Competitive Learning in Decision Trees

Dominique Martinez
Laboratoire d'Analyse et d'Architecture des Systèmes (LAAS-CNRS)
7, Av. Col. Roche
31077 Toulouse - France

Abstract

In this paper, a competitive learning rule is introduced in decision trees as a computationally attractive scheme for adaptive density estimation or lossy compression. It is shown by simulation that the adaptive decision tree performs at least as well as other competitive learning algorithms while being much faster.

keywords: competitive learning, decision tree, density estimation, vector quantization, neural networks.

1 Introduction

The use of decision trees for classification and regression is relatively familiar to statisticians (Breiman, Friedman, Olshen & Stone, 1984). Decision trees are built in a supervised fashion to perform hierarchical partitioning over the input space by means of axis-parallel hyperplanes. Orthogonal hyperplanes offer considerable benefits in terms of computational complexity because encoding requires no multiplications. Decision trees might be helpful for density estimation as well as lossy compression using vector quantization in which conventional design algorithms hit the complexity "barrier" when the input dimensionality is too high. Unfortunately, very little work has been done on combining decision trees with unsupervised learning.

In this paper, competitive learning is introduced in decision trees as a computationally attractive scheme for density estimation or lossy compression. In Section 2, the high resolution theory is used to analyze the performance of decision trees. High resolution theory assumes a large number of partition cells in order to consider the density approximately constant over each cell. This, in turn, allows us to derive a necessary condition for optimality in the case of density estimation or lossy compression.

If density estimation is the objective sought, the proper way of achieving quantization is through maximized quantizer entropy. Thus, the N partition cells possess an equal probability $\frac{1}{N}$ of winning the competition and the density is estimated as $\frac{1}{N\,V(S_i)}$ in each partition cell S_i of volume $V(S_i)$. If the goal is lossy compression, correct quantization will require minimizing the average distortion because entropy maximization obviates any possibility of further compression by lossless coding. Section 3 describes a competitive learning rule which minimizes the r-th power law distortion in the high resolution case. When $r = 2$, the learning rule minimizes the classical mean squared error distortion, aiming at lossy compression. When $r = 0$, the learning rule yields equiprobable quantization for density estimation purposes.

2 High-Resolution Theory in Decision Trees

Let $\mathbf{x} = (x_1, x_2, \cdots x_k)$ be a k-dimensional input vector drawn from a probability density function $p_X(\mathbf{x})$. Assume also that $p_X(\mathbf{x})$ vanishes outside a finite region S so that $\mathbf{x}$ takes on values in S with probability one.

Decision trees perform a hierarchical partitioning of the k-dimensional feature space by means of hyperplanes perpendicular to coordinates axes. An example of hierarchical partitioning is shown in Fig. 1. This results in a binary tree structure in which each internal node t stores two scalar quantities : an index l representing the dimension orthogonal to the hyperplane and a weight w_{tl} representing the location of the hyperplane on this axis. Any input vector $\mathbf{x}$ can be located with respect to this hyperplane through use of a single scalar comparison. If $x_l < w_{tl}$, the feature vector $\mathbf{x}$ is sent to the left child of node i. Similarly, the feature vector is sent to the right child if $x_l \geq w_{tl}$. The leaf nodes are associated with N disjoint partition cells S_i, $1 \leq i \leq N$, whose boundaries are determined by the weights stored above. The geometric boundaries of the cells S_i are defined by the lower and upper boundary point vectors $\mathbf{x}_i^L = (x_{i1}^L \cdots x_{ik}^L)$ and $\mathbf{x}_i^U = (x_{i1}^U \cdots x_{ik}^U)$. The cut value w_{tl} stored at node t may be used for dividing more than two quantization cells. Let $LB(t)$ be the number of cells S_i for which w_{tl} is a lower boundary point, *i.e.* $w_{tl} \equiv x_{il}^L$. Respectively, $UB(t)$ is the number of cells S_i for which w_{tl} is an upper boundary point, *i.e.* $w_{tl} \equiv x_{il}^U$. For the root node of the tree in Fig. 1, w_{11} is an upper boundary point for the cells S_3, S_4 and S_6 and it is a lower boundary point for the cell S_7. Therefore, we have $UB(1) = 3$ and $LB(1) = 1$.

