Training a Multilayered Perceptron to Compute the Euler Number of a 2-D Binary Image

Humberto Sossa[1(✉)], Ángel Carreón[1], and Raúl Santiago[2]

[1] Instituto Politécnico Nacional-CIC,
Av. Juan de Dios Bátiz S/N, Gustavo a. Madero,
07738 Mexico City, Mexico
humbertosossa@gmail.com,
angelcarreon01@hotmail.com
[2] Instituto Tecnológico de León,
Av. Tecnológico S/N, Frac. Julián de Obregón,
León, Guanajuato, Mexico
rsantiago66@gmail.com

Abstract. In this short communication, we explain how a Multilayered Perceptron (MLP) can be used to compute the Euler number or Genus of a 2-D binary image. We take as basis the results provided by a mathematical formulation that is known providing exact results in the computation of this important topological image feature to derive two MLP-based architectures, one useful for the 4-connected case and one useful for 8-connected case. We present results with a set of realistic images and compare our proposals in terms of processing with other approaches reported in literature.

1 Introduction

The Euler number or Euler characteristic is a feature that allows describing the topological structure of an image or an specific object in an image. As it is known, the Euler number has been used in many applications: industrial part recognition [1], real-time thresholding [2], object number calculation, [3], and real-time Malayan license plate recognition [4], to mention a few.

Mathematically speaking, the Euler number, e, of a digital binary image $I(x, y)$ can be obtained as follows:

$$e = o - h \tag{1}$$

In this case, o is the number of objects or (binary regions) in the image and h is the number of holes (i.e., isolated regions of the image's background).

Many methods have been developed to obtain the Euler number of a digital binary image. Some of these methods, can be found in [5–25].

The algorithm outlined in [5] was one of the first reported in literature. The most popular algorithm of this method is used by the MATLAB image processing tool. It calculates the Euler number of a binary image as:

J.F. Martínez-Trinidad et al. (Eds.): MCPR 2016, LNCS 9703, pp. 44–53, 2016.
DOI: 10.1007/978-3-319-39393-3_5

$$e = \frac{s1 - s3 - 2 \cdot x}{4}. \tag{2}$$

In this case:

1. $s1$ is the number of matrices $\left\{ \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \right\}$;
2. $s3$ is the number of matrices $\left\{ \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \right\}$, and
3. x is the number of matrices $\left\{ \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right\}$.

As we can see, before using (2), the MATLAB algorithm needs to perform up to 10 comparisons on each image pixel. Time complexity for this method is of $O(N^2)$ for a $N \times N$ image, which is linearly dependent on the number of pixels. For image processing tasks, where the data could be huge the constant term that is so often hidden in the *big-Oh* notation becomes important.

Artificial Neural Networks (ANN), on the other hand, have been successfully used in many tasks including signal analysis, noise cancellation, model identification, process control, object detection, and pattern recognition, and so on. Many ANN models have been reported in literature, since the very simple Threshold Logic Unit (TLU), introduced by McCulloch-Pitts [26] at the beginning of the 40's, passing by the well-known Perceptron, presented to the world by Rosenblatt in the 50's [27, 28] until the so called Morphological Neural Models with and without Dendritic Processing introduced by Ritter et al. in [29, 30, 31, 32], to mention a few.

In this paper, we show how an MLP can be used to compute the Euler number of a 2-D binary image. By making an analysis of the local results provided by a known formulation to compute the image Euler number we arrive at the specialized ANN architecture. We decided to use a MLP for its versatility since many years ago in a multitude of situations. This is the first, to our knowledge, that a MLP-based architecture is used to compute the Euler number of a 2-D binary image. It constitutes and original an interesting option to compute this topological describing feature.

The rest of this paper is organized as follows. In Sect. 2 we describe our proposed methodology to derive at the end to the specialized MLP based architecture to compute the Euler number of a 2-D binary image. We devote Sect. 3 to report the experimental results that validate the applicability of the derived specialized MLP architecture as well as a comparison with other approaches reported in literature. In short, in Sect. 4 we reach our conclusions and directions for present and future research.

2 Our Proposal

In this section we describe how an MLP can be used to obtain the Euler number of a 2-D digital binary image. We decided it to do so, because as it is known MLPs have shown to be an excellent options to solve many problems in multiple areas where pattern classification is required.

