NEURAL NETWORK APPROACHES FOR PREDICTIVE VECTOR QUANTIZATION OF AN IMAGE

Ján Mihalík, Rastislav Labovský

Abstract: The paper deals with a predictive vector quantization of an image based on a neural network architectures, where a vector predictor is implemented by three-layer neural network with various hidden nodes and bias units, sigmoid function as nonlinearity and where vector quantizer is implemented by Kohonen self-organizing feature maps, it means the codebook is obtained by neural network clustering algorithm. We have tested an influence of a number of hidden nodes, various convergence rates of a learning algorithm and a presence of the sigmoid function to a mean square prediction error. Next we have studied an influence of codebook size to a mean square quantization error, that means a performance of predictive vector quantization system for various bit rates. The image of Lena of size 512 × 512 pels was coded for various bit rates, where we have used one-dimensional and two-dimensional vector prediction of the blocks of pels.

Key words: Neural network, vector quantization, vector prediction, Kohonen self-organization feature maps, clustering algorithm, multilayer perceptron

Received: March 14, 2000
Revised and accepted: November 24, 2000

1. Introduction

The predictive vector quantization of an image based on a neural network architectures is a result of vector generalization of DPCM [1], [2], where neural networks are used to implement a vector predictor and a vector quantizer. The task of neural network vector predictor is exploiting of interblock correlations to get residual vectors, which can be then used by the neural network vector quantizer to exploit the correlations between pels of each residual vector. Because the conventional
techniques for design a nonlinear vector predictor are extremely complex and sub-
optimal due to the absence of a suitable model for the source data [6], it was
necessary to investigate new procedures in order to design nonlinear vector predi-
tor. It seems that the neural network (NN) can be a solution of this problem. In the
paper we have designed a three-layer neural network, that was used to implement
a nonlinear vector predictor. The nonlinear neural network vector predictor is deter-
dined by a number of input, hidden and output nodes , then by weights which
connect nodes to each other and by sigmoid function. The weights are obtained by
means of a error back-propagation training algorithm, that tries to minimize the
error between the real and predicted block of pels. We have studied influence of
a number of hidden nodes to a mean square prediction error (MSE_P), have tested
various rates of convergence of the learning algorithm and the dependence of the
MSE_P on the presence of sigmoid functions and values of bias units. Note that the
sigmoid functions are the tools to exploit higher order correlations.

The idea behind vector quantization is to exploit the correlations between pels
of vector by quantization a block of data as opposed to a scalar quantizer that
quantizes each pel independently. The vector quantizer is determined by a code-
book and to obtain it we have used a neural network clustering (NNC) algorithm,
which is not iterative [3]. The vector quantizer which codebook design is based on
the NNC algorithm is known as the Kohonen self-organization feature maps and
it is used because of rapid learning and its codebook design takes much less time
than the classical LBG (Linde, Buzo, Gray) algorithm design [4], [9]. In the NNC
vector quantizer the codebook is represented by the weights associated to the neu-
ral network, in our experiments the codebook is topologically organized (the nodes
are laid on two-dimensional pattern orthogonally). The number of weights of NNC
vector quantizer depends on vector dimension and on the number of reproduction
(quantization) vectors of codebook. The NN vector predictor is trained by using
an error back-propagation learning algorithm and NNC vector quantizer is trained
by using a neural network clustering algorithm.

2. Neural Network Predictive Vector Quantization System

The predictive vector quantization system of an image based on neural networks is
in Fig. 1 that includes coding (a) and decoding (b) subsystems.

