To appear in: ICANN '93: Proceedings of the International Conference on Artificial Neural Networks, Amsterdam, The Netherlands, September 13-16, 1993

# Vector Quantization with a Growing and Splitting Elastic Net

Bernd Fritzke
International Computer Science Institute
1947 Center Street, Suite 600
Berkeley, CA 94704-1105
USA
*fritzke@icsi.berkeley.edu*

Abstract

A new vector quantization method is proposed which generates codebooks incrementally. New vectors are inserted in areas of the input vector space where the quantization error is especially high until the desired number of codebook vectors is reached. A one-dimensional topological neighborhood makes it possible to interpolate new vectors from existing ones. Vectors not contributing to error minimization are removed. After the desired number of vectors is reached, a stochastic approximation phase fine tunes the codebook. The final quality of the codebooks is exceptional. A comparison with two well-known methods for vector quantization was performed by solving an image compression problem. The results indicate that the new method is significantly better than both other approaches.

## 1 Introduction

The huge amount of data in many technical applications (e. g. HDTV) makes it necessary to compress the original data before sending them over limited bandwidth channels. Usually correlations and structure in the data are removed to express the contained information using less storage. *Lossless* compression techniques allow a perfect reconstruction of the original information, but are rather limited in their ability to reduce the amount of data. *Lossy* compression techniques are much more effective but lead to a *reconstruction error* when the information is "decompressed".

One specific method of lossy data compression is *vector quantization* (VQ). The object of VQ are data which can be described as $n$-dimensional vectors. A limited set of *representative vectors*, the so-called *codebook*, is used to achieve compression. A given data vector is mapped onto the nearest codebook vector. Only the *number* of that codebook vector is transmitted. Assuming that the codebook is known to the receiver, he can reconstruct the original information to some degree by choosing the corresponding codebook vector.

The main problem in VQ is to find a codebook such that the loss of information caused by coding and reconstruction of the data is small. Equivalently, the mean distance between data vector and nearest codebook vector – the so-called *quantization error* (which *is* the reconstruction error in this case) – has to be minimized.

Many methods have been proposed to find a suitable codebook. The most well-known is the LBG-algorithm from Linde, Buzo, and Gray [6]. Recently also Kohonen feature maps have been used for VQ [7]. In this contribution a new method is introduced which is related to a recently proposed self-organizing network [2] and outperforms the LBG-algorithm as well as the Kohonen feature map w.r.t. the quality of the generated codebooks.

## 2 Principle

We assume from structured data in the sense that our data items are not distributed evenly over the $n$-dimensional space but rather concentrate in certain areas of $R^n$. This kind of distribution is sometimes denoted as *mixture distribution*. It is evident that for uniformly distributed data VQ techniques invariably lead to relatively large quantization errors.

The model we propose consists of a set $A$ of formal units. Every unit $c \in A$ has an associated $n$-dimensional *representative vector* $w_c \in R^n$. The set $W$ of all representative vectors is the current codebook. On the set of units we define a symmetric binary *neighborhood* relation $N \subset A \times A$ having the property that every unit has at most two neighbors:

$$(card\{x \in A | (x, c) \in N\} \in \{0, 1, 2\})(\forall c \in A). \tag{1}$$

In other words the set A is organized as a collection of one or more chains each of which consisting of one or more units.

The general idea of our method is to construct the codebook incrementally by *interpolating* new codebook vectors from existing ones. Interpolation is always done among topologically neighboring units, i.e. units $a$ and $b$ with $(a, b) \in N$. After each interpolation the current codebook is adapted with a fixed number of vectors from the original data.