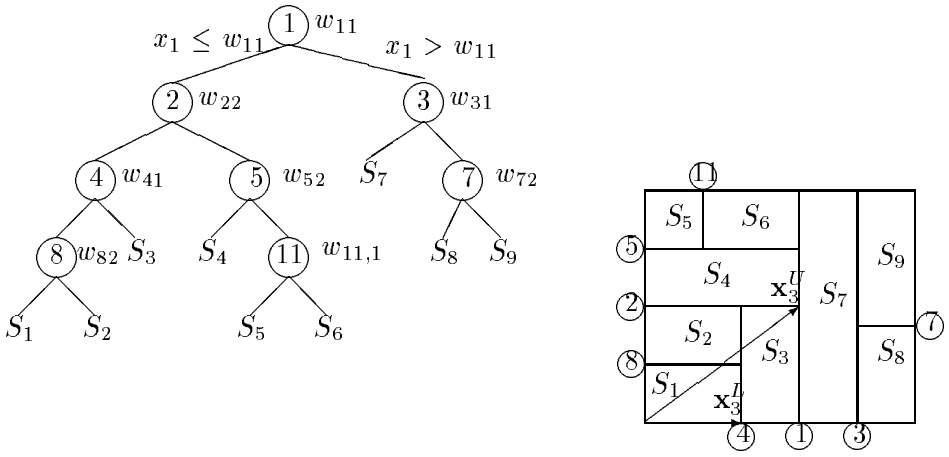


Figure 1: Example of hierarchical partitioning over the input space which could be induced by the binary tree. The lower and upper boundary point vectors of the cell S_3 are given by $\mathbf{x}_3^L = (w_{41}, 0)$ and $\mathbf{x}_3^U = (w_{11}, w_{22})$.

When $\mathbf{x} \in S_i$, it is quantized as $\mathbf{y}_i$ and it results in a quantization error. A frequent goal of competitive learning is the minimization of the average distortion

$$\begin{aligned} D_r(k) &= \frac{1}{k} \int_S ||\mathbf{x} - \mathbf{y}||_\nu^r p_X(\mathbf{x}) d\mathbf{x} \\ &= \frac{1}{k} \sum_{i=1}^{N} \int_{S_i} ||\mathbf{x} - \mathbf{y}_i||_\nu^r p_X(\mathbf{x}) d\mathbf{x} \end{aligned} \tag{1}$$

where the distortion measure $||.||_\nu^r$ is the rth power of the l_ν norm

$$||\mathbf{x}-\mathbf{y}_i||_\nu^r = (\sum_{l=1}^{k} |x_l - y_{il}|^\nu)^{\frac{r}{\nu}} \tag{2}$$

The high resolution theory assumes N large and $p_X(\mathbf{x})$ smooth so that

$$p_X(\mathbf{x}) \approx p_X(\mathbf{y}_i) = \frac{P(S_i)}{V(S_i)} \text{ for } \mathbf{x} \in S_i$$

where $P(S_i) = \int_{S_i} p_X(\mathbf{x})d\mathbf{x}$ and $V(S_i) = \int_{S_i} d\mathbf{x}$ denote the probability and volume of S_i, respectively. Then, Eq. (1) can be approximated as

$$\begin{aligned} D_r(k) &\approx \frac{1}{k}\sum_{i=1}^{N} p_X(\mathbf{y}_i) \int_{S_i} ||\mathbf{x}-\mathbf{y}_i||_\nu^r d\mathbf{x} \\ &= \sum_{i=1}^{N} p_X(\mathbf{y}_i) I(S_i)[V(S_i)]^{1+\frac{r}{k}} \end{aligned} \tag{3}$$

where $I(S_i)$ denotes the normalized moment of inertia of S_i about the point $\mathbf{y}_i$

$$I(S_i) = \frac{\int_{S_i} ||\mathbf{x}-\mathbf{y}_i||_\nu^r d\mathbf{x}}{k[V(S_i)]^{1+\frac{r}{k}}} \tag{4}$$

If the distortion is a rth power of the l_2 norm, Zador (1966, 1982) has shown that

$$D_r(k) = A(k,r)N^{-r/k}||p_X(\mathbf{x})||_{k/(k+r)}$$

with the term $A(k,r)$ being independent of $p_X(\mathbf{x})$. Gersho (1979) conjectured that, for large N, most cells are congruent to the polytope with minimum normalized moment of inertia and, thus, considered $A(k,r)$ as the normalized moment of inertia of the optimum polytope. However, the difficulty in utilizing this expression is that $A(k,r)$ is only known with certainty in a few cases.

In the following, we limit our study to single-letter distortion measures for which $\nu = r$ in Eq. (2). Therefore, distortion is the rth power of the l_r norm. This distortion measure is of interest in waveform coding because it allows to place a rth power law on every individual sample. The most commonly used powers are $r = 1$ (mean absolute error) (Kassam, 1978), $r = 2$ (mean squared error) (Gersho & Gray, 1992) and $r = \infty$ (mean maximum absolute error).