The input is a sequence of 16-dimensional vectors, which are obtained from
blocks of pels. We have used two ways of segmentation of images to blocks of
pels as we can see in Fig. 2. For one-dimensional vector prediction we use one
previous block A of 1 × 16 pels to predict the actual block X of 1 × 16 pels, where
the segmentation is made in direction first from up to down and then from left
to right (Fig. 2a) and for two-dimensional vector prediction we use four previous
blocks A, B, C, D of 4 × 4 pels to predict the actual block X of 4 × 4 pels, where
the segmentation is made in direction first from left to right and then from up
to down (Fig. 2b). The block of size 16 pels is a result of compromise between
two facts, the performance of the vector quantizer increases with vector dimension,
but on the other hand the codebook search complexity and storage requirements
increase exponentially with the vector dimension, therefore we have chosen 16-
dimensional vectors. The same optimum size of block for vector prediction is given
experimentally.
Fig. 1 Block scheme of neural network predictive vector quantization system.

Fig. 2 Segmentation of an image to blocks for a) one-dimensional and b) two-dimensional vector prediction where X is predicted block and A, B, C, D are previous blocks.

The conventional methods for design an optimum vector prediction need a lot of mathematical operations to get correlation coefficients which are then used to compute optimum prediction coefficients. It was very difficult task because of an algorithmic character of these methods, mainly if higher order correlations and larger dimensions of vectors are used. This disadvantageous are removed by the new non-algorithmic approach in vector prediction [5], [7], namely neural network vector prediction (NNVP). NNVP are suitable for larger vector dimensions and exploiting higher order correlations. The structure of NNVP is depicted in Fig. 3.
Fig. 3 The structure of NNVP.

The non-linear NNVP is determined by number of input, hidden and output nodes and by two matrices of weights which connect nodes to each other and by sigmoids as well as by values of bias units [5], [8]. The first matrix connects the input layer to the hidden and the second one connects the hidden layer to the output. In the input of one-dimensional NNVP (1DNNVP) we have got 16 + 1 pels and of two-dimensional NNVP (2DNNVP) 64 + 1 pels, where they are multiplied by the first matrix of weights to create the nodes of hidden layer. Then the sigmoid function (SF) is applied to the each node of hidden layer [11]. The same operations are made from the hidden layer to the output one by the second matrix of weights and sigmoid function, so we get the predicted block in the output layer consisting of 16 nodes. Mathematically

\[ \text{nod}_j = \sum_i x_i \cdot w_{ij} \quad h_j = \frac{\exp(\text{nod}_j)}{1 + \exp(\text{nod}_j)} \quad (1) \]

\[ \text{nod}_k = \sum_j h_j \cdot v_{jk} \quad \hat{x}_k = \frac{\exp(\text{nod}_k)}{1 + \exp(\text{nod}_k)} \quad (2) \]

where \(x_i, h_j\) and \(\hat{x}_k\) are input, hidden and output nodes respectively. The \(\text{nod}_j\) and \(\text{nod}_k\) are hidden and output nodes before application of sigmoids. Note that we can use the bias units in the input layer and hidden one. That means the number of nodes of input and hidden layers is increased by one. The main demand for a designer is to obtain weight matrices.

The function of NNC vector quantizer (NNCVQ) is the same as function of conventional vector quantizer. Each input vector is quantized to get a quantization vector which is the nearest vector to the input one in the Euclidean distance sense. Only one difference between NNCVQ and conventional VQ is in the way of obtaining of codebooks [3]. A characteristic feature of NNCVQ is that the similar input vectors are situated closely of each other in a cluster. The performance of NNCVQ is comparable to the performance of the LBG vector quantizer [4], but its learning rate is much higher then learning rate of LBG one.
Each quantization vector of NNCVQ is represented by one node (neuron) of the pattern of Kohonen network (Fig. 4) that is called Processing Element (PE).

The components of quantization vector are the weights \( w_k(i,j) \) connecting the input vector \( e \) to \( PE_{ij} \) which is a representative of the quantization vector. The number of PEs corresponds with a codebook size, for example if we use 4-bit codebook, the number of PEs must be \( 2^4 = 16 \), then the number of weights is \( R \times 16 \) where \( R \) is vector dimension. An idea behind NNC algorithm is that response of a neural network to external stimuli is in excitation of the closest neurons each other, as it is at nervous systems. Generally the patterns can be various (orthogonal, hexagonal, ...) and it is possible to organize nodes at line, at plane or at multidimensional space. In our experiments we have used two-dimensional orthogonal patterns where each PE has its neighboring PEs. The neighborhood of topologically close nodes can have different patterns too and it shrinks with time. When network is trained, the weight updating is made for all nodes at neighborhood, not only for winning PE. The advantages of NNCVQ are considerable mainly if the vector dimension and codebook size are larger values.

