Video Compression Algorithm Based on Neural Network Structures

Michal Knop[1], Robert Cierniak[1,⋆], and Nimit Shah[2]

[1] Institute of Computational Intelligence, Czestochowa University of Technology,
Armii Krajowej 36, 42-200 Czestochowa, Poland
cierniak@kik.pcz.czest.pl
[2] Department of Electrical Engineering
M. S. University of Baroda, Vadodara, India

Abstract. The presented here paper describes a new approach to the video compression problem. Our method uses the neural network image compression algorithm which is based on the predictive vector quantization (PVQ). In this method of image compression two different neural network structures are exploited in the following elements of the proposed system: a competitive neural networks quantizer and a neuronal predictor. For the image compression based on this approach it is important to correctly detect scene changes in order to improve performance of the algorithm. We describe the image correlation method and discuss its effectiveness.

1 Introduction

Multimedia data transmission is widely spread nowadays. Most of the applications require effective data compression in order to lower the required bandwidth or storage space. Various techniques of the data coding achieve this goal by reducing data redundancy. In most of the algorithms and codecs a spatial compensation of images as well as movement compensation in time is used. Video compression codecs can be found in such applications as:

1. various video services over the satellite, cable, and land based transmission channels (e.g., using H.222.0 / MPEG-2 systems [1]);
2. by wire and wireless real-time video conference services (e.g., using H.32x [2] or Session Initiation Protocol (SIP) [3]);
3. Internet or local area network (LAN) video streaming [4];
4. storage formats (e.g., digital versatile disk (DVD), digital camcorders, and personal video recorders) [5].

Currently, many image compression standards are used. The most popular are JPEG and MPEG. They differ in the level of compression as well as application. JPEG and JPEG2000 standards are used for image compression with an adjustable compression rate. There is a whole family of international compression standards of audiovisual data

⋆ Corresponding author.

L. Rutkowski et al. (Eds.): ICAISC 2014, Part I, LNAI 8467, pp. 715–724, 2014.

combined in the MPEG standard, which is described in more details in literature (see e.g. [6]). The best known members are MPEG-1, MPEG-2, and MPEG-4. We used a PVQ (Predictive Vector Quantization) algorithm in our work to compress a video sequence. It combines a VQ (Vector Quantization) [7], [8] and DPCM (Differential Pulse Code Modulation). More information on the techniques can be found in sources [9], [10], [11]. To detect a scene change we used image correlation method. Then we can change necessary parameters of the predictor and the codebook.

2 Video Compression Algorithm

The design of the compression algorithm described here is based on the existing algorithm described in [9–11]. Selected algorithm due to neural network features presents better adjustment to a frame and gives better compression. The extension includes a scene change detection algorithm, which is based on the correlation between frames. The diagram below (see Fig. 1) shows the proposed algorithm.

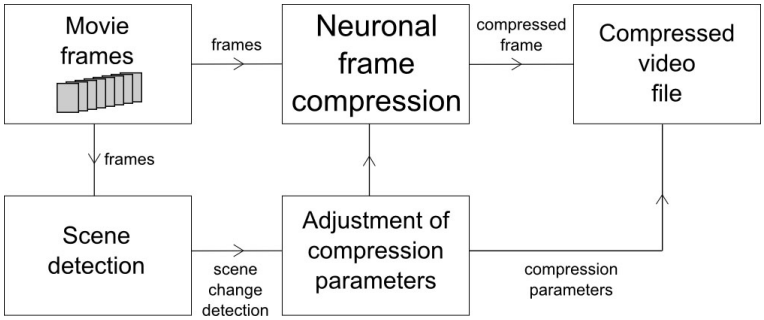


Fig. 1. Video compression algorithm

2.1 Neuronal Image Compression Algorithm

In the literature several methods for image compression have been proposed. Among them the vector quantization (VQ) technique has emerged as an effective tool in this area of research [12]. A special approach to image compression combines the VQ technique with traditional (scalar) differential pulse code modulation (DPCM) leading to the predictive vector quantization (PVQ). In this paper, we develop a methodology where the vector quantizer will be based on competitive neural network, whereas the predictor will be designed as the nonlinear neural network.

