BULETINUL Universităţii Petrol – Gaze din Ploieşti	Vol. LXI No. 3/2009	49 - 58	Seria Tehnică

Image to Sound Conversion: a Neural Approach

Adrian G. Moise[*], Adrian Ionuţ Constantin[**]

[*] Petroleum – Gas University of Ploieşti, 39 Bucureşti Blvd., Ploieşti, ROMÂNIA
e-mail: amoise@upg-ploiesti.ro

[**] Yokogawa Europe B.V. – Romania Branch, Bd. Dimitrie Pompeiu Nr.6, 020337 Bucureşti
e-mail: Adi_constantin@gmail.com

Abstract

A musical score has different appearance, from printed to hand written and a system to convert the image of such a score to a standard image and/or to a sound file could be useful in many areas such as: control, robotics, computer vision. The purpose of this paper is to develop a method to obtaining a direct conversion from a musical score to a sound file. The paper presents the design methodology of an artificial neural network used to recognize musical notes. The image of the musical score is captured using a web camera or it is directly loaded from a file. One of the main contributions of the authors is extracting the features of the musical score and exporting them to the artificial neural network. Then, the input/output codes are described, the network is trained and the results of different training methods with different neurons are compared. The musical notes obtained by using these methods are displayed and the musical score is played. The paper ends with conclusions and recommendation for future research.

Key words: *artificial neural network, musical note, training algorithm.*

Introduction

Artificial neural networks have known until now periods with extreme activity and periods with disappointing results. It seems that the first decade of 21st century is a period in which researche focuses more on practical applications in very diverse areas.

Starting with Hermann von Helmohltz [4] and Pavlov [8] who developed theories about learning, going on with Hebb [2, 3] who enunciated the principle of synaptic plasticity, to Kohonen [6] and Hopfield [5] who developed new structures and training methods, all periods have known practical applications for artificial neural networks. In the last decade, the main domains in which artificial neural networks proved their utility and efficiency are functions approximations, data classifying, pattern recognition, shape recognition, vocal identification, industrial process control, robotics, and financial prediction.

An interesting field for artificial neural networks is medicine, with applications like automatic diagnosis.

This paper can be included in the area of pattern recognition and automatic image to sound conversion.

Musical theory basic elements

The following elementary music theory elements [10] are necessary for the reader to understand the objective of this paper.

Stave. The stave is used as a background for the musical notes. 11 notes can be written on the stave, one note on each line and one note in between each adjacent lines.

Musical note. Each note on the stave has its own name. The English names for the main seven notes are C, D, E, F, G, A and B. They respectively correspond to the Eastern European names of Do, Re, Mi, Fa, Sol, La and Si. Do is sometimes called Ut, Sol is So and Si is Ti. An example of a simple musical stave is shown in Fig. 1. This paper is dealing with this kind of stave.

Note lengths. The length (duration) of a note means how long it lasts. In order to know or to find out the length of a note one can look at the tempo and the time signature and see what the note is like. In this paper we have taken into consideration four basic lengths which are represented in Fig. 2 together with the relationship between them. They are: the whole note, the half note, the quarter note and the eighth note. Other lengths can be added but the authors considered these enough in order to demonstrate the ideas.

Sound characteristics

Pitch. This is how high or low a note sounds. Pitch is related to the frequency of the fundamental sound wave of the note. Therefore, the sound pitch is always considered in respect to a reference point on the musical notes scale. Musicians do not use frequencies so they gave names to the musical notes. The length of a note is considered from the moment of sound impact to the last perceived vibration.

Intensity (loudness). This is also a subjective characteristic and a relative term. It is measured by the term called amplitude (in Physics) and it means how strong or weak a sound is. Intensity is directly proportional to the square of the amplitude: if the amplitude doubles, the intensity increases 4 times. For a human ear to perceive differences in intensity, this have to change at least 1.25 times.