3. Design of Vector Predictor Based on Multilayer Perceptron

3.1 Learning equations of NNVP

For design of NNVP we assume the ideal vector quantizer, the mean square quantization error equals to zero, practically it means the design of NNVP is made in opened-loop fashion (the input of NNVP is real image, not reconstructed image as in the closed-loop design). The design of NNVP is based on error back-propagation algorithm, where actual prediction square error is used to update weights between hidden-output and input-hidden layers. Let prediction square error between target \( (x_t) \) and predicted \( (\hat{x}_t) \) vector at the time \( t \) is defined as follows:

\[
D(\hat{x}_t) = \frac{1}{2} \| (x_t) - (\hat{x}_t) \|^2
\]  

(3)

The updating equation for weights between hidden and output layers is given by
\[ v_{jk}^{t+1} = v_{jk}^t - \eta \frac{\partial D(\hat{x}_i)}{\partial v_{jk}^t} \]  \hspace{1cm} (4)

in which \( \eta \) is learning rate constant [13]. For partial derivation above using equation (2) we get

\[ \frac{\partial D(\hat{x}_i)}{\partial v_{jk}^t} = \frac{\partial D(\hat{x}_i)}{\partial \hat{x}_k} \frac{\partial \hat{x}_k}{\partial \text{nod}_k} \frac{\partial \text{nod}_k}{\partial v_{jk}^t} \]  \hspace{1cm} (5)

where partial derivatives

\[ \frac{\partial D(\hat{x}_i)}{\partial \hat{x}_k} = -(x_k - \hat{x}_k) \]  \hspace{1cm} (6)

\[ \frac{\partial \hat{x}_k}{\partial \text{nod}_k} = \hat{x}_k(1 - \hat{x}_k) \]  \hspace{1cm} (7)

\[ \frac{\partial \text{nod}_k}{\partial v_{jk}^t} = h_j \]  \hspace{1cm} (8)

From equations (4), (5) using equations (6), (7), (8) we get an updating equation

\[ v_{jk}^{t+1} = v_{jk}^t + \eta h_j (x_k - \hat{x}_k) \hat{x}_k(1 - \hat{x}_k) \]  \hspace{1cm} (9)

for weights \( v_{jk} \) which connect hidden and output layers [1], [12].

Similarly for weights between input and hidden layers the updating equation is

\[ w_{ij}^{t+1} = w_{ij}^t - \eta \frac{\partial D(\hat{x}_i)}{\partial w_{ij}^t} \]  \hspace{1cm} (10)

For partial derivation using equations (1), (2) we get

\[ \frac{\partial D(\hat{x}_i)}{\partial w_{ij}^t} = \frac{\partial D(\hat{x}_i)}{\partial h_j} \frac{\partial h_j}{\partial \text{nod}_j} \frac{\partial \text{nod}_j}{\partial w_{ij}^t} \]  \hspace{1cm} (11)

where partial derivatives

\[ \frac{\partial D(\hat{x}_i)}{\partial h_j} = - \sum_k (x_k - \hat{x}_k) \hat{x}_k(1 - \hat{x}_k) v_{jk}^t \]  \hspace{1cm} (12)

\[ \frac{\partial h_j}{\partial \text{nod}_j} = h_j(1 - h_j) \]  \hspace{1cm} (13)

\[ \frac{\partial \text{nod}_j}{\partial w_{ij}^t} = x_i \]  \hspace{1cm} (14)