Each partition cell S_i of the decision tree is a hyperbox aligned with the coordinate axes and defined by the lower and upper boundary point vectors $\mathbf{x}_i^L = (x_{i1}^L \cdots x_{ik}^L)$ and $\mathbf{x}_i^U = (x_{i1}^U \cdots x_{ik}^U)$. Thus, the normalized moment of inertia of S_i can be expressed as

$$I(S_i) = \frac{1}{k[V(S_i)]^{1+\frac{r}{k}}} \int_{x_{ik}^L}^{x_{ik}^U} \cdots \int_{x_{i1}^L}^{x_{i1}^U} \sum_{l=1}^{k} |x_l - y_{il}|^r dx_1 \cdots dx_k \tag{5}$$

The high-resolution assumption enables us to take $\mathbf{y}_i$ at the center of its partition cell. Moreover, it is useful to rewrite the absolute value as

$$\begin{aligned}\int_{x_{il}^L}^{x_{il}^U} |x_l - y_{il}|^r dx_l &= \int_{x_{il}^L}^{y_{il}} (y_{il} - x_l)^r dx_l + \int_{y_{il}}^{x_{il}^U} (x_l - y_{il})^r dx_l \\ &= \frac{1}{2^r(r+1)}[\delta_{il}]^{r+1} \text{ , with } \delta_{il} = x_{il}^U - x_{il}^L \text{ .}\end{aligned}$$

Then, integrating (5) yields

$$I(S_i) = \frac{1}{k2^r(r+1)[V(S_i)]^{r/k}} \sum_{l=1}^{k} [\delta_{il}]^r \tag{6}$$

The inequality property between arithmetic and geometric means gives the following lower bound

$$I(S_i) \geq \frac{1}{2^r(r+1)} \tag{7}$$

with equality in case of hypercubical cells with $\delta_{il} =$ constant for all i. With the Gersho's conjecture, the minimum normalized moment of inertia given by Eq. (7) applies to all cells and, using Eq. (3), the distortion can be approximated as

$$D_r(k) \approx \frac{1}{2^r(r+1)} \sum_{i=1}^{N} p_X(\mathbf{y}_i)[V(S_i)]^{1+\frac{r}{k}} \tag{8}$$

A necessary condition for minimizing $D_r(k)$ is obtained by using Lagrange's multiplier method, as shown in (Panter & Dite, 1951) for scalar quantization and mean squared error criterion ($r = 2$ and $k = 1$)

$$P(S_i)[V(S_i)]^{\frac{r}{k}} = P(S_j)[V(S_j)]^{\frac{r}{k}} \text{ for all } i, j \tag{9}$$

From Eqs. (12), (8) and (9), it follows that every quantization cell S_i produces an identical distortion contribution to the total distortion. This *equidistortion principle* was first reported by Panter and Dite (1951) for scalar quantizers and then extended by Gersho (1979) to vector quantizers with a Voronoi or Dirichlet partition. Eq. (9) shows that the equidistortion principle also applies to decision trees. Also note that $r = 0$ in (9) leads to an equiprobable quantization, $P(S_i) = \frac{1}{N}$ for all i.

3 Unsupervised Learning in Decision Trees

To derive a suitable competitive learning rule, $\mathbb{1}_{S_i}(\mathbf{x})$ is first defined as the code membership function of the partition cell S_i, *i.e.*

$$\mathbb{1}_{S_i}(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{x} \in S_i \\ 0 & \text{if } \mathbf{x} \notin S_i. \end{cases}$$

The unsupervised learning rule, called $BAR_r(k)$, is given by

$$\begin{aligned}\Delta w_{tl} &= w_{tl}(n) - w_{tl}(n-1) \\ &= \eta \left(\sum_{i=1|w_{tl} \equiv x_{il}^L}^{N} [V(S_i)]^{\frac{r}{k}} \frac{\mathbb{1}_{S_i}}{LB(t)} - \sum_{i=1|w_{tl} \equiv x_{il}^U}^{N} [V(S_i)]^{\frac{r}{k}} \frac{\mathbb{1}_{S_i}}{UB(t)} \right) \end{aligned} \tag{10}$$

with η the learning rate. The $V(S_i)$s and $\mathbb{1}_{S_i}$s are defined at the previous time step. Assume that for input $\mathbf{x}$, $\mathbb{1}_{S_j} = 1$. The learning rule (10) then modifies S_j by increasing the cut values corresponding to the lower boundary point vector $\mathbf{x}_j^L$ and decreasing the cut values corresponding to the upper boundary point vector $\mathbf{x}_j^U$ in proportion to the $\frac{r}{k}$th power of the volume $V(S_j)$. Because this learning rule explicitly modifies the boundaries of the partition cells in a k-dimensional space, it is called a Boundary Adaptation Rule $BAR_r(k)$. Fig. 2 shows an example of adaptation in which $\mathbb{1}_{S_3} = 1$.