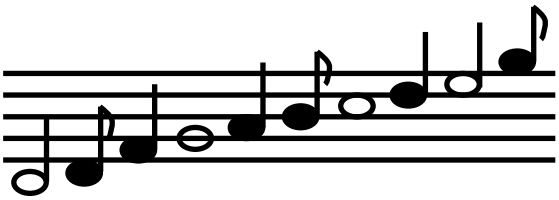

Fig. 1. Example of a stave.

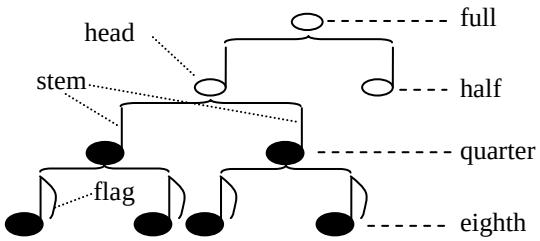

Fig. 2. Relationships between note lengths.

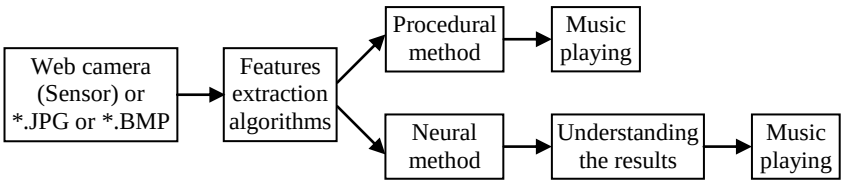

Fig. 3. Structure of the recognition system.

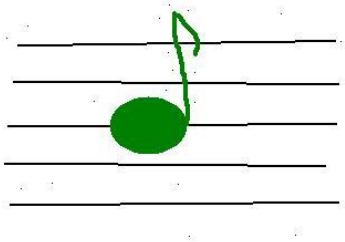

Fig. 4. The recognition algorithm will be applied to this image.

Note features

The structure of the recognition system is shown in Fig 3. The authors suggest the following phases to solve the proposed problem: acquiring an image using a web camera or loading the image from a file; identifying the stave lines of the current scale and deleting them from the image; identifying the properties of each note and erasing the current note; identifying the notes by using the procedural method; exporting the characteristics to the neural network; training the net using the training set; testing the network; displaying the notes and playing the scale. Finding the characteristics is the main processing step and it has as an objective identifying the properties of each musical note. This algorithm includes the image pre-processing and extracting the note properties. This step is important because it does not contain redundancy elements and if the step of extracting the note properties fails, the whole program will be affected. In order to avoid failure the following input data should be accepted.

Input data. The program accepts as input data an image, *.bmp or *.jpg, that will be processed to obtain the characteristics. The authors took into consideration two ways for loading the image into the program: a) a *.bmp or *.jpg file with a 320 x 240 pixels resolution is loaded; b) a web camera is used to capture the scale drawn on a piece of paper (preferable white paper). An example of an image that could be accepted is shown in Fig 4. One can see some noise pixels, stem, and flag are not „standard" and stave lines are not exactly parallel and the note is green.

The image has to have the following properties: to have a five line stave, there are not overlapped notes (or two, three voices), the distance between two consecutive notes is at least one note (or, it can be setup in the program).

Image pre-processing

Image conversion to binary (bitmap). Binary images contain only two gray levels: white and black. Such an image requires only one bit per pixel to represent it and it can be stored in a very compact form if the bits are packed eight to a byte. The first step after acquiring the image is to obtain the black and white version of it. The conversion is made as follows:

Read each pixel of the image on rows and columns and identify the local color levels (red – R, green – G, blue – B). Then, convert the color to grayscale according to the effective luminance of a pixel formula [7]

$$Y = 0.3 \cdot R + 0.59 \cdot G + 0.11 \cdot B \tag{1}$$

Then, the grayscale image is converted to binary using a threshold procedure: looking at each pixel gray level and decide whether it should be made white (255) or black (0). The decision is made by comparing the pixel gray level with a given threshold. If the pixel value is less than the threshold, the pixel is set to 0, otherwise it becomes 255. After this step, the scale is converted to black and white, as shown in Fig. 5.