From equations (10), (11) using equations (12), (13), (14) we get

\[ w_{ij}^{t+1} = w_{ij}^t + \eta x_i h_j (1 - h_j) \sum_k (x_k - \hat{x}_k) \hat{x}_k(1 - \hat{x}_k) v_{jk}^t \]  \hspace{1cm} (15)

It is possible to add momentum terms
Mihalík J., Labovský R.: Neural network approaches ...

\[ \mu(v^{t}_{jk} - v^{t-1}_{jk}) \quad \text{and} \quad \mu(w^{t}_{ij} - w^{t-1}_{ij}) \]

to equations (9) and (15), which improve the rate of convergence and smooth the weight changes [1]. The final updating equations are given by

\[ v^{t+1}_{jk} = v^{t}_{jk} + \eta h_j(x_k - \hat{x}_k) \hat{x}_k (1 - \hat{x}_k) + \mu(v^{t}_{jk} - v^{t-1}_{jk}) \]  
(16)

\[ w^{t+1}_{ij} = w^{t}_{ij} + \eta x_i h_j (1 - h_j) \sum_k (x_k - \hat{x}_k) \hat{x}_k (1 - \hat{x}_k) v^{t}_{jk} + \mu(w^{t}_{ij} - w^{t-1}_{ij}) \]  
(17)

Note that \( i = 1, 2, \ldots, I; j = 1, 2, \ldots, J; k = 1, 2, \ldots, R \), where \( I, J, R \) are numbers of input, hidden, output nodes, respectively.

3.2 Error back-propagation learning algorithm

Error back-propagation learning algorithm can be summarized as follows:

**Step 1:** Initialize the network by setting all weights \( w_{ij} \) and \( v_{jk} \) to small random values, set bias units and learning rate constants \( \eta, \mu \) to chosen values, set threshold \( \varepsilon \) to small value and previous mean square prediction error \( MSE^{prev}_p \) to large value.

**Step 2:** Set \( t = 1 \), \( SUM = 0 \).

**Step 3:** Present the input vector \( x_t \) and compute the predicted vector using equations (1) and (2). Compute the square prediction error:

\[ SE_P = ||x_t - \hat{x}_t||^2 \]

**Step 4:** Adjust the weights, starting with \( v_{jk} \) and working back to \( w_{ij} \) using equations (16) and (17), set \( t = t + 1 \) and \( SUM = SUM + SE_P \).

**Step 5:** If \( t \leq T \) go to step 3, where \( T \) is number of input vectors otherwise compute current \( MSE^{cur}_p = SUM/(RT) \) if \( MSE^{prev}_p - MSE^{cur}_p > \varepsilon \)

\( MSE^{prev}_p = MSE^{cur}_p \) and go to step 2 otherwise stop.

4. Vector Quantizer Design Based on Kohonen Self-Organizing Feature Maps

4.1 Mathematical basis of Kohonen networks

Each of PE is subscripted by indexes \((i, j)\) where \( i = 1, \ldots, L \) and \( j = 1, \ldots, M \), that determine its position on orthogonal pattern and they are connected to the input vector by weights \( w_k(i, j) \), where \( k = 1, \ldots, R \) and \( L \times M \) determines the size of codebook. The number of weights \( L \times M \times R \) of NNC vector quantizer depends on vector dimension and on number of quantization vectors of its codebook. The square distance between the input error vector and each vector of weights of PE\((i, j)\) after presenting a new one is given

\[ d^2(i, j) = \sum_k (e_k - w^t_k(i, j))^2 \]  
(18)
For selected PE\((i, j)\) with minimum \(d^2(i, j)\) and for its neighbor PEs the weight updating equation is

\[
w_{k}^{t+1}(i, j) = w_{k}^{t}(i, j) + g(t)[e_{k}^{t} - w_{k}^{t}(i, j)]
\]  (19)

where \(g(t)\) is a gain adaptation function defined as

\[
g(t) = g_0 \exp\left(-\frac{t}{T_g}\right)
\]  (20)