Note that the internal nodes involved in dividing many quantization cells could be more frequently updated than other nodes. Therefore, denominator terms $LB(t)$ and $UB(t)$ in (10) are needed in order to not favor the updates of particular nodes in the tree. It can be shown that $BAR_r(k)$ satisfies the necessary condition Eq. (9) at convergence.

With respect to the scalar case, one gets $k = 1$, $LB(t) = UB(t) = 1$ for all t and the partition cells become intervals : $S_i \equiv [w_i, w_{i-1})$, given an ordered set of boundary points $w_i > w_{i-1}$ for $i = 1 \cdots N$. Eq. (10) reduces to

$$\Delta w_i = \eta(\delta_{i+1}^r \mathbb{1}_{S_{i+1}} - \delta_i^r \mathbb{1}_{S_i}) \tag{11}$$

which has been previously introduced for adaptive scalar quantization in (Martinez & VanHulle, 1995). In addition, it is proven that the system of ordinary differential equations associated with Eq. (11) converges to a unique boundary point solution.

4 Experimental Results

$BAR_r(k)$, Eq. (10) with $r \neq 0$, attempts to minimize the r-th power law distortion aiming at lossy compression. If $r = 0$, the learning rule $BAR_0(k)$ attempts to maximize Shannon's entropy aiming at density estimation. At convergence, the partition cells win the competition with an equal probability $\frac{1}{N}$ (*equiprobable quantization*) and the probability density function can be estimated as

$$p_X(\mathbf{x}) \approx p_X(\mathbf{y}_i) = \frac{1}{N \ V(S_i)} \text{ for } \mathbf{x} \in S_i$$

4.1 Density estimation (r=0)

4.1.1 Stationary distributions

In order to demonstrate the performance of decision trees trained with $BAR_0(k)$ in terms of entropy maximization, we will consider both symmetrical and asymmetrical

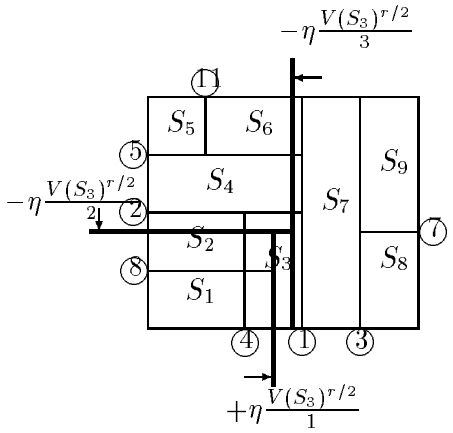


Figure 2: Example of adaptation of the decision tree shown in Fig. 1. It is assumed that for input $\mathbf{x}$, $\mathbb{1}_{S_3} = 1$. Only nodes 1,2 and 4 defining the partition cell S_3 are updated. For lossy compression minimizing the r-th power law distortion ($r \neq 0$), Eq. (10) gives the following modifications: $\Delta w_{11} = -\eta\frac{V(S_3)^{r/2}}{3}$, $\Delta w_{41} = +\eta\frac{V(S_3)^{r/2}}{1}$ and $\Delta w_{22} = -\eta\frac{V(S_3)^{r/2}}{2}$. For density estimation ($r = 0$), the modifications simplify to $\Delta w_{11} = -\frac{\eta}{3}$, $\Delta w_{41} = +\eta$ and $\Delta w_{22} = -\frac{\eta}{2}$.

k-dimensional input probability density functions, for k up to 3. Entropy performance results are listed in Table 1 for Uniform, Gaussian and Exponential densities of unit standard deviation. The values given are averages $\pm$ standard deviations over 10 runs. For each run, the decision tree was first built up in a greedy fashion by considering the splitting of node and axis contributing most to the decrease in average distortion. Then, it was trained on an i.i.d sequence of 20,000 samples and the entropy value was estimated using 40,000 samples. We observe that the relative difference between the entropy value obtained at convergence and the maximum entropy value was observed to be less than $2\%_0$ in every case. In addition, there is no significant difference in entropy performance for symmetrical or asymmetrical densities. These results clearly show the efficiency with which $BAR_0(k)$ converges toward equiprobable quantization.

$\mathrm{BAR}_0(k) \rightarrow$ pdf $\downarrow$	k=1, N=8	k=2, N=64	k=3, N=128
Uniform	2.999 $\pm$ 2.3E-4	5.997 $\pm$ 5.7E-4	6.993 $\pm$ 9.1E-4
Gaussian	2.999 $\pm$ 3.0E-4	5.997 $\pm$ 5.9E-4	6.993 $\pm$ 7.6E-4
Exponential	2.999 $\pm$ 2.0E-4	5.989 $\pm$ 1.0E-2	6.993 $\pm$ 8.6E-4

