Encoding and Classification in a Model of Olfactory Cortex

Todd Leen, Max Webb and Steve Rehfuss [1]

Dept. of Computer Science and Engineering
Oregon Graduate Institute of Science and Technology
19600 N.W. von Neumann Drive, Beaverton OR 97006-1999

Abstract

We observe that the computational model of olfactory cortex given by Ambros-Ingerson, Granger and Lynch [1] is closely related to multistage vector quantization. Variations of the architecture and learning rules are given. We evaluate the performance of the various models applied to encode and classify vowels extracted from spoken letters.

1 Introduction

Based on physiological and anatomical studies, as well as simulations of a biologically faithful model, Ambros-Ingerson *et al.* [1] propose an abstract model of the computation carried out in olfactory bulb and cortex. This paper identifies that computation as a neural implementation of multi-stage vector quantization [2, 3]. Using speech data, we compare the performance of this scheme with a neural tree-search quantizer and with a traditional competitive learning algorithm. The learning rule is enhanced with a conscience mechanism to insure that neural resources are efficiently used. The biological model posits a constancy of olfactory bulb activity. Although not discussed in [1], it is reintroduced here as a pattern rescaling, and is shown to lead to noise immunity.

2 Models and Learning Rules

2.1 The Biological Model

In the biological model [1, and references therein], layer I and II cortical cells receive input from the olfactory bulb via the lateral olfactory tract (LOT), and from more rostrally located cortical cells. Local inhibitory interneurons divide the cortex into patches. Within each patch, cells compete for activation resulting in winner(s)-take-all activity.

Cues (odors) are sampled repetitively. At each sampling cycle, the synapses to winning cells are incremented according to an LTP rule. Feedback from cortex to bulb selectively inhibits the activity of those bulb (mitral) cells that contribute to the cortical activity. Thus during the next several samples of the same odor, a different set of LOT axons are active and cortex is presented with a different view of the same environmental cue. In addition, an excitatory-inhibitory network maintains roughly constant bulb activity.

[1] T.L. and M.W. are supported by the Office of Naval Research under contract N00014-90-1349. T.L receives additional support under DARPA grant MDA 972-88-J-1004. S.R. is supported by a Wilson Clark fellowship.

0-7803-0164-1/91/0000-0553$01.00©1991 IEEE

2.2 Multi-stage Vector Quantization

The abstract model discards many of the details of its biological counterpart, but retains the features of repetitive sampling, competition, synaptic plasticity, and bulb inhibition through feedback from the cortex. The most striking feature retained by the abstract model is its hierarchical organization.

The system consists of a set of subnets S_i, $i = 1 \ldots D$. Each cell within subnet i has an associated weight vector $w_j^{(i)}$, $j = 1 \ldots m_i$ where m_i is the number of cells allocated to S_i. Input patterns seen by cells in S_i are represented by vectors $x^{(i)} \in R^N$. The first subnet is given the raw environmental input $x^{(1)} = x$ (representing the signal from the olfactory bulb).

At the i^{th} sample of a cue, cells within S_i compete for activation. The winning weight vector is given by

$$w_{win}^{(i)} = \arg \min_{w_j} d(w_j^{(i)}, x^{(i)}), \tag{1}$$

where $d(.,.)$ is the Euclidean (or other appropriate) distance measure [2]. The winning weight vector is moved in the direction of the pattern,

$$\delta\, w_{win}^{(i)} = \alpha\, (x^{(i)} - w_{win}^{(i)}) \tag{2}$$

where α is the learning rate. At the $i + 1^{th}$ sample of the cue, subnet S_{i+1} is fed the input

$$x^{(i+1)} \equiv x^{(i)} - w_{win}^{(i)}, \tag{3}$$

and weights in S_{i+1} compete according to (1) and update according to (2). Subnet S_i performs vector quantization on the patterns $x^{(i)}$. The signal $x^{(i+1)}$ passed to subnet S_{i+1} is just the ***quantization error*** from subnet S_i . Thus each subnet quantizes the quantization errors from the previous subnet.