## 3 Adaptation

One adaptation step can be described as follows:

1. Choose an input vector $\xi$ from the original data.
2. Determine the best-matching unit (bmu) $s$ with the nearest representative vector by

$$\|\xi - w_s\| = \min_{c \in A} \|\xi - w_c\| \tag{2}$$

3. Update the local quantization error of the bmu:

$$\Delta\tau_s = \|\xi - w_s\|^2. \tag{3}$$

4. Adapt the bmu and its direct neighbors:

$$\Delta w_s = \varepsilon_b(\xi - w_s) \tag{4}$$

$$\Delta w_n = \varepsilon_n(\xi - w_n) \quad (\forall n, (n, s) \in N) \tag{5}$$

Thereby $\varepsilon_b$ and $\varepsilon_n$ are constant parameters and $\varepsilon_b \gg \varepsilon_n$ holds. Furthermore for a unit $c$ we denote with $\tau_c$ a local variable used for summation of quantization error. Throughout the paper the notation $\Delta x = y$ stands for $x^{\text{new}} = x^{\text{old}} + y$.

Since not only the best-matching unit, but also its direct neighbors are adapted, topological neighbors in the structure tend to have similar representative vectors. At the same time the reference vectors approach regions with many input signal.

## 4 Insertion

Always after a fixed number $\lambda$ of adaptation steps a new unit is inserted. The local error variables of the units indicate whether the corresponding representative vectors lie in regions of $R^n$ with large mean quantization error. It seems to be a promising strategy to insert new units in those regions. Therefore, the unit $q$ with maximum error variable is determined by

$$\tau_q = \max_{c \in A} \tau_c. \tag{6}$$

Two possible cases can occur: the unit $q$ may have topological neighbors or it may be single due to previous deletion of its neighbors (see below).

If $q$ has one or two direct neighbors, the direct neighbor $f$ with the largest error variable is determined. A new unit $r$ is inserted into the chain between $q$ and $f$. The representative vector of $r$ is interpolated from the vectors of $q$ and $f$ according to

$$w_r = w_q + \gamma(w_f - w_q) \qquad \text{whereby} \qquad \gamma = \frac{\tau_f}{\tau_q + \tau_f}. \tag{7}$$

By interpolating among topological neighboring units chances are good that the new representative vectors also lie in regions with many data vectors and do further reduce the total quantization error. The purpose of the formula for $\gamma$ is to position the new unit more closely (in input vector space) to the unit with the larger error variable. If both $q$ and $f$ have the same error value, $\gamma$ has the value 0.5. If the error value of $f$ approaches zero, then so does $\gamma$ leading in the extreme case to identical representative vectors of $q$ and $r$.

The new unit $r$ has a local error variable, too. This is initialized with a certain non zero value. The idea behind this is that $r$, if it had existed already since the beginning, would probably have got a certain number of those signals which have actually been mapped onto $q$ and $f$. Therefore, it is justified to reduce the error variables of $q$ and $f$ by a certain amount and initialize the error variable of $r$ with the sum of these amounts. The strength of the reduction depends on the distance of the representative vectors and thus on $\gamma$. It is defined through

$$\Delta\tau_q = -\frac{(1-\gamma)}{2}\tau_q, \qquad\qquad \Delta\tau_f = -\frac{\gamma}{2}\tau_f, \tag{8}$$

and the new cell $r$ gets an initial error value according to

$$\tau_r^{\text{initial}} = -(\Delta\tau_q + \Delta\tau_f). \tag{9}$$

so that the total sum of all error variables stays equal. One should note that the amount which is subtracted from the error variables of $q$ and $f$ is indirectly proportional to the distance of the new unit in input vector space and can be at most 50%.

If the unit is single – i.e. has no topological neighbors – a new unit $r$ is created and made a topological neighbor of $q$. The representative vector of $r$ is copied from $q$. Half of the error value of $q$ is subtracted and given to $r$ as an initial error value.

## 5 Deletion

We have assumed a non-uniform distribution of data vectors. It can thus occasionally happen that through the interpolation method described above representative vectors are created with values very far away from all possibly occurring input signals. These vectors do not contribute to the reduction of quantization error and, therefore, the corresponding units are removed. The problem is to decide whether a given vector fulfills this condition. In the case of a finite set of possible data vectors (e.g. in image compression) the following strategy suffices:

- The input signals are chosen cyclically in a fixed order.
- Each unit which has not been best-matching unit for a complete cycle is removed.