We assume that an image is represented by an $N_1 \times N_2$ array of pixels $\mathbf{X} = [x_{n_1,n_2}]$; $n_1 = 1,2,\ldots,N_1$, $n_2 = 1,2,\ldots,N_2$. The image is portioned into contiguous small blocks $\mathbf{Y}(k_1,k_2) = [y_{m_1,m_2}(k_1,k_2)]$ of the dimension $M_1 \times M_2$; $m_1 = 1,2,\ldots,M_1$, $m_2 = 1,2,\ldots,M_2$:

$$\mathbf{Y}(k_1,k_2) = \begin{bmatrix} y_{1,1}(k_1,k_2) & \cdots & y_{1,M_2}(k_1,k_2) \\ \vdots & \ddots & \vdots \\ y_{M_1,1}(k_1,k_2) & \cdots & y_{M_1,M_2}(k_1,k_2) \end{bmatrix}, \tag{1}$$

where we identify: $k_1 = 1,2,\ldots,K_1 = \frac{N_1}{M_1}$, $k_2 = 1,2,\ldots,K_2 = \frac{N_2}{M_2}$.

The arrays (1) will be represented by the corresponding vectors

$$\mathbf{V}(k_1,k_2) = [v_1(k_1,k_2),\ldots,v_L(k_1,k_2)]^{\mathrm{T}}, \tag{2}$$

where: $L = M_1 \cdot M_2$, $v_1(k_1,k_2) = y_{1,1}(k_1,k_2)$, $v_L(k_1,k_2) = y_{M_1,M_2}(k_1,k_2)$. It means that the original image is represented by $\frac{N_1 \cdot N_2}{L}$ vectors $\mathbf{V}(k_1,k_2)$. The successive input vectors to the encoder $\mathbf{V}(t)$; $t = 1,2,\ldots,K_1 \cdot K_2$ correspond to vectors $\mathbf{V}(k_1,k_2)$ in the line-by-line order.

The general architecture of the predictive vector quantization algorithm (PVQ) is depicted in Fig.2. This architecture is a straightforward vector extension of the traditional (scalar) differential pulse code modulation (DPCM) scheme (see e.g. [9, 10]).

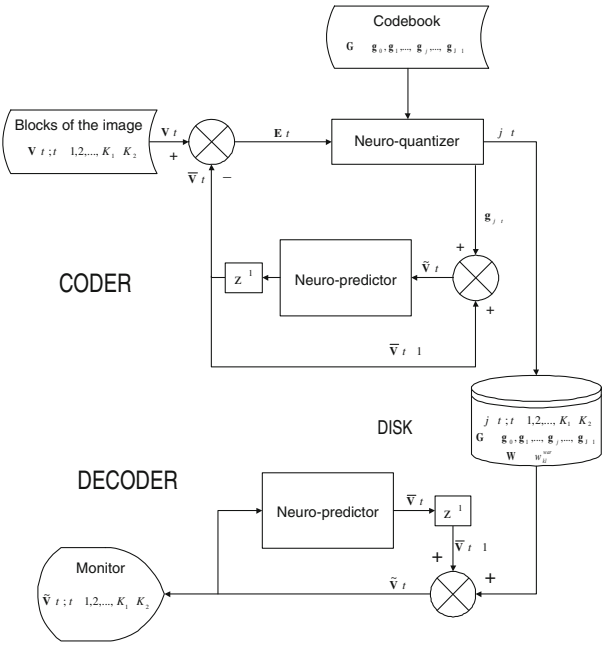


Fig. 2. The architecture of the image compression algorithm

The block diagram of the PVQ algorithm consists of the following elements: encoder and decoder, each containing an identical neural-predictor, codebook and neural vector quantizer. The successive input vectors $\mathbf{V}(t)$ are introduced to the encoder. The differences $\mathbf{E}(t) = [e_1(t),e_2(t),\ldots,e_L(t)]^T$ given by the equation

$$\mathbf{E}(t) = \mathbf{V}(t) - \overline{\mathbf{V}}(t) \tag{3}$$

are formed, where: $\overline{\mathbf{V}}(t) = [\overline{v}_1(t),\overline{v}_2(t),\ldots,\overline{v}_L(t)]^T$ is the predictor of $\mathbf{V}(t)$. Statistically, the differences $\mathbf{E}(t)$ require fewer quantization bits than the original subimages $\mathbf{V}(t)$. The next step is vector quantization of $\mathbf{E}(t)$ using the set of reproduction vectors