To train a MLP to obtain the Euler number of a 2-D binary image we can proceed as usual by firstly selecting a set of P training samples, for example: $M \times N$ 2-D binary images: $\mathbf{I} = \{I_1, I_2, \ldots, I_P\}$. Before training the ANN, suppose we divide set $\mathbf{I}$ into q sub-sets of images such that each sub-set has the same Euler number, according to (1). With this in mind, we could proceed, for example, as illustrated in Fig. 1(a) by presenting, on the one hand, as input to a known Euler number computation method, that in turn outputs a correct value of e for each image: $I_k, k = 1, 2, \ldots, P$. As can be appreciated from this same figure, each image is also presented at the input of the untrained MLP, that produces a value: $\hat{e}$, as an estimate of e. The resulting error: E, could be then used, in an iterative way, to adjust the MLP weights until it is ready to compute the Euler number of an unknown input image as illustrated in Fig. 1(b).

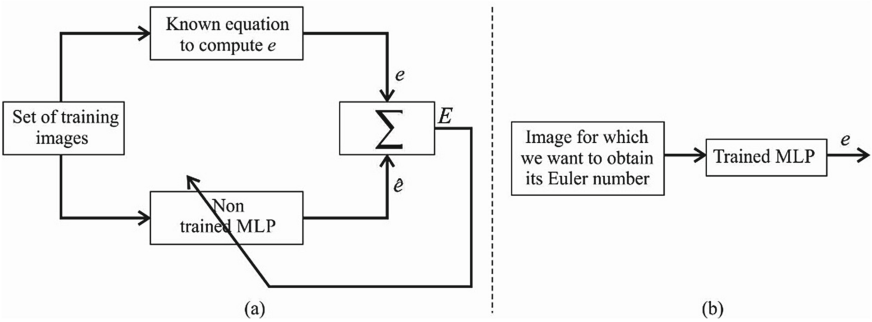

Fig. 1. A first alternative to train a MLP to compute the Euler number of a 2-D binary image. (a) Training of the MLP. (b) Testing of the MLP.

It is clear that if we apply the above described strategy, we would have several inconveniences. A first inconvenience would be the following: If we use a three layer MLP, the number of input neurons would be $M \times N$ (the image size); the number of output neurons would be directly proportional to the number of values NV of e $\{\ldots, -3, -2, -1, 0, 1, 2, 3, \ldots\}$ needed to be computed for an input image. In short, the correct number of hidden neurons to obtain the desired values for e would be certainly very big and rather difficult to find.

A second inconvenience would be that because the Euler number of an image is a function of the number of its objects and its holes, lots of training images would be required to reach good training results, this is because we have too many possibilities. Instead of using an architecture like this, we propose to derive a specialized one as follows. Let us first consider the following two expressions, introduced in [33] to compute the Euler number of a 2-D digital binary image in terms of only three comparisons. For the case of 4-connected regions (regions where their pixels are allowed to be connected only by their sides), the authors propose computing the image Euler number as:

$$e = \#\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} - \#\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} + \#\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{3}$$

On the other hand, for the case of 8-connected regions (regions where the pixels are allowed to be connected by their sides and corners), the authors propose to compute e by means of the following equation:

$$e = \#\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} - \#\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} - \#\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{4}$$

As referred in [33], these two equations seem to be the smallest expressions (in terms of the necessary operations) that allow computing the Euler number of a 2-D digital binary image, providing exact values as if (1) was used. To appreciate the validity of (3) and (4), let us consider the four academic examples shown in Fig. 2.

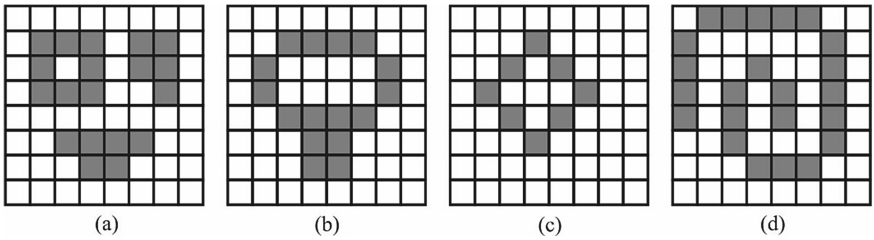

Fig. 2. Examples to numerically validate the functioning of (3) and (4).

Table 1. Application of (3) and (4) to the four example of Fig. 2.

