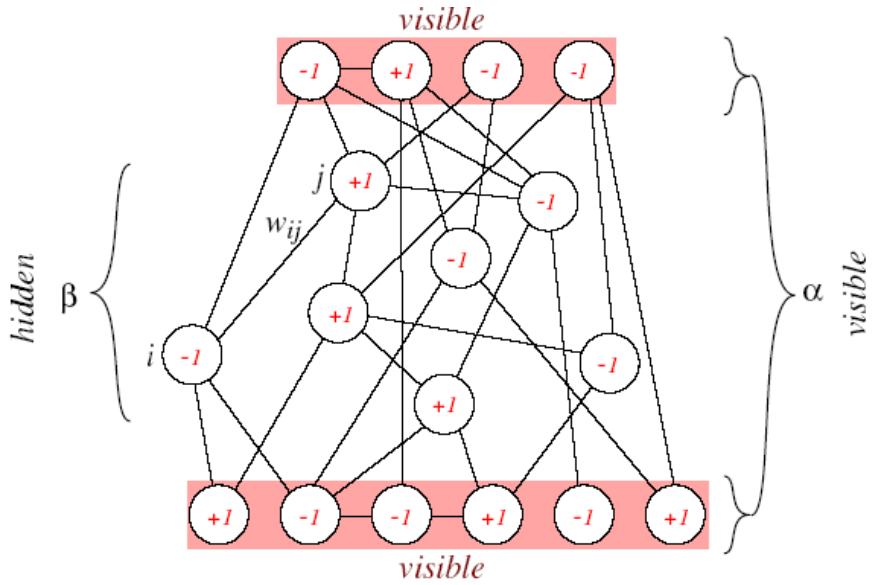


Figure 1: Boltzmann network

The weight update w_{ij} on each iteration is calculated as

$$\Delta w_{ij} = \frac{\eta}{T}\left[\underbrace{\epsilon_Q[s_i s_j]_{\alpha^i \alpha^0 clamped}}_{learning} - \underbrace{\epsilon[s_i s_j]_{\alpha^i clamped}}_{unlearning}\right]. \quad (2)$$

The first term is informally referred to as the learning component (as the visible units are held to values given by the teacher), and the second term is referred to as the unlearning component (where the output units are free to vary). If $w_{ij} = 0$, we have achieved the desired weights.

2 Exercise tasks

2.1 Boltzmann learning

1) Download the file data_boltzmann.mat that contains the test data consisting of three categories and ten patterns for each category as it is shown in the table below.

ω_1	ω_2	ω_3
−−+−+−++	−++−+++−	+++++−++
−−++−+++	−++−+−−−	+−+−++−+
+++++−−+	−+++−++−	+−++−+++
−+−+−++−	−++−−+−+	+−++−+−−
−−−−+++−	−+−−−−++	+−++−++−
−−−++++−	−+−−++−+	+−−+−++−
−+−+++−−	−−+−++−−	+++−+−−+
−+−++−−+	−+−+−+++	+−++−+−+
+−+−+−++	−−−++−++	++−−+−++
+−+−−+++	−++−++−+	+−−+−++−

2) Using the file train_boltzmann.m train a Boltzmann network to create a classifier for distinguishing between the three categories. Choose the number of hidden units to be at least ceil($\log_2$n), which is the lower bound for the number of hidden units. n is the number of distinct patterns in the training dataset. Set the cooling rate to 0.99.

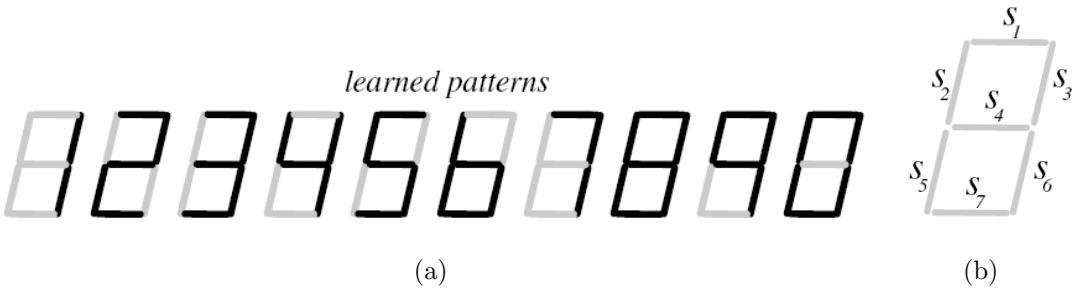
3) Use the trained network to classify each pattern from ω_1 and, thus, verify that most of the patterns have been learned. Use the function test_boltzmann.m for it. Run the classification several times and evaluate the average percent of incorrect classifications for this class? Repeat classification, but for distinguishing each pattern from $\omega_2; \omega_3$.

4) Use the trained network to classify the patterns: −−−−+−++, -++-++++, ++++++++.

5) Increase the number of hidden units (but to be not greater than n, which is the upper bound on the minimum number of hidden units) and repeat steps (b)-(c). How has the classification error changed?

2.2 Boltzmann learning: pattern recognition with missing features.

1) Train a Boltzmann network consisting of seven input units and ten category units with the characters of a seven-segment display shown in Fig. 2. You have to define the matrix of input patterns and matrix of targets using binary values $\{-1, 1\}$ according to the figure below.



(a) (b)

Figure 2: Ten numeral patterns for seven-segment display.

2) Use the network to classify each of the ten patterns and, thus, verify that most of them have been learned.

3) Explore the pattern completion in your network in the following way: devise $3-4$ deficient patterns with 1 or 2 missing features, which can be unambiguously completed. Use your trained network to learn (complete) these patterns. Try another $3-4$ patterns with missing features, but for characters, which can be ambiguously completed.

4) Add more hidden units and compare the performance of the network in completing the ambiguous/unambiguous characters with different number of hidden units.