Table 1: The entries listed in the table are entropy values for an input dimension k up to 3. Performance is given for Uniform, Gaussian and Exponential probability density functions (pdf) of unit variance and zero mean. The values are averages over 10 runs with standard deviation. They are estimated by using 40,000 samples drawn from the respective densities. The training set consists of 20,000 samples. The learning rate is linearly decreasing from 0.01 to zero for $BAR_0(1)$ and $BAR_0(2)$ and from 0.03 to zero for $BAR_0(3)$. The number of partition cells is $N = 8$, 64 and 128 for $k = 1$,2 and 3, respectively

Figures 3(A1)-(A3) show the true densities (Uniform, Gaussian and Exponential) and the partitions given by the decision tree ($N = 64$ and $k = 2$) at convergence. The partitions reveal smaller cells in high-density regions and larger cells in low-density regions relative to the current underlying input distribution. Thus, the learning rule provides an adaptive focusing mechanism. Figures 3(B1)-(B3) and 3(C1)-(C3) show the density estimated as a multivariate histogram, constructed with $\hat{p}(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathbb{1}_{S_i}}{V(S_i)}$, and as a frequency polygon, constructed by linear interpolation, respectively.

4.1.2 Piecewise stationary distributions

Consider the sequence $\{\mathbf{x}(t)\}, t = 1 \cdots 20000$, of independently distributed random variables. The sequence is generated from a two-dimensional (2D) parity problem in which a change occurs when $t = 10000$. Before this change, samples are drawn from unit variance, Gaussian processes with mean vectors $(0,0)$, $(1,1)$. When $t = 10000$, the mean vectors suddenly change to $(0,1)$ and $(1,0)$. We trained the decision tree with $BAR_0(2)$ and a fixed learning rate $\eta = 0.01$ in order to adaptively partition the input space into $N = 64$ equiprobable cells. The starting weight configuration was obtained by growing a decision tree over the entire training sequence, as previously.

The performance of the adaptive neural tree is envisaged in terms of entropy maximization and compared with a number of unsupervised competitive learning algorithms: standard competitive learning (standard CL), the original conscience learning (Conscience 1) (DeSieno, 1988) and its slightly modified version (Conscience 2) (VanDenBout, 1989), neural gas learning (Martinetz, Berkovich & Shulten, 1993), the original Frequency-Sensitive Competitive Learning (FSCL 1) (Ahalt, Krishnamurthy, Chen & Melton, 1990) and its modified version (FSCL 2) (Galanopoulos & Ahalt, 1996). Since FSCL 2 was known to approximate the density as $p(\mathbf{x})^{(3\beta+1)/(3\beta+3)}$ (Galanopoulos & Ahalt, 1996), a large β value ($\beta = 10$) was chosen for the simulations. Note that FSCL 1 is similar to FSCL 2 when β is set equal to one. For Conscience learning 1 and 2, the conscience factor was 10 and 2, as suggested in (DeSieno, 1988) and (VanDenBout, 1989), respectively. After extensive simulations, a fixed learning rate value $\eta = 0.04$ and $\eta = 0.01$ was chosen for the unsupervised competitive learning algorithms and the decision tree, respectively. At each time step, entropy performance was estimated for the different algorithms with respect to the current input distribution and plotted as a time evolution shown in figure 4(A). Results are averages over 20 runs. We observe that standard CL, neural gas, Conscience 1 and FSCL 1 lead to very poor entropy performance. This is not surprising since standard CL and neural gas, which are stochastic gradient descent algorithms for minimizing the mean squared error distortion, are known to approximate the density as $p(\mathbf{x})^{1/3}$. Thus, they tend to over-estimate the low density regions and under-estimate the high density regions. On the other hand, the decision tree matches the input density more closely and performs at least as well as the best competitive learning algorithms such as Conscience 2 and FSCL 2. Thus, the learning rule provides an adaptive focusing mechanism capable of tracking time-varying distributions. Furthermore, it was found to be at least 20 times faster than any of the other rules. A highly time-consuming part of competitive learning algorithms is due to encoding. Figure 4(B) compares the encoding speed for a set of 20000 k-dimensional input vectors in the different algorithms. The number of partition cells scales as $N = 32\ k$, for k up to 10. The encoding complexity of the competitive