Noise rejection. When images are captured using a web camera, there will be noise pixels or groups of pixels due to scale light irregularities or incorrect conversion. If these pixels would not be rejected, they would create problems in notes identification. The noise rejection function

eliminates all the black pixels that have 4-connected and 8-connected white pixel neighbors. The result of running this function is shown in Fig. 6.

Scale characteristics

Stave lines. After the noise rejection is done, the next step is enclosing the image into an area bordered by two vertical lines. That means, all the lines will have the same length after the procedure is applied. The authors called this process identifying the start and stop points. The algorithm is the following: read the image columns upwards from the lower left corner. When five consecutive transitions from 1 to 0 and five consecutive transitions from 0 to 1 will be found, the corresponding vertical line will be considered. Similarly, to get the left-hand sideline, the columns will be read from the lower right. After applying these procedures, the original image will be bordered by the two lines just found, as in Fig. 7.

The authors used the code below to find the left vertical line. A similar program segment was used to find the right vertical line.

Identifying and correcting the stave lines. To identify the lines the left vertical line that marks the beginning of the five lines is read as well as the line that marks the end of the valid image. From the first reading, the start and end coordinates of each stave line are found. From a second reading, the same coordinates are extracted and one should verify that they are identical to the first ones. If there are differences between the two readings, the following rule is applied: if the bottom border of a stave line has the coordinate Y (in the coordinate system xy) less than the coordinate Y discovered in the preceding step (the coordinates $x=0$, $y=0$ represent the upper left corner of an image) then the Y coordinate of the line is the old coordinate, otherwise Y is the new coordinate. The same algorithm applies for the upper frontier. The algorithm applies for all the lines and so the parallelism of the lines will be obtained. Two functions have been developed to realize these operations: `void gaseste_frontiere_stanga()` and `void gaseste_frontiere_reunite();`

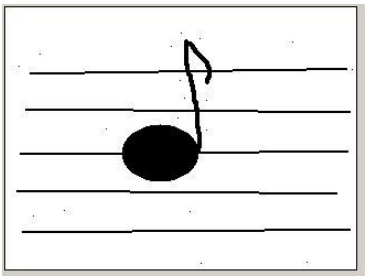

Fig. 5. The initial image is converted to black and white.

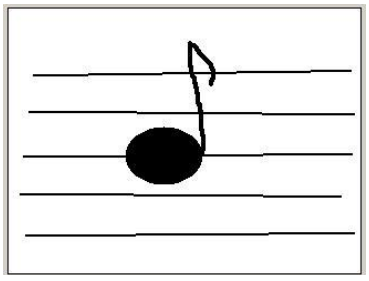

Fig. 6. Noise rejection applied to the initial image.

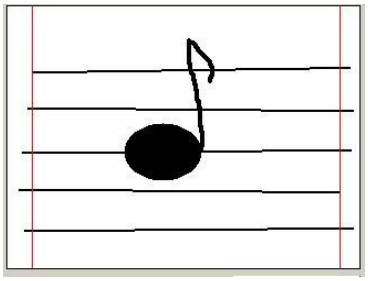

Fig. 7. Finding the lines that border the initial image.

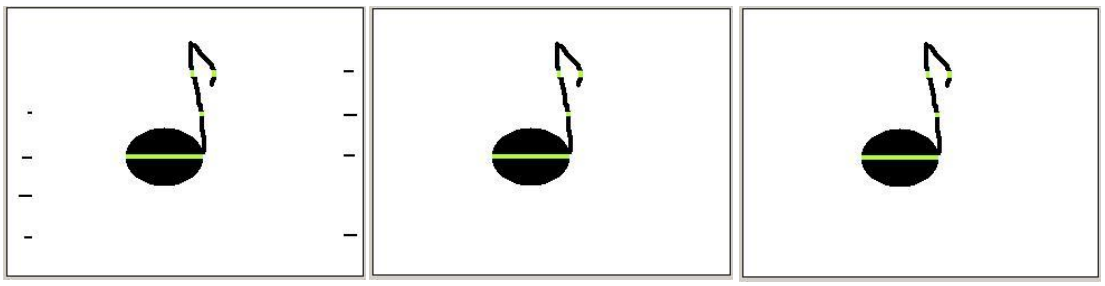

Fig. 8. Mark and delete the stave lines.