in which \(g_0\) is a constant defining initial value and \(T_g\) is a constant determining the rate of decreasing of the gain adaptation function. The updated neighborhood of winning PE is defined as follows

\[
o(t) = o_1 + o_2 \exp\left(-\frac{t}{T_o}\right)
\]  (21)

where \(o_1, o_2\) are constants defining initial values and \(T_o\) is a constant determining the rate of decreasing of the neighborhood [10]. Note that the most difficult task is obtaining of constants \(g_0, T_g, o_1, o_2, T_o\). The result of NNC learning algorithm depends on all these constants. The values of these constants were decreasing found experimentally.

### 4.2 Neural network clustering learning algorithm

When we have a bit rate \(n\) then the size of codebook is \(N = 2^n\), chose parameters \(L, M\) of orthogonal two-dimensional pattern so that \(L \times M = 2^n\), set the constants \(g_0, T_g, o_1, o_2, T_o\).

**Step 1:** Set weights to small values and \(t = 1\)

**Step 2:** Present input vector \(x_t\) and compute the square distances \(d^2(i, j)\) between the input vector and vectors of weights of PE\((i, j)\) by equation (18)

**Step 3:** Select the node PE\((i, j)\) with minimum \(d^2(i, j)\) and update its weights and weights of neighbor nodes by equation (19) inside of the neighborhood defined by equation (21)

**Step 4:** Set \(t = t + 1\), if \(t \leq T\) go to step 2 otherwise stop

### 5. Experiments and Simulations Results

We have developed predictive vector quantization systems on the basis of neural approaches with two methods of segmentation of the input image of Lena of size \(512 \times 512\) to blocks of 16 pels. For both segmentations we have developed the 1DNNVP and 2DNNVP, where 1DNNVP predicts blocks of \(1 \times 16\) pels and the 2DNNVP predicts blocks of \(4 \times 4\) pels. Note, that for 1DNNVP the one previous block of \(1 \times 16\) pels and for 2DNNVP the four previous blocks of \(4 \times 4\) pels are used for prediction. After obtaining error images we have applied NNCVQ at various bit rates and we have obtained constants needed for NNCVQ design.
5.1 Results of 1DNNVP

We have changed the number of hidden nodes from 5 to 50 with step 5 nodes, \( \eta = \mu = 0,05 \), bias units value = \(-1\), for 2000 iterations through the whole image. The chosen results of optimization for 10, 20, 30, 40, 50 hidden nodes are shown in Fig. 5. MSE\(_P\) for 5 hidden nodes is 92,023 (\( G = 196,372 \)), for 50 hidden nodes MSE\(_P\) is 64,547 (\( G = 273,877 \)), that means the difference of MSE\(_P\) =27,476 and of \( G \) is about 77.5. Note that the \( G \) is predicted gain defined as

\[
G = \frac{\text{MSV}}{\text{MSE}_P}
\]  \hspace{1cm} (22)

and MSV is a mean square value of input image. It is evident from Fig. 5 that utilizing more than 30 hidden nodes did not affect the results significantly. In Fig. 6 there is shown influence of sigmoid function, where two 1DNNVP are depicted. The one is with sigmoid for \( \eta = \mu = 0,05 \) and the other is without sigmoid for \( \eta = \mu = 0,001 \), both for 30 hidden nodes. After 2000 iterations for 1DNNVP with sigmoid MSE\(_P\)=65,616 (\( G = 269,416 \)) and for 1DNNVP without sigmoid MSE\(_P\)=67,369 (\( G = 262,405 \)). It means that the presence of sigmoid improves performance of 1DNNVP in point of view of predictive gain \( G \) about 7. For comparison MSE\(_P\)=66,151 for two-layer linear 1DNNVP.