In the signal-processing literature, this scheme is called *multi-stage vector quantization* [2, 3]. It was introduced to ameliorate the computational requirements of high bit-rate vector quantization.

2.3 Tree-Search Quantization

If the original data is naturally clustered (e.g. according to classes), then the above procedure can be interpreted as a hierarchical clustering scheme. In this case it is desirable to be able to partition clusters independently. The above architecture does not have this capability. To see this, consider two disjoint clusters C_1 and C_2 in the original data space. Assume that at S_1, each cluster is "won" by a different weight $w_1 \leftrightarrow C_1$, $w_2 \leftrightarrow C_2$. After convergence of w_1 and w_2, the cells in S_2 see the union of C_1 and C_2 with their centroids translated to the origin, i.e. two superimposed clusters. Thus these cluster are not independently partitioned by S_2 and later subnets.

This problem can be solved by specifying a tree structure on the subnets. Each weight $w_j^{(i)}$ in S_i, has a specified set of children in S_{i+1}. If weight $w_j^{(i)}$ in S_i wins, then only its designated children in S_{i+1} enter the competition. This architecture ensures that clusters can be independently partitioned, though at the cost of increased storage requirements.

[2] In the original formulation, the competition is based on the inner product $[w_{win} = \arg \max_{w_j} w_j \cdot x]$. The choice in (1) is more compatible with the weight update (2). Kohonen [4] discusses metric compatibility between competition and learning rules.

2.4 Conscience

The unequal use of cells, particularly the occurrence of "dead cells" (cells that do not win on *any* input pattern), is problematic in competitive learning schemes. To alleviate this difficulty, various schemes have been suggested to ensure that each of the cells is utilized. Kohonen's self-organizing feature map [4, 5] overcomes the problem by collecting cells into neighborhoods that are moved about as groups by the learning rule. Grossberg [6] suggests the use of variable-thresholds to overcome the problem.

Here we adopt the frequency-sensitive competitive learning technique given by Ahalt *et al.* [7]. The competition is based on the modified distance

$$\tilde{d}(w_j^{(i)}, x^{(i)}) \;\equiv\; n_j^{(i)}\, d(w_j^{(i)}, x^{(i)}) \tag{4}$$

where $n_j^{(i)}$ is the number of times $w_j^{(i)}$ has won the competition. Cells which dominate the competition acquire large n and are, by (4), effectively farther from the patterns. In this way cells which consistently win are removed from the competition, implementing a "conscience-like" mechanism [8].

This embellishment insures that cells (here cells within each subnet) win with roughly equal frequency. In a flat quantizer this maximizes the output entropy resulting in an optimal code [7, 9].

2.5 Constant Bulb Activity and Pattern Rescaling

In the biological model, each sampling cycle produces roughly constant bulb activity. This implies that the spatial pattern of active LOT axons has about the same number of points for each cue. Taking neurons and synapses as slightly noisy, a useful effect of this constancy is to represent cues on the LOT with similar signal-to-noise ratios.

In the abstract model, repetitive sampling is represented by (3), in which each subnet receives as input the quantization error of the previous subnet. As the weights in the first several subnets converge to their optimal values, the signals fed to subsequent subnets will become successively smaller. With noiseless neurons and connections, and arithmetic of sufficiently high precision this is not a problem. It is, however, unacceptable for analog hardware implementations or for digital hardware implementations using low-precision arithmetic.

The constancy of bulb activity in the biological model can be reintroduced into the abstract model by *rescaling* the vectors (3) presented to each subnet, before competition and learning. To evaluate this notion we map the original data onto the unit hypersphere. Then a convenient family of invertible[3] scaling functions is given by

$$x^{(i)} \to r_\epsilon(x^{(i)})\, x^{(i)} \tag{5}$$

with

$$r_\epsilon(x) = \frac{1+\epsilon}{\|x\| + \epsilon}, \tag{6}$$

and ϵ a positive constant. As $\epsilon \to 0$, $r_\epsilon(x) \to 1/\|x\|$; i.e. rescaling to unit magnitude. As $\epsilon \to \infty$, $r_\epsilon(x) \to 1$; i.e. no rescaling. For finite positive ϵ, (5) maps vectors with norm in (0,1) to vectors of larger magnitude, still in (0,1).