Marking and deleting the stave lines. To help identifying the notes, the authors considered necessary that after the stave lines have been identified, corrected, and marked, they have to be deleted from the image but without affecting the notes integrity. Deleting the lines does not mean deleting their coordinates in the memory. In Fig. 8 (on the previous page), this process is shown for the example taken into consideration before.

```
void gaseste_punct_de_start()
        {
          front_pozitiv = 0;
          front_negativ = 0;
  for(index_x=0; index_x <= rezolutie_x; index_x++)
  {
  for (index_y = rezolutie_y ; index_y > 0; index_y--)
     {
       if (matrice[index_x, index_y] == 0 && matrice[index_x, index_y-1] == 1)
                front_pozitiv++;
       if (matrice[index_x, index_y] == 1 && matrice[index_x, index_y-1] == 0)
                front_negativ++;
       if (front_pozitiv == 5 && front_negativ == 5)
              {
                punct_de_start[0] = index_x;
                dialog("Start point has been found ;");
               //stop the loop
             index_y = 0;
             index_x = 320;}
       if (front_negativ > 5 || front_pozitiv > 5)
              {
                  index_y = 0;
                  front_negativ = 0;
                  front_pozitiv = 0;}}
              // verify the start point
           if (front_pozitiv > 5 || front_negativ > 5)
            MessageBox.Show("More than 5 edges have been discovered
            for the start point! ");
        }
          punct_de_start[1] = rezolutie_y;
}
```

Identifying characteristics. To identify a note one should find some characteristics that define the note. One of them is the stave line that is a relationship with the note (Although the stave has five lines, in the program should appear seven stave lines, numbered from 0 to 6, two are imaginary lines, because the total number of notes that can be identified is 13, and they can be represented on almost two octaves.) There are two kinds of relationships that can exist between notes and lines: the note is on the line n or under the line n. Another characteristic is given by the note head: full head or empty head. The flag gives the third characteristic: note with or without flag.

In order to find the line interacting to the note, one should identify the following points: the left end, the bottom end, the right end, and the upper end of the note. To find the left end of a note, the image is read starting with the lower left corner, on columns, until a 1 (black) pixel is found. Because there is a possibility to be many pixels (a segment) on the left end of the note, the pixel in the middle of the segment is kept. To find the bottom end of a note, the image is read starting from the left end to the right, row by row. The reading ends when the y coordinate of a pixel is smaller than the previous y coordinate.

The relative center of the note is found by using the following reasoning. Since we have the left end (x_s, y_s) and the bottom end (x_j, y_j), the center of the note will have the coordinates (x_j, y_s). By finding the center of the note one can identify if the note head is full or empty: if the central pixel has the value 1, the note has a full had, otherwise the head is empty. The upper end and the right end of the note can be found by reading the image starting

from the center of the note in a vertical direction (for the upper end) and in a horizontal direction (for the right end).

After all these algorithms are applied, the five characteristic points of a note are found, as it is shown in Fig. 9.

The flag and the stem. To find the length of a note (stem and flag) the image is read according to the representation in Fig. 10.

The blue points represent positive edges while the green points represent negative edges. If in the end of the reading the maximum number of positive edges followed by negative edges equals 2, then the note has a flag, if it equals 1, the note has only a stem.

The characteristics will be converted into binary as follows.

- the first 7 bites represent the line number which interacts with the note. For example, if the note interacts with the line 3, then the binary number will be 0001000.

- bit 8 represents the note position against the interacting line. The value is 1 if the note is on the line, or 0 if the note is under the line.

- bit 9 represents the head of the note. It is 1 if the note has full head, 0 if it has an empty head.

- bit 10 represents the stem. It is 1 if the note has a stem, 0 if it has not.