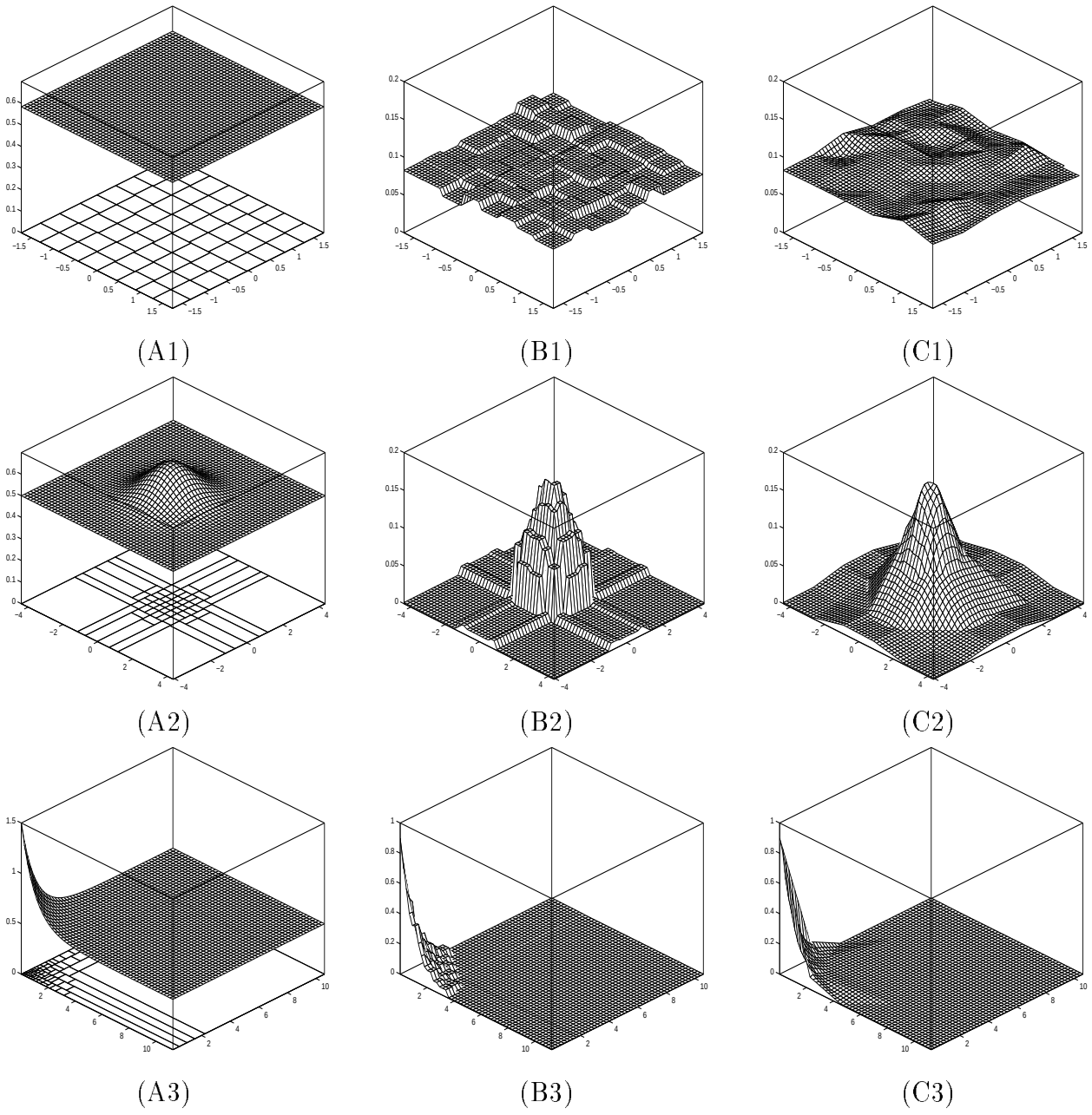
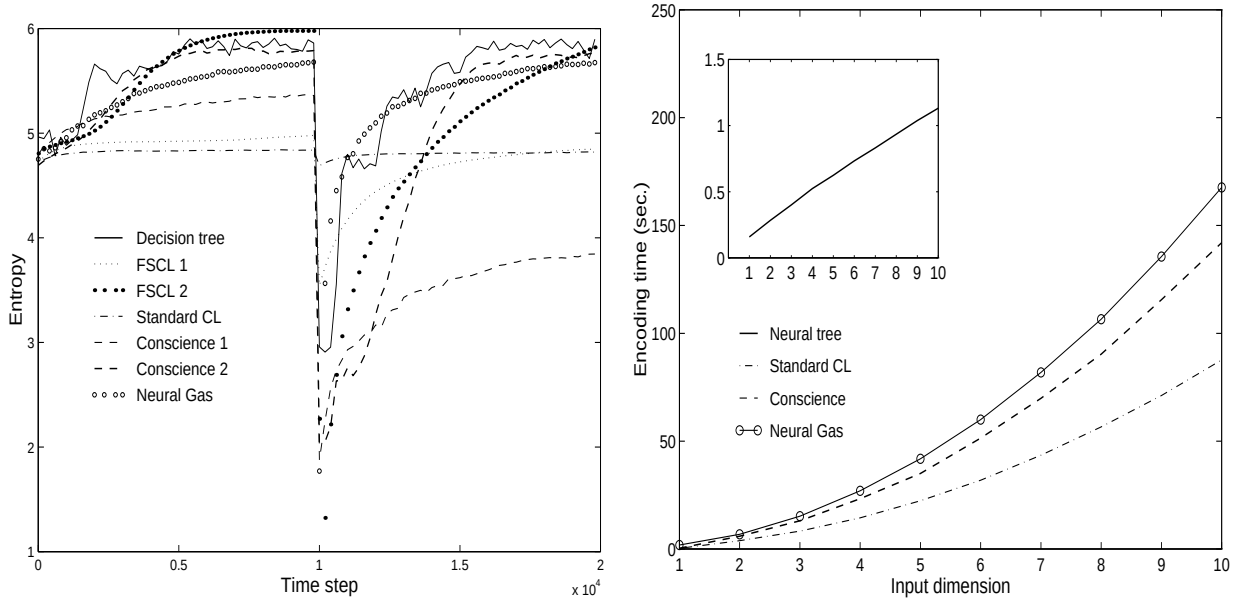