[3] To use the system as a vector quantizer, the rescaling must be invertible so that codewords can be assigned to points in the input space. If the rescaling is not invertible, as is likely in the biological system, one can presumably still use the system for clustering.

To simulate noise and low-precision arithmetic, we perturb calculations (2) and (3), by the addition of a random vector. The competition (1) is perturbed by adding a random vector to $(w - x)$ before calculating $d(w, x)$. The components of the random vectors are uniformly distributed in $[-\beta, \beta]$, with β an experimental parameter.

3 Experimental Results

Experimental data consists of vowels extracted from spoken letters [10]. The data base contains 52 utterances of the letters of the English alphabet, spoken by 150 native English speakers. The data base is divided into five equal parts, each corresponding to 30 speakers. Subsets of the first three are used for training, the fourth and fifth are used for testing the trained networks. The utterances in the test sets are from speakers not included in the training set. As both test sets gave similar results, we report results from only one here.

Each utterance was digitized to 16 bit accuracy at 16kHz sampling rate. A DFT was computed over a 10 ms sampling window, at 3 ms time increments. Our final pattern vectors are the lowest 32 DFT coefficients, time-averaged over the central 1/3 of the vowel. These coefficients span the frequency range from 0 to 4 kHz. The experiments reported here were conducted on the vowels from utterances of the letters *A, E, F, O,* and *R.*

The various models were trained on 449 vowel samples. Convergence was tracked by recording the quantization mean square error (mse) every 10 epochs. The learning rate was initially set at $\alpha = 0.01$ and the networks trained to convergence. The learning rate was then dropped to $\alpha = 0.001$ and training continued to convergence. In general ≤ 100 total epochs sufficed. As an alternative, one could decrease the learning rate according to the requirements of stochastic approximation theory [11].

3.1 Quantization Performance

We report results for networks with three different branching ratios for the tree and multistage architectures. These are compared with a standard one-stage (flat) competitive learning algorithm with equal number of quantization points. Both the tree and multistage networks are three levels deep with branching ratios of 3, 4, and 5. Table 1 gives the quantizer performance on the training data and on one set of 150 test vectors. Table entries with (2σ) error bounds are the means of six experiments with different initial random weight vectors. The remaining entries are from a single experimental run.

Architecture	Points	Training		Test	
		mse	entropy	mse	entropy
Flat	27	0.532±0.007	4.751±0.003	0.614±0.014	4.684±0.012
Tree	27	0.568±0.005	4.747±0.003	0.656±0.018	4.596±0.026
Multistage	27	0.756±0.004	4.501±0.025	0.797±0.009	4.439±0.045
Flat	64	0.370	5.992	0.502	5.742
Tree	64	0.406	5.987	0.549	5.677
Multistage	64	0.646	5.567	0.671	5.409
Flat	125	0.260	6.946	0.481	6.197
Tree	125	0.294	6.939	0.486	6.246
Multistage	125	0.547	6.570	0.582	6.216

Table 1: Quantizer Performance

The table gives both the quantization mse and the output entropy [4]. For branching ratio 3, the tree and multistage topologies have 27 quantization points. The maximum output theoretical entropy is 4.755 bits. For branching ratios 4 and 5, the networks have 64 and 125 quantization points, with maximum output entropies of 6.0 and 6.966 bits.

The tree structure gives consistently lower mse than the multistage topology by a large margin, but slightly higher mse than the flat topology. The output entropy of both the flat and tree structures approaches the theoretical maximum quite closely, indicating nearly equal use of the quantization points. The output entropy for the multistage structure is somewhat lower, as is consistent with the higher mse. The differences between the multistage and flat/tree results seem less pronounced in the test data than in the training data, perhaps indicating differences in generalization.