- bit 11 represents the flag. It is 1 if the note has a flag, 0 if it has not.

The characteristics for the note in the example above are 00010001111.

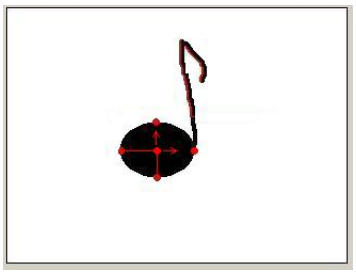

Fig. 9. Identifying the ends and the center of a note.

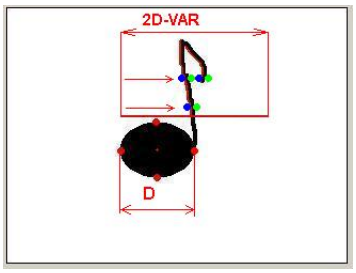

Fig. 10. Identifying the length of a note.

Network design

In this paper, the authors focused their research on using feed forward artificial neural networks to identify the musical notes.

The 11 features mentioned above will be inputs for a totally interconnected neural net which has 11 neurons on the input layer, a hidden layer with 100 neurons and 2 neurons on the output layer, as shown in Fig. 11. The activation function for the neurons in the output layer was the linear function.

The back propagation algorithm [1, 7, 9] was used to train the network and a fragment of the training set is shown in Table 1. For example, 1000000 represents the note DO. Output $t1$ indicates the note number (from 1 to 13, see Fig. 1) and the output $t2$ gives the note length (a real number 1 for a whole note, 0.5 for a half note, 0.25 for a quarter note or 0.125for an eighth note).

The training algorithm was used to train the network with different activation functions (sigmoid, radial, hyperbolic tangent, saturation function, etc.). Some of the results are shown in Table 2 and in Fig. 12 and Fig. 13.

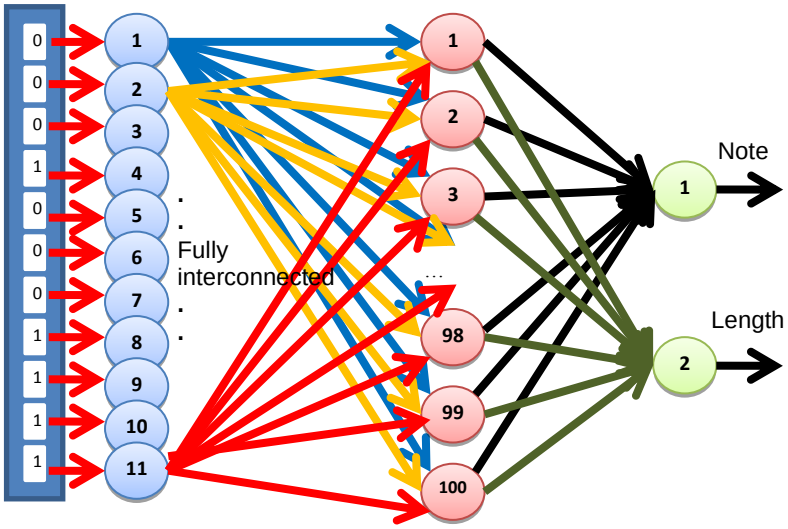

Fig. 11. The network used to recognize musical notes.

Table 1. The training set for the network

Note	Length	Line 1	Line 2	Line 3…6	Line 7	Position	Full head	Stem	Flag	t1	t2
do	full	1	0	…	0	1	0	0	0	1	1
	…….										
	eighth	1	0	…	0	1	1	1	1	1	0.125
re	full	0	1	…	0	0	0	0	0	2	1

Table 2. Target and output for different activation functions

Input	Target		Output / sigmoid	
00010000010	6	0.5	6.0007	0.500006
00001000010	8	0.5	7.99466	0.475832
00000100010	10	0.5	10.0041	0.487419
			Output / radial	
00010000010	6	0.5	6.01118	0.519833
00001000010	8	0.5	8.00092	0.508911
00000100010	10	0.5	10.0054	0.488395