Figure 3: Figures 3(A1)-(A3) show the true densities (Uniform, Gaussian and Exponential) and the partitions obtained from the decision tree with $N = 64$ and $k = 2$ at convergence. Figures 3(B1)-(B3) and 3(C1)-(C3) show the density estimated as a multivariate histogram, constructed with $\hat{p}(\mathbf{x}) = \frac{1}{N} \sum_{i=1}^{N} \frac{\mathbb{1}_{S_i}}{V(S_i)}$, and as a frequency polygon, constructed by linear interpolation, respectively.

neural networks increases exponentially with k since the nearest-neighbor search requires $k\ N$ multiply and accumulate operations. As expected, the slowest algorithm was the neural-gas which requires ranking all the neurons and not only determining the winner. On the other hand, the decision tree is many orders of magnitude faster than the other algorithms because its encoding operation only requires comparisons, whose complexity may be assumed to be negligible.



(A) (B)

Figure 4: (A) Entropy performance over time for $k = 2$ and $N = 64$. Results are averages over 20 runs. (B) Time in sec. taken to encode a set of 20 000 k-dimensional samples generated from a uniform distribution. The number of partition cells is $N = 32\ k$, for k up to 10.

4.2 Vector quantization (r=2)

Vector quantization design using the Generalized Lloyd Algorithm (GLA) involves iterative applications of the nearest neighbor partition and centroid estimation on the training data set so as to monotonically decrease the distortion function towards a local minimum (Linde, Buzo & Gray, 1980). As a result, the quantizer is designed for a given source based on its long term statistical behavior. However, there are many situations in which communication systems may have to carry signals of changing statistics, e.g. speech or images, and the operational data may therefore be quite different from that of the training data. In order to minimize the quantizer-source mismatch due to nonstationnary inputs, the decision tree is continuously adapted with a fixed learning rate value so as to match the observed local statistics of the input sequence.

Table 2 shows the signal-to-noise ratio (SNR) in dB estimated for Uniform, Gaussian and Exponential densities of unit standard deviation. The vector quantizer and the decision tree were optimized by using the GLA and $BAR_2(k)$ for each of these densities, respectively. Because the decision tree is continuously adapted by using $BAR_2(k)$ with a fixed learning rate, the boundaries of the partition cells are modified at each time step.

Therefore, the output $\mathbf{y} = \mathbf{y}_i$ is reconstructed as the midpoint of the active partition cell S_i when $\mathbf{x} \in S_i$.

In the event of no quantizer-source mismatch, the SNR obtained from the GLA is, at the most, 1 dB higher than the SNR obtained from $BAR_2(k)$. However, performance degrades significantly for the GLA when the quantizer is used with sources other than those for which it was optimized. If the training set is drawn from an Exponential density while the test set is generated either from a Uniform or a Gaussian density, the decision tree trained with $BAR_2(k)$ outperforms the GLA by at least 10 dB.

	k=1			**k=2**			**k=3**		
Test → Train ↓	**U**	**G**	**E**	**U**	**G**	**E**	**U**	**G**	**E**
U									
GLA	18.01	12.35	3.27	17.92	12.37	3.56	17.87	12.67	3.66
$BAR_2(k)$	18.03	14.00	13.97	17.79	13.89	12.55	17.84	13.89	8.67
G									
GLA	15.00	14.58	6.05	15.76	15.09	6.79	15.90	15.31	7.31
$BAR_2(k)$	18.03	14.00	13.98	16.73	14.05	12.51	16.20	14.40	10.47
E									
GLA	1.75	1.83	15.12	2.19	2.20	16.57	2.40	2.42	16.43
$BAR_2(k)$	15.90	13.51	14.46	13.16	12.60	15.28	11.35	10.40	15.83

Table 2: The entries listed in the table are SNR values in dB for a transmission rate of 3 bits/sample and an input dimension k up to 3. Performance is given for different densities (Uniform (U), Gaussian (G) and Exponential (E)) of unit variance and zero mean. The training and test sets consist of 20,000 and 40,000 samples drawn from the respective densities, respectively. A quantizer-source mismatch occurs when the quantizer is used with densities other than those for which it was optimized. $BAR_2(k)$ is used with a fixed learning rate $\eta = 0.01$ for $BAR_2(1)$ and $\eta = 0.1$ for $BAR_2(2)$ and $BAR_2(3)$.