3.2 Classification Performance

The vowel data has a natural cluster structure, indicated by scatter plots of the first several principal components. Given this, and the presumed discriminatory function of olfactory cortex, it is natural to evaluate the classification power of the different models.

To use the structure as a classifier each quantization point is assigned the class of the majority of training data points that are mapped to it. This class assignment is then used to assign classes to the test data points. A problem arises when classifying test data points that fall into a quantization cell not assigned a class by the training process (this typically happens in the multistage architecture). In this case, we pick one of the "hierarchically closest" quantization cells, and use its assigned class. A quantization cell corresponds to a sequence of winning weights. The hierarchically closest cell to a given cell is the one that shares that longest prefix with that sequence. Results are given in table 2. Classification performance follows the trend in mean square error and output entropy; for any number of quantization points, the flat architecture performs best, closely followed by the tree architecture, with the multistage architecture a distant third. In the table above the tree architecture occasionally outperforms the flat architecture, but this occurs only in the single runs of the 4 × 4 × 4 and 5 × 5 × 5 tree architectures; this result did not reappear with another test set, or in the greater number of runs of the 3 × 3 × 3 architectures.

Architecture	Points	Training	Test
Flat	27	11.804±2.623	12.444±2.514
Tree	27	15.999±1.133	13.222±0.916
Multistage	27	16.481±0.257	17.222±0.497
Flat	64	5.568	10.667
Tree	64	8.463	6.667
Multistage	64	15.367	9.333
Flat	125	6.013	8.667
Tree	125	6.236	6.667
Multistage	125	10.468	14.000

Table 2: Classifier Performance (% error)

We emphasize that this is an unsupervised learning paradigm and the classification results merely reflect the fact that the vowel data is clustered somewhat according to class. In

[4] MSE is the mean of the individual quantization errors, calculated using squared Euclidean distance. Output entropy views the quantizer as a source, and is given by $-\Sigma p(y) log_2(p(y))$, where the sum is over all quantization points, y, and where $p(y)$ is the probability of y being emitted.

comparison, single and multi-layer perceptrons score at $\approx 3\%$ error rate on the same test data.

3.3 Resource Requirements

From an engineering standpoint, we would like to optimize the performance relative to a given cost. Therefore, it is instructive to compare various architectures which are equal, not in the number of quantization points, but in the space resources required. For each architecture, the storage required is proportional to the number of weights in the network.

The flat topology defines M quantization points using $M+1$ weights[5]. A tree with branching ratio B and depth D gives $M = B^D$ at a cost of $(B^{D+1} - 1)/(B - 1)$ weights. In comparison, to define $M = B^D$ quantization points, the multistage topology uses only $B \times D + 1$ weights.

Table 3 compares the performance of the three topologies with (roughly) equal numbers of weights. The multistage architecture has a clear advantage with respect to mse as expected; it generates the most quantization points for the number of weights. (Note that the 13^3 multistage net generates 2197 quantization points, or roughly 5 times as many quantization cells as training data points.) On the test data, the multistage data show an anomalously high classification error compared to mse. This may be related to our error scoring procedure.

Architecture	weights	Training		Test	
		mse	class error %	mse	class error %
Flat	15	0.658	16.704	0.715	12.00
$2 \times 2 \times 2$ Tree	15	0.858	22.049	0.929	18.667
$5 \times 5 \times 5$ Multistage	16	0.547	10.468	0.582	14.000
Flat	41	0.601	11.804	0.581	14.000
$3 \times 3 \times 3$ Tree	40	0.568 ± 0.005	15.999 ± 1.133	0.656 ± 0.018	13.222 ± 0.916
$13 \times 13 \times 13$ Multistage	40	0.317	2.227	0.394	13.333

Table 3: Performance – Equal Resource

3.4 Noise Immunity

To determine the effect of constant bulb activity on immunity to noise, we performed experiments in which varying amounts of noise were injected into the system, and varying amounts of rescaling were done. For these experiments, the input data was initially normalized to lie on the unit sphere. Computations were perturbed by noise with components uniformly distributed in $[-2^{-k}, 2^{-k}]$, as described above. With vectors in the unit ball, this simulates the use of fixed point arithmetic with k bits of precision.