	Image (a)	Image (b)	Image (c)	Image (d)
(3)	2	4	8	7
(4)	2	0	0	1

Table 1 summarizes the results obtained for these four binary images by means of (3) and (4). In the case of the image (a), all three binary regions, as can be appreciated, are 4-connected, thus the computed results by (3) and (4) are the same. In the case of image (b), some of the pixels are only 8-connected and some others are 4-connected, the reader can see that the obtained results are different by the application of both equations is different. In the case of image (c), when (3) is applied, all pixels are considered as disconnected, that is why an "8" is obtained, however when (4) is used a "0" is obtained due to for this equation the eight pixels as considered as a connected object with a hole. Finally, in the case of image (d), if 4-connectivity is considered, we see that we have seven connected regions; that is why by means of (3) we obtain a "7". On the other side, if 8-connectivity is considered, as can be appreciated from Fig. 2(d), all the pixels are taken as connected forming a spiral, thus the value for e in terms of (4) is "1", as expected.

Suppose now we want to design two specialized MLP architectures that allow computing the Euler number of a 2-D binary image, one based on (3) and the other based on (4). To accomplish this goal, let us represent the four numbers of each of each of the three terms of (3) and (4) by the four variables: v_1, v_2, v_3 and v_4. It is not difficult to see that the three arrangements used by (3) and (4) are three of the sixteen possibilities depicted in Table 2.

Table 2. Values for (3) and (4).

	v_1	v_2	v_3	v_4	Results for (3)	Results for (4)
1	0	0	0	0	0	0
2	0	0	0	1	0	0
3	0	0	1	0	0	0
4	0	0	1	1	0	0
5	0	1	0	0	0	0
6	0	1	0	1	0	0
7	0	1	1	0	0	-1
8	0	1	1	1	0	0
9	1	0	0	0	1	1
10	1	0	0	1	1	0
11	1	0	1	0	0	0
12	1	0	1	1	0	0
13	1	1	0	0	0	0
14	1	1	0	1	0	0
15	1	1	1	0	-1	-1
16	1	1	1	1	0	0

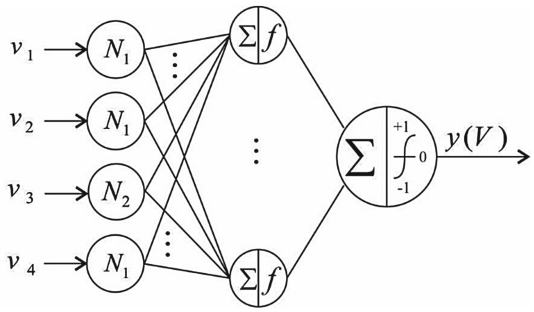

Fig. 3. Sketch of the specialized MLP architecture to compute the Euler number of a 2-D binary image.

From (3) we can also see that the first and third resulting values are both positive (rows 9 and 10), while the second term is negative (row 15). The remaining combinations, according to Table 2, sixth column, are zero. However, from (4) we can

appreciate that the first term is positive as shown in row 9, while the second and third terms are both negative as depicted in rows 7 and 15, respectively. Column 6 summarizes the results for (3), while column 7 resumes the results for (4).

In both cases, we propose to use these 16 values to train a MLP that allows producing as output one of the three values: $\{-1, 0, 1\}$, that in turn allow us determining the Euler number of a 2-D binary image in both cases of 4 and 8 connectivity. Figure 3 depicts a sketch of the specialized MLP architecture. It has four input neurons, nine hidden neurons and one output neuron. All activation functions of hidden neurons were chosen as sigmoidal, however the activation function for the output neuron is a hyperbolic tangent to approach at the end of training the three desired values: $\{-1, 0, 1\}$. For adjusting the connections weights among neurons, we have used standard Backpropagation rule with a learning rate of $\alpha = 0.1$. All the synaptic weights were initialized to random values between 0.0 and 1.0. At 5000 iterations good output values were obtained, but because these three values never correspond to the values of $-1, 0, 1$, we took output y of the MLP and if $y > 0.5$ then $r = 1$, else if $y \geq -0.5$ and $y \leq 0.5$ then $r = 0$ else if $y \leq -0.5$ then $r = -1$. Training was attained at 0.7442 s.

The reader can easily verify that all 16 values shown in Table 2 are correctly classified into their corresponding three classes; $-1, 0, 1$, in both cases of 4 and 8 connectivity. This guaranties that at the moment of computing the image Euler number by means of this specialized ANN, we will obtain a correct value as if (1) was used in both cases of 4 and 8-connected images.