In Fig. 7 there is shown an influence of various learning rates \( \eta \) and \( \mu \) to a rate of convergence for 1DNNVP with 30 hidden units, bias units value = \(-1\). It seems that higher learning rates are better if the number of iterations is below 5000, because the rate of convergence is high, but we have made experiments up to 30000 iterations, where behind 7500 iterations the achieved MSE\(_P\) is better for lower learning rates.

![Fig. 5 MSE\(_P\) as a function of number of iterations for various number of hidden nodes.](image-url)
Fig. 6 $MSE_p$ as a function of number of iterations for 1DNNVP with and without sigmoid.

Fig. 7 $MSE_p$ as a function of number of iterations for various learning rates.

In Fig. 8 there are predicted and error images of Lena by 1DNNVP for 30 hidden nodes, with sigmoids, bias unit value = −1, after 2000 iterations, $\eta = \mu = 0.05$, $MSE_p=71,306$ and $G = 247,91$. 
Fig. 8 Predicted (a) and error (b) images of Lena by 1DNNVP.

Fig. 9 $MSE_p$ as a function of number of iterations for various number of hidden nodes.

5.2 Results of 2DNNVP

For the same experiments as above, we have changed the number of hidden nodes from 5 to 50 with step 5 nodes, $\eta = \mu = 0.05$, bias units value $= -1$, for 2000 iterations. The chosen results of optimization for 10, 20, 30, 40, 50 hidden nodes are shown in Fig. 9. For 5 hidden nodes $MSE_p=127,698$ ($G = 138,436$) and for 50 hidden nodes $MSE_p=79,260$ ($G = 223,038$), that means the difference of $MSE_p=48,438$ and of $G$ is about 85. It is evident from Fig. 9 that utilizing more
Fig. 10 \( \text{MSE}_p \) as a function of number of iterations for 2DNNVP with and without sigmoid.

Fig. 11 \( \text{MSE}_p \) as a function of number of iterations for various learning rates.

than 40 hidden nodes did not affect the results significantly. In Fig. 10 there is shown influence of sigmoid function. The one is with sigmoid for \( \eta = \mu = 0.05 \) and the other is without sigmoid for \( \eta = \mu = 0.001 \), both for 30 hidden nodes. After 2000 iterations for 2DNNVP with sigmoid \( \text{MSE}_p = 85,268 \) (\( G = 207,322 \)) and for 2DNNVP without sigmoid \( \text{MSE}_p = 117,991 \) (\( G = 149,824 \)). It means that the presence of sigmoid improves performance in point of view of \( G \) about 57.
In Fig. 11 there is shown an influence of various learning rates $\eta$ and $\mu$ to a rate of convergence for 2DNNVP with 30 hidden units, bias units value $= -1$. The predicted and error images of 2DNNVP for chosen parameters are similar to the results of 1DNNVP in Fig. 8., but 2DNNVP achieves lower performances than the 1DNNVP. The predictive gain $G = 247,917$ of 1DNNVP is about 40 higher than $G = 207,488$ of 2DNNVP for the same parameters, i.e. 30 hidden units, bias units $= -1$, $\eta = \mu = 0.05$, with sigmoids, after 2000 iterations. The achieved results show that performance of vector prediction of 1DNNVP is higher than performance of 2DNNVP from the point of view of MSE$_P$ and $G$.

5.3 Results of NN predictive vector quantization system

For next experiments we have chosen simulation of 1DNNVP and 2DNNVP with parameters above. Results of predictive vector quantization system with NNCVQ and 1DNNVP are shown in Tab. I. Note that the bit rates at the first column from top to bottom correspond to the size of the codebooks (orthogonal pattern) $8 \times 8$, $4 \times 8$, $4 \times 4$, $2 \times 4$, $2 \times 2$ codevectors. We investigated an influence of size of NNCVQ codebook to the performance of predictive vector quantization system. The values of constants $g_0$, $T_g$, $o_1$, $o_2$, $T_o$ was found experimentally. In the input of NNCVQ there is a sequence of error vectors with power 71,306. At the 7th and 8th column there are quantization mean square error (MSE$_q$) and quantization signal to noise ratio (SNR$_q$) of NNCVQ depicted. At the last column there is a resulting system signal to noise ratio (SNR$_s$) of the whole predictive vector quantization system.