$\mathbf{G} = [\mathbf{g}_0, \mathbf{g}_1, ..., \mathbf{g}_J]$ (codebook), where $\mathbf{g}_j = [g_{1j}, g_{2j}, ..., g_{qj}]^T$ (codewords). For every L-dimensional difference vector $\mathbf{E}(t)$, the distortion (usually the mean square error) between $\mathbf{E}(t)$ and every codeword $\mathbf{g}_j, j = 0, 1, \ldots, J-1$ is determined. The codeword $\mathbf{g}_{j^0}(t)$ is selected as the representation vector for $\mathbf{E}(t)$ if

$$d_{j^0} = \min_{0 \le j \le J} d_j, \tag{4}$$

where we can take a measure d in expression (4) as e.g. the Euclidean distance. When adding the prediction vector $\overline{\mathbf{V}}(t)$ to the quantized difference vector $\mathbf{g}_{j^0}(t)$ we get the reconstructed approximation $\widetilde{\mathbf{V}}(t)$ of the original input vector $\mathbf{V}(t)$, i.e.

$$\widetilde{\mathbf{V}}(t) = \overline{\mathbf{V}}(t) + \mathbf{g}_{j^0}(t). \tag{5}$$

The predicted vector $\overline{\mathbf{V}}(t)$ of the input vector $\mathbf{V}(t)$ is made from past observation of reconstructed vector $\widetilde{\mathbf{V}}(t-1)$. In our approach, the predictor is a nonlinear neural network specifically designed for this purpose. In future research we plan to employ orthogonal series nonparametric estimates for the predictor design [13–15], neuro-fuzzy predictor [16–19], and decision trees for mining data streams [20–25].

The appropriate codewords $j^0(t)$ are broadcasted via the transmission channel to the decoder. In the decoder, first the codewords $j^0(t)$ transmitted by the channel are decoded using codebook and then inverse vector-quantized. Next, the reconstructed vector $\widetilde{\mathbf{V}}(t)$ is formed in the same manner as in the encoder (see relation (4)).

2.2 Scene Detection

The parameters of the neural image compression algorithm are strictly determined basing on given compressed image. This comes from the fact that these parameters are established through the learning process of the neural networks applied in this neural compression algorithm. In the case of the video compression, every frame of the film will be processed as a separate image. Unfortunately, that a file containing our compressed film will include an additional information about parameters of this compression. To avoid this situation, we could assign the same compression parameters to several consecutive compressed frame of the video. This concept is based on the assumption that these frames are similar each to other in an acceptable level. Clearly, if these frames are not similar we should use separately for every frame parameters determined for a given frame. For instance, we observed this situation when a change of the scene in the film is encountered. In our concept, we will try to detect the key frame which separates neighboring scenes. Thanks to this idea, we save a space in the file containing compressed film, assigning the same parameters to all frames from a given scene, and we improve quality of the compressed film giving different compression parameters for significantly different frames.

In this context, the scene detection is a crucial problem. Among many other approaches [26–28], the methods based on the correlation coefficient are worth consideration. Correlation coefficient is the number indicating the level of the linear ratio between two random variables. Cuts, gradual transitions, and motion can be distinguished in the video frames using this parameter. For the cuts, the difference between

the two frames is large, and the correlation of these frames is low. For a gradual transition, pixel values of two adjacent frames are different, but are similar in the edges and textures, so the correlation in the spatial domain is high [29]. A histogram of brightness changes slightly for motion scenes that take place on the same background. However, for scenes of gradual transition or cuts, it changes gradually or abruptly.