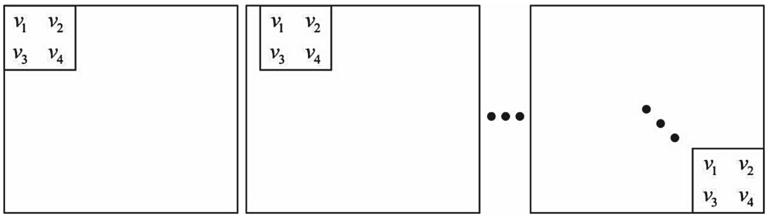

Fig. 4. Sequential way to apply (5) over an image.

To locally compute the Euler number of a 2-D binary image $I(x, y)$ we proceed sequentially as illustrated in Fig. 4. As can be seen from this figure, the generated MLP is displaced from the upper left position down to the right position inside the image, obtaining each time one of the three values: 1, 0 or 1. At the end of the application of this very simple procedure, we should obtain the Euler number of any 2-D binary image $I(x, y)$ as follows:

$$e = \sum_{k=1}^{T} r(k) \tag{5}$$

In this case, T is the number of times the trained MLP is applied to image $I(x, y)$; $r(k) = -1, 0, 1$ is the local result output by the MLP applied to a sector of $I(x, y)$.

3 Experimental Results

To numerically validate the correct functioning of the two derived ANNs, we first took 100 binary images of 256 × 256 pixels. In the case of the first the 50 images, objects are 4-connected; in the remaining 50 images objects are 8-connected. Due to space limitations, results for only 16 of these images are shown in Fig. 5. Table 3 depicts the values of e for these 16 images by the application of (5) as illustrated in Fig. 4. Only the results with the MLP for the 4-connected cases are shown in this table. As expected, in all cases, the correct Euler number for the 16 images was correctly calculated. The reader can demonstrate that for any other 2-D binary image, the desired e should be correctly computed.

All the experiments were run on a desktop computer with an Intel (R) Core(TM) i7 950 CPU 3.07 GHZ × 8 cores and 18 GB of RAM; operating system: Windows 7 Ultimate ×64.

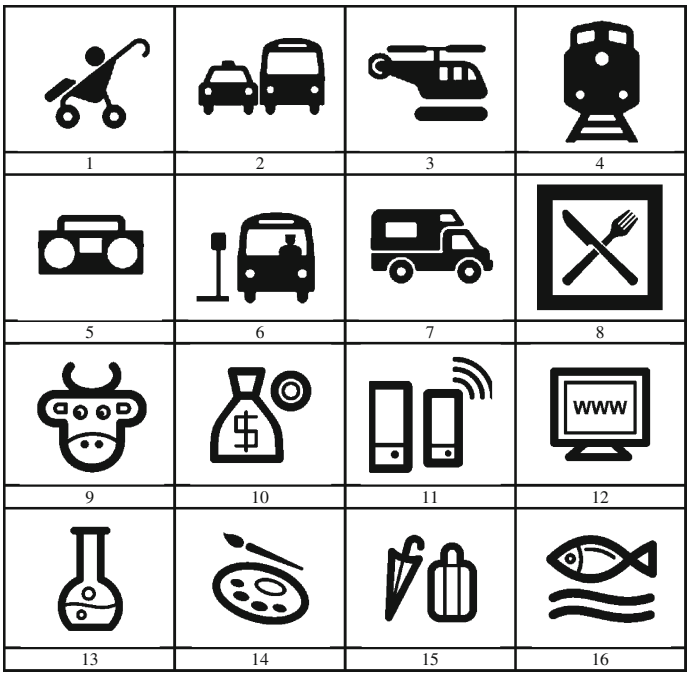

Fig. 5. 16 of the binary images used to validate the functioning of the derived MLPs.

Table 3. Values of e for the 16 images shown in Fig. 5.

Image 1	Image 2	Image 3	Image 4	Image 5	Image 6	Image 7	Image 8
$e = 1$	$e = -5$	$e = -2$	$e = -6$	$e = -3$	$e = -2$	$e = -1$	$e = 3$
Image 9	Image 10	Image 11	Image 12	Image 13	Image 14	Image 15	Image 16
$e = 1$	$e = -2$	$e = 3$	$e = 0$	$e = -1$	$e = 6$	$e = -5$	$e = 1$

Table 4. Cpmparisson with other proposals.