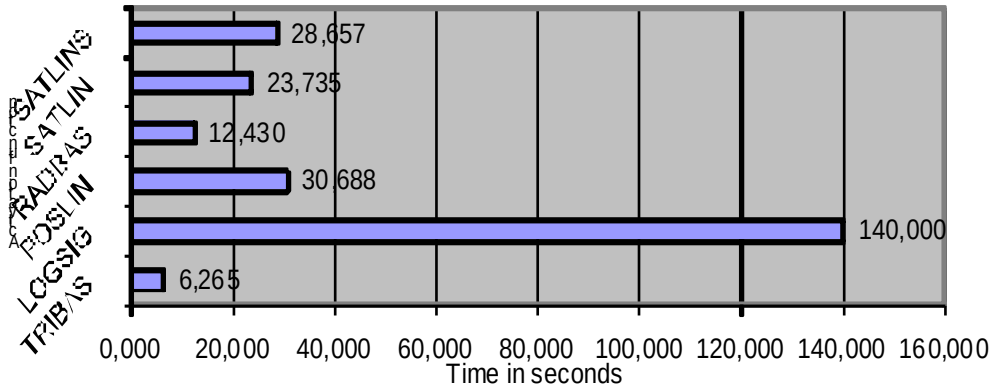

Fig. 12. The necessary time to train the network.

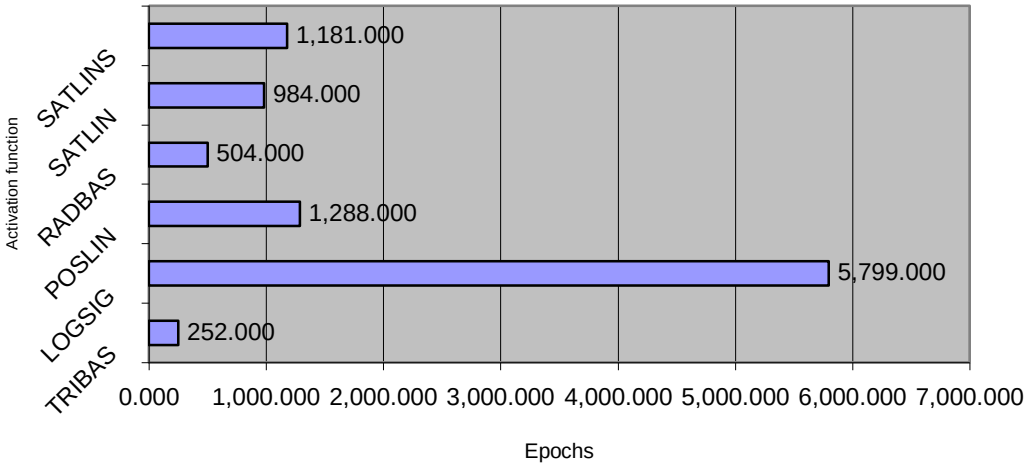

Fig. 13. The number of epochs to train the network.