Differences between objects motion in the scene and the scene change can be obtained by comparison of the key frames with subsequent frames. The key frame histogram H_{kf} can be defined as:

$$H_{kf}(r_k) = n_k, \tag{6}$$

where r_k is the k-th level of brightness, n_k is the number of pixels in frame of the brightness level r_k.

For N frames in the video, we calculate the histogram H_i, i = 2,3, ..., N. The correlation between H_{kf} and H_i can be defined as:

$$corr(H_{kf}, H_i) = \frac{\sum_{j=0}^{m}(H_{kf}(j) - h_{kf})(H_i(j) - h_i)}{\sqrt{\sum_{j=0}^{m}(H_{kf}(j) - h_{kf})^2 \sum_{j=0}^{m}(H_i(j) - h_i)^2}}, \tag{7}$$

where m is the number of brightness scale levels r_k; h_{kf}, h_i are mean values of H_{kf} and H_i [30], respectively, and can be defined as:

$$h_i = \frac{1}{m}\sum_{j=0}^{m} H_i(j). \tag{8}$$

Then, the correlation value computed from Eq. (7) is compared with a threshold. If the correlation value is lower than the assumed threshold, the algorithm determines a new key frame. A diagram of the proposed scene change detection algorithm is shown in Fig. 3.

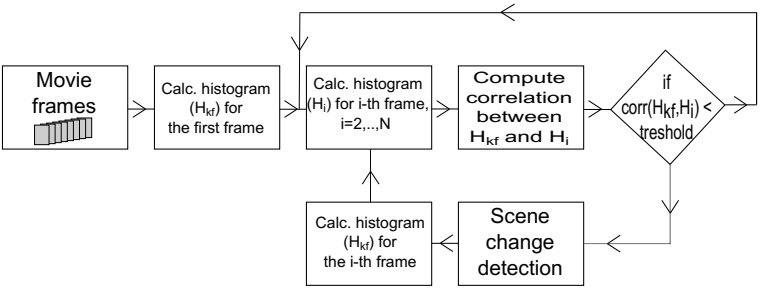


Fig. 3. Scene change detection algorithm

3 Experimental Results

The efficiency of the algorithm was tested on a set of frames extracted directly from a video file of a 576x416 resolution with 256 levels of gray. Four tests were conducted.

In the first and second tests, the frames were compressed within single scene (Fig. 4). In the first test the frames were compressed with a separate codebook and predictor for each of the frames (Fig. 5). For the second test, a single codebook and predictor were used for all frames (Fig. 6).

a) b) c)

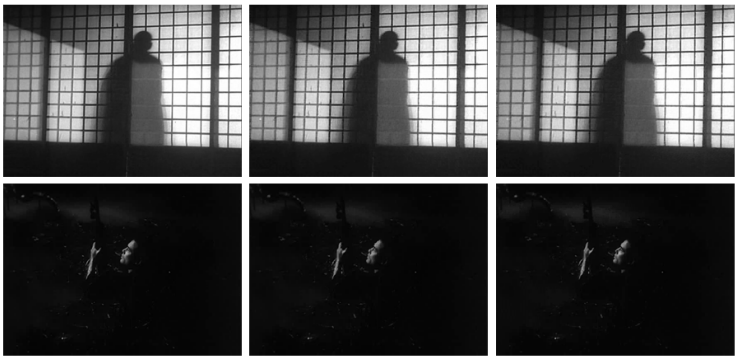
Fig. 4. Original sequence (a); compressed sequence: test 1 (b); compressed sequence: test 2 (c)

Fig. 5. Difference between frames in the test 1

A transit frames between scenes were chosen for the third and fourth tests based on the scene change detection algorithm (Fig. 7). In this algorithm each frame is compared with the keyframe. When the new scene is detected the algorithm marks a new keyframe (see Fig. 8).