This is easily realizable by numbering the input signals and marking at every adaptation step the best-matching unit with the current signal number. To be independent of any order of the input signals, it seems appropriate to generate a fixed *random* sequence of all input signals and use this sequence cyclically.

The removal of units splits the chain into subchains which will eventually also split. However, if the desired codebook size is small compared with the total number of training signals (and otherwise vector quantization does not make sense), insertions of units are much more frequent than deletions. The advantage of deletions lies in the fact that at the end of the process every unit is really used, i.e. has some input signals which are mapped onto it.

## 6 Stochastic Approximation

Once a codebook of the desired size has been generated it is fine tuned by *stochastic approximation* [8]. This denotes a number of adaptation steps with a strength $\alpha$ decaying

$$\text{slowly:} \quad \sum_{t=1}^{\infty} \alpha(t) = \infty, \qquad \text{but not too slowly:} \quad \sum_{t=1}^{\infty} \alpha^2(t) < \infty. \tag{10}$$

One specific sequence of parameters fulfilling the above conditions is the harmonic series

$$\alpha(t) = 1/t \qquad \text{or its more general form} \qquad \alpha(t) = (c+1)/(c+t) \tag{11}$$

for a non-negative constant $c$.

For the stochastic approximation phase of our algorithm the neighborhood relations are completely ignored. Every adaptation step consists thus of:

1. Generation of an input signal $\xi$
2. Determination of the best-matching unit $s$
3. Adaptation of the best-matching unit by

$$\Delta w_s = \alpha(t)\varepsilon_b(\xi - w_s) \tag{12}$$

## 7 Simulation Results

To investigate the quality of the new algorithm, a large number of simulations have been performed. The new method has been compared with two other VQ techniques, the LBG-algorithm [6] and the Kohonen feature map [5].

The test problem was to find a suitable codebook for the well-known "Lenna"-image. We used a version of this picture consisting of $480 \times 480$ pixels each one coded by an 8 bit gray value.

To achieve a good overall compression ratio, we did not apply the VQ-methods directly to the test image. Instead we followed the approach of Schweitzer e.a. [7] and used a multi-layer perceptron (MLP) with linear activation functions to pre-process the picture. We divided the picture in square blocks of $8 \times 8$ pixels and trained an MLP with 64 input units, 16 hidden units, and 64 output units to reproduce these blocks at the output units when they were presented to the input units.

The small number of hidden units forces the network to find an efficient coding of the information in order to be able to reproduce it at the output layer. Baldi and Hornik [1] have proved that in such cases an MLP with linear activation functions performs a transformation equivalent to the optimal Karhunen-Loeve transform. By representing each of the 3600 blocks by the 16-dimensional vector of hidden unit activations, the dimensionality of the original data can be reduced by a factor four.

The resulting vector set was given as input data to the three VQ methods. With every method codebooks of size 256 were determined. Every codebook vector can be identified by an 8 bit number. The whole image thus can be transmitted with only 0.125 bits per pixel (bpp) compared to 8 bits/pixel for the original image.

For both the Kohonen method and the new approach a large number of parameters has been tested in preliminary runs. The final test runs were then made with those parameters giving the best results in the preliminary runs. Different random order of the chosen input vectors provided variations in the results.

The LBG method was always run until convergence. The initial codebook vectors were chosen at random from the input vectors.

Once the codebooks were determined the trained MLP was used to reconstruct the image. By comparison with the original image mean square error (MSE) and signal-to-noise ratio (SNR) were obtained. Moreover, subjective image quality was rated.