<table>
<thead>
<tr>
<th>Bit rate (bit/pel)</th>
<th>$g_0$</th>
<th>$T_g$</th>
<th>$o_1$</th>
<th>$o_2$</th>
<th>$T_o$</th>
<th>MSE$_q$</th>
<th>SNR$_q$ (dB)</th>
<th>SNR$_s$ (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.375</td>
<td>0.5</td>
<td>4096</td>
<td>0.0</td>
<td>4</td>
<td>1500</td>
<td>33,677</td>
<td>3,257</td>
<td>32,857</td>
</tr>
<tr>
<td>0.3125</td>
<td>0.7</td>
<td>4096</td>
<td>0.0</td>
<td>6</td>
<td>1000</td>
<td>38,315</td>
<td>2,697</td>
<td>32,297</td>
</tr>
<tr>
<td>0.25</td>
<td>0.5</td>
<td>4096</td>
<td>0.5</td>
<td>1</td>
<td>500</td>
<td>43,814</td>
<td>2,115</td>
<td>31,714</td>
</tr>
<tr>
<td>0.1875</td>
<td>0.1</td>
<td>4096</td>
<td>0.5</td>
<td>1</td>
<td>3000</td>
<td>50,735</td>
<td>1,478</td>
<td>31,077</td>
</tr>
<tr>
<td>0.125</td>
<td>0.1</td>
<td>4096</td>
<td>0.5</td>
<td>1</td>
<td>1000</td>
<td>57,635</td>
<td>0,924</td>
<td>30,523</td>
</tr>
</tbody>
</table>

Tab. I Results of predictive vector quantization system with NNC VQ and 1DNNVP.

For comparison there is an evaluation of predictive vector quantization system with NNCVQ and 2DNNVP in Tab. II. The input of NNCVQ is a sequence of error vectors with power 85,200.

<table>
<thead>
<tr>
<th>Bit rate (bit/pel)</th>
<th>$g_0$</th>
<th>$T_g$</th>
<th>$o_1$</th>
<th>$o_2$</th>
<th>$T_o$</th>
<th>MSE$_q$</th>
<th>SNR$_q$ (dB)</th>
<th>SNR$_s$ (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.375</td>
<td>0.9</td>
<td>4096</td>
<td>0.5</td>
<td>5</td>
<td>1000</td>
<td>28,350</td>
<td>4,778</td>
<td>33,605</td>
</tr>
<tr>
<td>0.3125</td>
<td>0.5</td>
<td>4096</td>
<td>0.0</td>
<td>3</td>
<td>1500</td>
<td>32,830</td>
<td>4,141</td>
<td>32,968</td>
</tr>
<tr>
<td>0.25</td>
<td>0.3</td>
<td>4096</td>
<td>0.0</td>
<td>3</td>
<td>500</td>
<td>38,209</td>
<td>3,482</td>
<td>32,309</td>
</tr>
<tr>
<td>0.1875</td>
<td>0.9</td>
<td>2048</td>
<td>0.5</td>
<td>3</td>
<td>2500</td>
<td>44,517</td>
<td>2,819</td>
<td>31,645</td>
</tr>
<tr>
<td>0.125</td>
<td>0.1</td>
<td>4096</td>
<td>0.5</td>
<td>1</td>
<td>500</td>
<td>54,072</td>
<td>1,974</td>
<td>30,801</td>
</tr>
</tbody>
</table>

Tab. II Results of predictive vector quantization system with NNC VQ and 2DNNVP.
Our expectations was that by quantization of error images the performance of vector quantizer and so the performance of predictive vector quantization system will be higher with 1DNNVP than with 2DNNVP. But the achieved results after quantization showed quite different facts, the performance of predictive vector quantization systems with 2DNNVP was higher. The only one reason of this could be fact that intrablock correlations are higher between components of error vectors for 2DNNVP. Therefore we carried out a correlation analysis of error vector sequences for both cases of NNVP and the values of correlation coefficients ensured us in our assumptions, where correlation coefficients in global point of view was higher for 2DNNVP then for 1DNNVP. The values of some correlation coefficients are in the Tab. III.