In the test 3 the same codebook and predictor were used before and after the scene change. As the results show, this approach is insufficient in case of a major scene change (Fig.9). For the fourth test, the scene transition was detected and separate codebooks and predictors were created for frames before and after the scene transition (Fig.10).

Fig. 6. Difference between frames in the test 2



a) b) c)

Fig. 7. Original sequence (a); compressed sequence: test 3 (b); compressed sequence: test 4 (c)

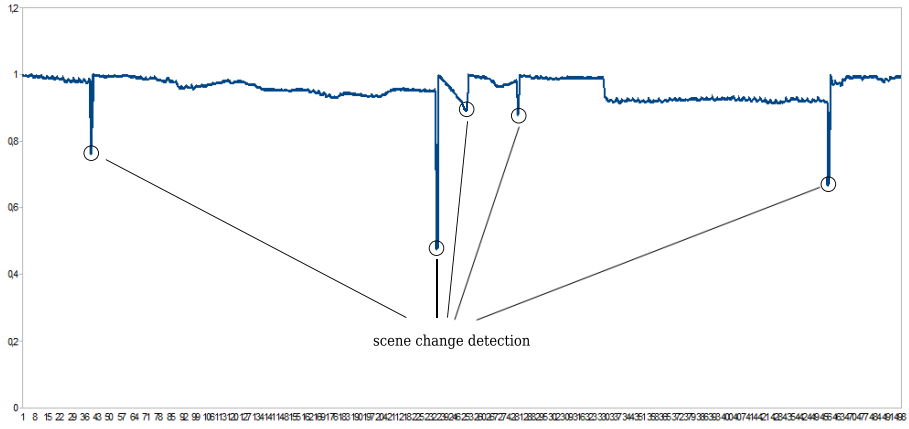


Fig. 8. Scene change detection

Fig. 9. Difference between frames in the test 3

Fig. 10. Difference between frames in the test 4

4 Conclusions

The tests show that the scene change detection algorithm is especially useful for the presented compression algorithm. It is apparent that without the scene detection a video sequence compressed by our algorithm would exhibit a poor quality of frames after the scene transition. On the other hand, the number of data resulting from including the compression parameters for every frame would greatly impact on the output files size.

References

1. Generic coding of moving pictures and associated audio information–Part 1: Systems. Int. Telecommun. Union-Telecommun. (ITU-T), Recommendation H.222.0, MPEG-2 Systems (1994)
2. Narrow-band visual telephone systems and terminal equipment. Int. Telecommun. Union-Telecommun. (ITU-T), Recommendation H.320 (1999)
3. Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., Schooler, E.: SIP: Session initiation protocol. Internet Eng. Task Force (IETF). Request for Comments (RFC), vol. 3261 (2002)
4. Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V.: RTP: A transport protocol for real-time applications. Internet Eng. Task Force (IETF). Request for Comments (RFC), vol. 1889 (1996)
5. Sullivan, G.J., Wiegand, T.: Video compression – from concepts to the H.264/AVC standard. Proceedings of the IEEE 93, 18–31 (2005)
6. Clarke, R.J.: Digital compression of still images and video. Academic Press, London (1995)
7. Gray, R.: Vector quantization. IEEE ASSP Magazine 1, 4–29 (1984)
8. Gersho, A., Gray, R.M.: Vector quantization and signal compression. Kluwer Academic Publishers (1992)
9. Rutkowski, L., Cierniak, R.: Image compression by competitive learning neural networks and predictive vector quantization. International Journal of Applied Mathematics and Computer Science 18, 147–157 (1996)
10. Cierniak, R., Rutkowski, L.: On image compression by competitive neural networks and optimal linear predictors. Signal Processing: Image Communication 15, 559–565 (2000)
11. Cierniak, R.: An image compression algorithm based on neural networks. In: Rutkowski, L., Siekmann, J.H., Tadeusiewicz, R., Zadeh, L.A. (eds.) ICAISC 2004. LNCS (LNAI), vol. 3070, pp. 706–711. Springer, Heidelberg (2004)
12. Fowler, J.E., Carbonara, M.R., Ahalt, S.C.: Image coding using differential vector quantization. IEEE Transactions on Circuits and Systems for Video Technology 3, 350–367 (1993)
13. Rutkowski, L.: A general approach for nonparametric fitting of functions and their derivatives with applications to linear circuits identification. IEEE Transactions Circuits Systems CAS-33, 812–818 (1986)
14. Rutkowski, L.: Identification of MISO nonlinear regressions in the presence of a wide class of disturbances. IEEE Transactions on Information Theory IT-37, 214–216 (1991)
15. Greblicki, W., Rutkowska, D., Rutkowski, L.: An orthogonal series estimate of time-varying regression. Annals of the Institute of Statistical Mathematics 35, 215–228 (1983)
16. Korytkowski, M., Rutkowski, L., Scherer, R.: From ensemble of fuzzy classifiers to single fuzzy rule base classifier. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2008. LNCS (LNAI), vol. 5097, pp. 265–272. Springer, Heidelberg (2008)
17. Rutkowski, L., Przybyl, A., Cpalka, K.: Novel on-line speed profile generation for industrial machine tool based on flexible neuro-fuzzy approximation. IEEE Transactions on Industrial Electronics 59, 1238–1247 (2012)
18. Korytkowski, M., Rutkowski, L., Scherer, R.: On combining backpropagation with boosting. In: IEEE International Joint Conference on Neural Network (IJCNN) Proceedings, Vancouver, July 16-21, vols. 1-10, pp. 1274–1277 (2006)
19. Greenfield, S., Chiclana, F.: Type-reduction of the discretized interval type-2 fuzzy set: approaching the continuous case through progressively finer discretization. Journal of Artificial Intelligence and Soft Computing Research 1, 193 (2011)