Method	Average time in seconds over the 100 images
Ref. [14]	14.00
Ref. [22]	18.00
Ref. [5]	20.00
Ref. [17]	73.00
MLP 4-connected case	78.08
MLP 8-connected case	73.52
Ref. [9]	151.00
Ref. [11]	1647.00

The average time to process an image of 256×256 pixels was of 0.7808 (4-connected case) and 0.7352 (8-connected case) seconds, respectively if the MLP is sequentially applied as depicted in Fig. 4. If (3) or (4) are applied over an image the average time reduces to 0.1557 and 0.1456 s, respectively. As can be seen more time is needed to obtain the Euler number of a binary image if the ANN based method is applied. This is normal due to more processing time is required to attain the same goal. What we want to show in this paper is that it is possible to compute the binary image Euler number by means of a Multi-layered Perceptron.

Compared with other standard formulations to compute the Euler number of a 2-D binary image reported in literature in terms of time, we observe in Table 4 that our proposals are, of course, not the fastest but neither the slowest. Both MLP-based proposals are slower that the methods reported in [5, 14, 22, 17] but faster that the methods reported in [9, 11].

Compared with other ANN implementations, such the one reported in [34] where a Morphological Neural Network with Dendritic Processing (MNNDP) is trained to accomplish the same task, the corresponding average times over the set of 100 images used in this paper were of 402.0 s (MNNDP based implementation) and 75.8 s (MLP based implementation), respectively.

4 Conclusions and Future Trends

In this paper we have shown that an MLP can be used to correctly determine the Euler number of a 2-D binary image. Through an analysis of the local operations implied in the application of known formulations, (3) and (4), we have derived a specialized architecture.

Although our proposed MLP based implementation is slower that the sequential implementation of (3) and (4), it constitutes an original and interesting alternative for the automatic computation of the 2-D binary image Euler number by means of an MLP.

Although our proposed methodology can be adapted to any ANN model, we have presented results with the MLP in both cases of 4 and 8 connectivity. A parallel work in this same direction with morphological neural networks is reported in [34].

Nowadays, we also are working toward the natural extension of our proposal in the case of 3-D binary images where objects will now be represented by voxels and not by pixels as usual.

We are also working to obtain an efficient implementation of the MLP to be run a GPU platform under CUDA. For this we are first implementing or MLP in matrix form. Because CUDA allows to manage the execution threats in matrix form, each execution threat can be a value inside a matrix, this way it will not be necessary to iterate over an image as illustrated in Fig. 4. Each threat will execute over each image pixel the necessary operations of lecture, its processing until the MLP output. In theory, the execution time will be equivalent to only iteration of the ANN.

Acknowledgements. The authors would like to thank IPN-CIC under project SIP 20151187 and 20161126, and CONACYT under projects 155014 and 65 within the framework of call: Frontiers of Science 2015, for the economic support to carry out this research. The second authors thanks CONACYT for the economic support to carry out his Master studies.

References

1. Yang, H.S., Sengupta, S.: Intelligent shape recognition for complex industrial tasks. IEEE Control Syst. Mag. **8**(3), 23–29 (1988)
2. Snidaro, L., Foresti, G.L.: Real-time thresholding with Euler numbers. Pattern Recogn. Lett. **24**, 1533–1544 (2003)
3. Lin, X., Ji, J., Gu, G.: The Euler number study of image and its application. In: Proceedings of 2nd IEEE Conference on Industrial Electronics and Applications (ICIEA 2007), pp. 910–912 (2007)
4. Al Faqheri, W., Mashohor, S.: A real-time Malaysian automatic license plate recognition (M-ALPR) using hybrid fuzzy. Int. J. Comput. Sci. Netw. Secur. **9**(2), 333–340 (2009)
5. Gray, S.B.: Local properties of binary images in two dimensions. IEEE Trans. Comput. **20**(5), 551–561 (1971)
6. Dyer, C.R.: Computing the Euler number of an image from its quadtree. Comput. Vis. Graph. Image Process **13**, 270–276 (1980)
7. Beri, H., Nef, W.: Algorithms for the Euler characteristic and related additive functionals of digital objects. Comput. Vis. Graph. Image Process **28**, 166–175 (1984)
8. Beri, H.: Computing the Euler characteristic and related additive functionals of digital objects from their beentree representation. Comput. Vis. Graph. Image Process **40**, 115–126 (1987)
9. Chen, M.H., Yan, P.F.: A fast algorithm to calculate the Euler number for binary images. Pattern Recogn. Lett. **8**(12), 295–297 (1988)
10. Chiavetta, F., Di Gesú, V.: Parallel computation of the Euler number via connectivity graph. Pattern Recogn. Lett. **14**(11), 849–859 (1993)
11. Díaz de León S., J.L., Sossa, H.: On the computation of the Euler number of a binary object. Pattern Recogn. **29**(3), 471–476 (1996)
12. Bribiesca, E.: Computation of the Euler number using the contact perimeter. Comput. Math Appl. **60**, 1364–1373 (2010)
13. Sossa, H., Cuevas, E., Zaldivar, D.: Computation of the Euler number of a binary image composed of hexagonal cells. J. Appl. Res. Technol. **8**(3), 340–351 (2010)