<table>
<thead>
<tr>
<th></th>
<th>c_{11}</th>
<th>c_{12}</th>
<th>c_{13}</th>
<th>c_{14}</th>
<th>c_{15}</th>
<th>c_{16}</th>
<th>c_{17}</th>
<th>c_{18}</th>
</tr>
</thead>
<tbody>
<tr>
<td>1D</td>
<td>1</td>
<td>0,618</td>
<td>0,293</td>
<td>0,161</td>
<td>0,095</td>
<td>0,065</td>
<td>0,043</td>
<td>0,022</td>
</tr>
<tr>
<td>2D</td>
<td>1</td>
<td>0,419</td>
<td>0,143</td>
<td>0,081</td>
<td>0,605</td>
<td>0,367</td>
<td>0,135</td>
<td>0,054</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>c_{19}</th>
<th>c_{110}</th>
<th>c_{111}</th>
<th>c_{112}</th>
<th>c_{113}</th>
<th>c_{114}</th>
<th>c_{115}</th>
<th>c_{116}</th>
</tr>
</thead>
<tbody>
<tr>
<td>1D</td>
<td>0,024</td>
<td>0,006</td>
<td>-0,007</td>
<td>-0,02</td>
<td>-0,008</td>
<td>0,02</td>
<td>0,008</td>
<td>0,01</td>
</tr>
<tr>
<td>2D</td>
<td>0,309</td>
<td>0,251</td>
<td>0,136</td>
<td>0,063</td>
<td>0,16</td>
<td>0,15</td>
<td>0,106</td>
<td>0,061</td>
</tr>
</tbody>
</table>

Tab. III Values of correlation coefficients for 1DNNVP and 2DNNVP.

Fig. 12 a) Original image of Lena, b) quantized image by NN predictive vector quantization system with NNCVQ and 2DNNVP at bit rate 0,125 bpp.

If we compare the performances of predictive vector quantization systems, we can see that for bit rate 0,375 bpp the performance with 1DNNVP is 32,857 dB and with 2DNNVP is 33,605 dB, the difference is 0,748 dB, whereas for bit rate 0,125 bpp the performance with 1DNNVP is 30,523 dB and with 2DNNVP is 30,801, the difference is only 0,278 dB. It seems from achieved results that for higher bit
rates it is convenient to use 2DNNVP whereas for lower one it may use simple 1DNNVP. In Fig. 12 there are original image of Lena and its quantized image by NN predictive vector quantization system with 2DNNVP and NNCVQ at bit rate 0,125 bpp.

6. Conclusion

In this paper the neural network predictive vector quantization system as powerful technic for image data compression was proposed and its performance together with those ones of NN vector predictor and NNC vector quantizer were evaluated. In our experiments we have used two methods of neural network vector prediction, the one-dimensional and two-dimensional. We have tested NN vector predictors for the changes of all parameters which could affect their performances. It was found that the one-dimensional NN vector prediction achieves better results than two-dimensional one. Next the error images were quantized by NNC vector quantizer based on orthogonal pattern at various bit rates. The results were unexpected, because the performance of neural network predictive vector quantization system with 2DNNVP was higher than performance of that one with 1DNNVP with higher predictive gain. The achieved results ensure us that the neural network predictive vector quantization system is capable to reduce designing time and computational complexity. The high performance with SNRS over 30 dB at 0,125 bpp of predictive vector quantization system based on neural networks and the high visual quality of encoded images show to a capability of neural network approaches.

Acknowledgement

The work was supported by the Grant Agency of Ministry of Education of Slovak Republic under Grant No. 1/7387/20.

References