The results of our simulations are shown in the following table:

| | # of runs | MSE | SNR (dB) | bits per pixel |
|---|---|---|---|---|
| MLP only | 1 | 29.79 | 31.32 | $\geq 2.000$[1] |
| LBG | 100 | $156.68 \pm 2.2\%$ | $24.11 \pm 0.4\%$ | 0.125 |
| Kohonen map | 200 | $151.09 \pm 3.7\%$ | $24.27 \pm 0.7\%$ | 0.125 |
| growing elastic net | 200 | $138.72 \pm 2.1\%$ | $24.64 \pm 0.4\%$ | 0.125 |

It can be seen that Kohonen's model produced slightly better results than the LBG method. The growing and splitting elastic net, however, reliably produces codebooks with a quantization error more than 10 percent below the error of Kohonen's model. The running time of the two self-organizing networks was about equal in our experiments whereas the LBG-algorithm took considerably longer (about three times). This has, however, to be investigated further.

Good values for the simulation parameters of our model were $\varepsilon_b = 0.24$, $\varepsilon_n = 0.014$, $\lambda = 100$. The stochastic approximation was performed with two to four cycles through all 3600 input signals. The constant $c$ (see eqn. 11) always had the value 2000.

The subjective quality of the reconstructed images correlates often rather well with the quantitative error measures as can be seen in fig. 1. This is, however, not always the case since humans tend to find errors in certain regions (e.g. faces) more disturbing than in others.

## 8 Discussion

We presented a new vector quantization method which incrementally builds up a codebook through interpolation. The new method produces significantly better results for an image compression problem than two other methods it was compared with.

The incremental character of the new method also has the important advantage that the size of the codebook can be increased until the quantization error falls below a given bound. For the other methods a complete restart with a larger codebook would be necessary if the final error was still above that bound.

The presented network is a variant of a more general class of algorithms the underlying principle of which is a growth process controlled by some insertion criterion. In this paper the insertion criterion has been based on the quantization error. Another interesting possibility is to locally count the input signals which leads to networks which can estimate the probability density of the input vectors (see [3]). Recently also a combination of a growing network with a radial basis function network has been proposed using the classification error to guide the growth process [4].

## 9 References

[1] Baldi, P. & K. Hornik, "Neural Networks and Principal Component Analysis: Learning from Examples Without Local Minima," *Neural Networks* 2 (1989), 53–58.

[2] Fritzke, B., "Growing cell structures – a self-organizing network in k dimensions," in *Artificial Neural Networks II*, I. Aleksander & J. Taylor, eds., North-Holland, Amsterdam, 1992, 1051–1056.

[3] Fritzke, B., "Kohonen feature maps and growing cell structures – a performance comparison," in *Advances in Neural Information Processing 5*, L. Giles, S. Hanson & J. Cowan, eds., Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[4] Fritzke, B., "Growing Cell Structures – a self-organizing network for unsupervised and supervised learning," International Computer Science Institute, TR-93-026, Berkeley, 1993.

[5] Kohonen, T., "Self-Organized Formation of Topologically Correct Feature Maps," *Biological Cybernetics* 43 (1982), 59–69.

[1] This depends on the number of bits used to encode a hidden unit activation. If 8 bits are used, the rate is 2 bpp.

MLP transform only, SNR = 31.32, $\geq 2$ bpp

LBG-algorithm, SNR = 24.11 dB, 0.125 bpp

Kohonen feature map, SNR = 24.28 dB, 0.125 bpp

Growing elastic net, SNR = 24.61 dB, 0.125 bpp

Figure 1: Typical reconstructed pictures produced by the various algorithms. The MLP-transform gives an image nearly indistinguishable from the original. The VQ-methods differ in their ability to properly encode this this image at a given bit rate (0.125 bpp).


[6] Linde, Y., A. Buzo & R.M. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communication* COM-28 (1980), 84–95.

[7] Schweizer, L., G. Parladori, G.L. Sicuranza & S. Marsi, "A fully neural approach to image compression," in *Artificial Neural Networks*, T. Kohonen, K. Mäkisara, O. Simula & J. Kangas, eds., North-Holland, Amsterdam, 1991, 815–820.

[8] Wasan, M.T., *Stochastic Approximation*, Cambridge Univ. Press, 1969.