Our initial experiments indicate that the rescaling prescription is indeed effective. Without rescaling the mse deteriorates gradually with increasing noise, with an abrupt increase at a noise level corresponding to about 4–5 bits of precision. At a precision level of 6 bits, rescaling (with $\epsilon = 0$) reduces the mse by about a factor of 2 relative to no rescaling. Choices for ϵ ranging between 0 and 10.0 provide intermediate amounts of noise immunity. For a noise level corresponding to 4 bits of precision, rescaling does provide an improvement in mse, however the algorithm remains severely crippled at this noise level. Our experiments have also shown that choice of rescaling functions, including no rescaling, doesn't affect the convergence time, just the quality of the final result. This may depend somewhat on the weight vector initialization.

[5] The flat structure is implemented as a tree of depth one with a single weight in the first level and branching ratio M. For simulator convenience, the other architectures also have a single weight at the first level.

4 Conclusions

This study demonstrates the efficacy of neural implementations of multistage and tree-search quantization. For fixed branching ratio we have seen that the tree-search quantizer consistently outperforms the multi-stage structure, though at considerable resource cost. For networks with equal neural resource, the multistage architecture returns significantly lower mse than the flat and tree-search architectures. Finally, experiments show that pattern rescaling offers a degree of noise immunity. It is intriguing to us that a model based on physiology is so closely related to an important signal processing technique and that the biological system has apparently developed an architecture of great efficiency.

Acknowledgments

We appreciate stimulating conversation with Dan Hammerstrom, José Ambros-Ingerson and Richard Granger. It is a pleasure to thank our colleague, Ronald Cole, for his interest in this work, and for providing high-quality speech data.

References

[1] Jose Ambros-Ingerson, Richard Granger, and Gary Lynch. Simulation of paleocortex performs hierarchical clustering. *Science*, 247:1344–1348, 1990.

[2] Biing-Hwang Juang and A.H. Gray Jr. Multiple stage vector quantization for speech coding. In *Proceeding of the IEEE International Conference on Acoustics and Signal Processing*, pages 597–600, 1982.

[3] Robert M. Gray. Vector quantization. *IEEE ASSP Magazine*, pages 4–29, April 1984.

[4] T. Kohonen. The self-organizing map. *Proceedings of the IEEE*, pages 1464–1480, 1990.

[5] T. Kohonen. *Self-Organization and Associative Memory, 2nd Edition*. Springer-Verlag, Berlin, 1988.

[6] S. Grossberg. Competitive learning: From interactive activation to adaptive resonance. *Cognitive Science*, 11:23–63, 1987.

[7] Stanley C. Ahalt, Ashok Krishnamurthy, Prakoon Cheen, and Douglas Melton. Competitive learning algorthms for vector quantization. *Neural Networks*, 3:277–290, 1990.

[8] D. DeSieno. Adding a conscience to competitive learning. In *Proceedings of the IEEE International Conference on Neural Networks*, pages 1117–1124, 1988.

[9] John Makhoul, Salim Roucos, and Herbert Gish. Vector quantization in speech coding. *Proceedings of the IEEE*, 73:1551–1589, 1985.

[10] Ron Cole, Yeshwant Muthusamy, and Mark Fanty. The isolet spoken letter database. Technical Report CSE 90-004, Oregon Graduate Institute of Science & Technology, March 1990.

[11] L. Ljung. Analysis of recursive stochastic algorithms. *IEEE Trans. Automatic Control*, 22:551–575, 1977.