20. Pietruczuk, L., Duda, P., Jaworski, M.: A new fuzzy classifier for data streams. In: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2012, Part I. LNCS (LNAI), vol. 7267, pp. 318–324. Springer, Heidelberg (2012)
21. Jaworski, M., Duda, P., Pietruczuk, L.: On fuzzy clustering of data streams with concept drift. In: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2012, Part II. LNCS (LNAI), vol. 7268, pp. 82–91. Springer, Heidelberg (2012)
22. Rutkowski, L., Pietruczuk, L., Duda, P., Jaworski, M.: Decision trees for mining data streams based on the McDiarmid's bound. IEEE Transactions on Knowledge and Data Engineering 25, 1272–1279 (2013)
23. Pietruczuk, L., Duda, P., Jaworski, M.: Adaptation of decision trees for handling concept drift. In: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2013, Part I. LNCS (LNAI), vol. 7894, pp. 459–473. Springer, Heidelberg (2013)
24. Rutkowski, L., Jaworski, M., Pietruczuk, L., Duda, P.: Decision trees for mining data streams based on the gaussian approximation. IEEE Transactions on Knowledge and Data Engineering 26, 108–119 (2014)
25. Rutkowski, L., Jaworski, M., Pietruczuk, L., Duda, P.: The CART decision tree for mining data streams. Information Sciences 266, 1–15 (2014)
26. Chen, L., Feris, R., Turk, M.: Efficient partial shape matching using Smith-Waterman algorithm. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPRW, pp. 1–6 (2008)
27. Adhikari, P., Gargote, N., Digge, J., Hogade, B.G.: Abrupt scene change detection. World Academy of Science, Engineering and Technology 18, 687–692 (2008)
28. Seeling, P.: Scene change detection for uncompressed video. In: Technological Developments in Education and Automation, pp. 11–14 (2010)
29. Li, Z., Liu, G.: A novel scene change detection algorithm based on the 3D wavelet transform. In: IEEE International Conference on Image Processing, ICIP, pp. 1536–1539 (2008)
30. Radwan, N.I., Salem, N.M., El Adawy, M.I.: Histogram correlation for video scene change detection. In: Wyld, D.C., Zizka, J., Nagamalai, D. (eds.) Advances in Computer Science, Engineering & Applications. AISC, vol. 166, pp. 765–773. Springer, Heidelberg (2012)