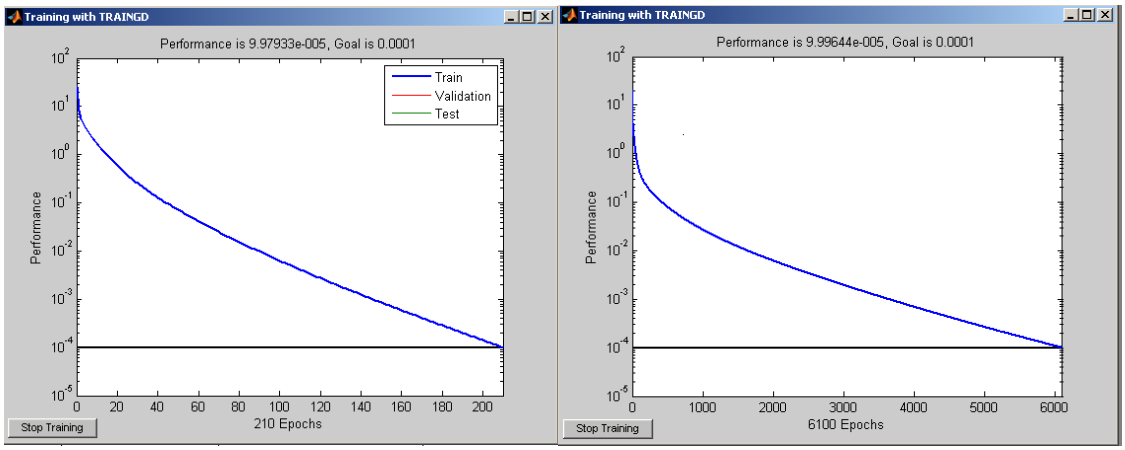

Fig. 14. Error evolution with a triangular and sigmoid activation function.

Twelve different activation functions have been used for the hidden layer neurons and only six ended the training process. They were sigmoid, radial basis, hyperbolic tangent, triangular basis, linear saturation, and linear symmetrical saturation. The triangular basis was the fastest and the sigmoid was the slowest (Fig. 14).

After comparing the results with different activation functions, different algorithms were taken into consideration. The fastest algorithm was `traingdx` (gradient descent with momentum and variable learning rate) and the slowest was (gradient descent with variable learning rate).

Playing a scale. The above described method was used to play the two scales shown in Fig. 15.

The features extracted for the seven notes are shown in Table 3 and the results obtained after training the network are shown in Table 4. One can see that errors are in the range of 0.01 and the results are very accurate.

Conclusions

Artificial neural networks are useful structures for identification applications. The authors developed such an application for musical notes recognition and playing scales directly from a video image. The main contributions of the paper are the algorithms developed to find the 11

features of the notes. These features have been used as inputs for a feed forward net. There have been taken into consideration many activation functions and many training algorithms and comparative experimental results are given in the text. Errors in recognition were very small and the recognition was very accurate.

There are still some difficulties when implementing such a system. One of them is the capture and scale binary representation in the computer memory. Although the scale is ideal, that means the stave lines are parallel, this scale will not be the same in memory. For example, the lines will not be parallel. This is due to some external elements such as the vision angle, illumination, camera resolution, etc. The influence of these factors is minimized in the pre-processing stage.

This work can be continued by considering other artificial structures that could be better used for recognizing and playing more difficult scales with much more stiles of musical notes.

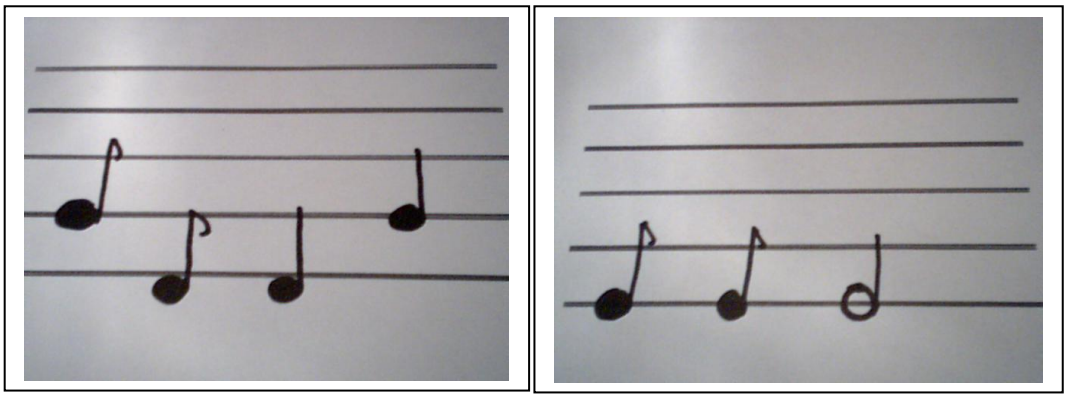

Fig. 15. Two scales to be recognized and played.