5 Conclusion

A decision tree performs a hierarchical clustering over the input space by means of axis-parallel hyperplanes and, thus, encoding does not require multiplication. In this paper, a competitive learning rule has been introduced in decision trees as a computationally attractive scheme for adaptive density estimation or lossy compression. Because the proposed learning rule updates the active partition cell after each time step, it can be considered as belonging to neural competitive learning. However, it markedly differs from existing competitive learning algorithms, (*e.g* Yair, Zeger & Gersho, 1992; Lee & Peterson, 1990; Ahalt, Krishnamurthy, Chen & Melton, 1990; Kohonen, 1995). Instead of finding the centroids of the partition cells, the proposed learning rule does just the opposite by finding the boundary point vectors that separate the partition cells. Since it adapts boundary points directly, the rule is referred to as Boundary Adaptation Rule ($BAR_r(k)$). As demonstrated by simulation, $BAR_r(k)$ provides an adaptive focusing mechanism capable of tracking time-varying distributions and performs at least as well, at high rate,

as the best vector quantization algorithms while being much faster. We are currently investigating several lines of research, including the potential for hardware implementation, the possibility for gradually changing the overall tree structure by splitting and merging leaf nodes, and the use of general hyperplanes, albeit at the expense of a higher level of complexity.

References

Ahalt S.C., Krishnamurthy A.K., Chen P. & Melton D.E. (1990). Competitive learning algorithms for vector quantization. *Neural Networks*, **3**, 277–290.

Breiman L., Friedman J.H., Olshen R.A. & Stone C.J. (1984). Classification and regression trees. *The Wadsworth statistics/probability series*, Belmont,CA:Wadsworth.

DeSieno D. (1988). Adding a conscience to competitive learning. *Proc. Int. Conf. on neural networks*, (San Diego), **1**, 117–124.

Galanopoulos A.S. & Ahalt S.C. (1996). Codeword distribution for frequency sensitive competitive learning with one-dimensional input data. *IEEE Trans. Neural Networks*, **7**, 3, 752–756.

Gersho A. & Gray R. M. (1992). Vector quantization and signal compression. *Kluwer Academic Publisher*, Boston/Dordrecht/London.

Gersho A. (1979). Asymptotically optimal block quantization. *IEEE Trans. on Information Theory*, **25**, 4, 373–380.

Kassam S. A. (1978). Quantization based on the mean-absolute-error criterion. *IEEE Trans. on Communication*, **26**, 2, 267–270.

Kohonen T.(1995). Self-Organizing Maps. *Springer-Verlag.*

Lee T-C. & Peterson A. M. (1990). Adaptive vector quantization using a self-development neural network. *IEEE Journal on Selected Arears in Communications*, **8**, 8, 1458–1471.

Linde Y., Buzo A. & Gray R.M. (1980). An algorithm for vector quantizer design. *IEEE Trans. on Communication*, **28**, 1, 84–95.

Martinetz T.M., Berkovich S.G. and Shulten K.J. (1993). Neural-gas' network for vector quantization and its application to time-series prediction. *IEEE Trans. Neural Networks*, **4**, 558–569.

Martinez D. & Van-Hulle M. M. (1995). Generalized Boundary Adaptation Rule for minimizing rth power law distortion in high resolution quantization. *Neural Networks*, **8**, 6, 891–900.

Panter P. F. & Dite W. (1951). Quantization distortion in Pulse-Count modulation with nonuniform spacing of levels. *Proc. IRE*, **39**, 44–48.

Van Den Bout D. E. (1989). TInMANN: The integer markovian artificial neural network. *Proc. Int. Conf. on neural networks*, (San Diego), **2**, 205–211.

Yair E., Zeger K. & Gersho A. (1992). Competitive learning and soft competition for vector quantizer design. *IEEE Trans. on Signal Processing*, **40**, 2, 294–309.

Zador P. L. (1966). Asymptotically quantization error of continuous signals and the quantization dimension. *Unpublished memorandum, Bell Laboratories*,

Zador P. L. (1982). Asymptotically quantization error of continuous signals and the quantization dimension. *IEEE Trans. on Information Theory*, **28**, 2, 139–149.