14. Sossa, H., Cuevas, E., Zaldivar, D.: Alternative way to compute the Euler number of a binary image. J. Appl. Res. Technol. **9**(3), 335–341 (2011)
15. Imiya, A., Eckhardt, U.: The Euler characteristics of discrete objects and discrete quasi-objects. Comput. Vis. Image Underst. **75**(3), 307–318 (1999)
16. Kiderlen, M.: Estimating the Euler characteristic of a planar set from a digital image. J. Vis. Commun. Image Represent. **17**(6), 1237–1255 (2006)
17. Di Zenzo, S., Cinque, L., Levialdi, S.: Run-based algorithms for binary image analysis and processing. IEEE Trans. Pattern Anal. Mach. Intell. **18**(1), 83–89 (1996)
18. Sossa, H., Cuevas, E., Zaldivar, D.: Computation of the Euler number of a binary image composed of hexagonal cells. JART **8**(3), 340–351 (2010)
19. Sossa, H., Rubio, E., Peña, A., Cuevas, E., Santiago, R.: Alternative formulations to compute the binary shape Euler number. IET-Comput. Vis. **8**(3), 171–181 (2014)
20. Yao, B., Wu, H., Yang, Y., Chao, Y., He, L.: An Improvement on the Euler number computing algorithm used in MATLAB. In: IEEE Region 10 Conference on TECNON 2013–2013, Xi'an, China, 22–25 October 2013
21. He, L., Chao, Y., Suzuki, K.: A linear-time two-scan labelling algorithm. In: Proceedings of IEEE International Conference on Image Processing (ICIP 2007), pp. V-241–V-244, San Antonio, TX, USA, September 2007
22. He, L.F., Chao, Y.Y., Susuki, K.: An algorithm for connected-component labeling, hole labeling and euler number computing. J. Comput. Sci. Technol. **28**(3), 468–478 (2013)
23. He, L., Chao, Y.: A very fast algorithm for simultaneously performing connected-component labeling and Euler number computing. IEEE Trans. Image Process. **24**(9), 2725–2735 (2015)
24. Yao, B., He, L., Kang, S., Chao, Y., Zhao, X.: A novel bit-quad-based Euler number computing algorithm. SpringerPlus **4**(735), 1–16 (2015)
25. Yao, B., Kang, S., Zhao, X., Chao, Y., He, L.: A graph-theory-based Euler number computing algorithm. In: Proceeding of the 2015 IEEE International Conference on Information and Automation, pp. 1206–1209, Lijiang, China, August 2015
26. MuCulloch, W.S., Pitts, W.H.: A logical calculus of the ideas immanent in nervous activity. Bull. Math. Biophys. **5**, 115–133 (1943)
27. Rosenblatt, F.: The perceptron: a probabilistic model for information storage and organization in the brain. Psychol. Rev. **65**, 386–408 (1958)
28. Rosenblatt, F.: Principles of Neurodynamics: Perceptron and Theory of Brain Mechanisms. Spartan, Washington, DC (1962)
29. Ritter, G.X., Sussner, P.: An introduction to morphological neural networks. In: Proceedings of the 13th International Conference on Pattern Recognition, vol. 4, pp. 709–717 (1996)
30. Sussner, P.: Morphological perceptron learning. In: IEEE ISIC/CIRA/ISAS Joint Conference, pp. 477–482 (1998)
31. Ritter, G.X., Beaver, T.W.: Morphological perceptrons. Int. Joint Conf. Neural Netw. **1**, 605–610 (1999)
32. Ritter, G.X., Iancu, L., Urcid, G.: Morphological perceptrons with dendritic structure. In: 12th IEEE International Conference in Fuzzy Systems (FUZZ 2003), vol. 2, pp. 1296–1301 (2003)
33. Sossa, H., et al.: 2-D Binary Image Efficient Euler Number Computation, Paper under preparation
34. Sossa, H., Carreón, A., Guevara, E., Santiago, R.: Computing the 2-D image Euler number by an artificial neural network. In: Accepted to Be Presented at IJCNN 2006, Vancouver, Canada, 24–29 July 2016