Table 3. Features extracted for the seven notes in Fig. 15

Note 1	0	0	1	0	0	0	0	1	1	1	1
Note 2	0	1	0	0	0	0	0	0	1	1	1
Note 3	0	1	0	0	0	0	0	0	1	1	0
Note 4	0	0	1	0	0	0	0	1	1	1	0
Note 5	0	1	0	0	0	0	0	1	1	1	1
Note 6	0	1	0	0	0	0	0	1	1	1	1
Note 7	0	1	0	0	0	0	0	1	0	1	0

Table 4. The network output after training

Note no.	**Note value**	**Length**
Note 1	4.99868 – sol	0.119873 – eighth
Note 2	1.99765 – re	0.131964 – eighth
Note 3	2.00323 – re	0.229565 – quarter
Note 4	4.98409 – sol	0.253838 – quarter
Note 5	2.99876 – mi	0.122752 – eighth
Note 6	2.99876 – mi	0.122752 – eighth
Note 7	2.99876 – mi	0.494526 – half

References

1. Chauvin, Y., Rumelhart, D. E. – *Backpropagation: Theory, Architecture, and Applications*. Lawrence Erlbaum; 1st edition, 1995.
2. Hebb, D.O. – *The organization of behavior*. New York, Wiley.
3. Hebb, D.O. – *Distinctive features of learning in the higher animal*. J. F. Delafresnaye (Ed.). *Brain Mechanisms and Learning*. London: Oxford University Press, 1961.
4. Hermann von Helmholtz – *On Goethe's Scientific Researches* (1853) and *On the Aim and Progress of Physical Science* (1869). In: Helmholtz, Science and Culture, David Cahan, ed. Chicago: University of Chicago Press, 1-17 and 204-225, 1995.
5. Hopfield, J.J. – *Neural networks and physical systems with emergent collective computational abilities*. Proceedings of the National Academy of Sciences of the USA, vol. 79 no. 8 pp. 2554-2558, April 1982.
6. Kohonen, T. – *Self-Organizing Maps*. Springer Series in Information Sciences, Vol. 30, Springer, Berlin, 1995.
7. Moise, A. – *Neural networks for pattern recognition*. MatrixRom Ed, Bucharest, 2005.
8. Pavlov, I. P. – *Conditioned Reflexes: An Investigation of the Physiological Activity of the Cerebral Cortex.* Translated and Edited by G. V. Anrep, London, Oxford University Press, 1927.
9. Samad, T. – *Back-propagation is significantly faster if the expected value of the source unit is used for update,* In International Neural Network Society Conference Abstracts, 1988.
10. Schmidt–Jones, C. – *Pitch: Sharp, Flat, and Natural Notes*. http://cnx.org/content/m1094.

Conversia unei imagini în sunet: o abordare neuronală

Rezumat

Partiturile muzicale au forme diferite, de la cele tipărite la cele scrise de mână. La rândul lor, cele scrise de mână arată foarte diferit una de cealaltă. Din acest motiv, un sistem care să convertească o astfel de partitură la o imagine standard şi/sau la un fişier de tip sunet care să poată fi ascultat este deosebit de util în domenii diverse: conducere automată, robotică, vedere artificială. Scopul urmărit în acest articol este dezvoltarea unei metode originale pentru a converti o partitură muzicală într-un fişier de tip sunet. În articol se prezintă în detaliu modul în care se poate proiecta o reţea neuronală artificială care să poată fi folosită pentru recunoaşterea notelor muzicale. Imaginea unei partituri este preluată cu o cameră web sau este încărcată direct dintr-un fişier. Una dintre contribuţiile importante ale autorilor este extragerea caracteristicilor notelor din partitură şi exportarea lor către reţeaua neuronală. Sunt prezentate exemple de secvenţe de program care realizează această operaţie. Reţeaua este antrenată şi sunt discutate comparativ rezultate experimentale obţinute cu diferite tipuri de neuroni şi diferiţi algoritmi de antrenare. Notele obţinute prin folosirea acestei metode sunt afişate şi partitura este interpretată automat. Articolul se încheie cu recomandări pentru cercetări ulterioare şi comentarii legate de metoda propusă.